The ISC Int'l Journal of Information Security

July 2018, Volume 10, Number 2 (pp. 141–162)

http://www.isecure-journal.org

Feature-based Malicious URL and Attack Type Detection Using Multi-class Classification

Dharmaraj R. Patil [1,*], and Jayantrao B. Patil [1]

[1] *Department of Computer Engineering, R. C. Patel Institute of Technology, Shirpur-425405, India*

ARTICLE INFO.

Article history:
Received: 7 January 2018
First Revised: 20 April 2018
Last Revised: 15 March 2018
Accepted: 18 March 2018
Published Online: 20 March 2018

Keywords:
Malicious URLs, Feature Extraction, Multi-class Classification, Batch and Online Learning, Web Security.

ABSTRACT

Nowadays, malicious URLs are the common threat to the businesses, social networks, net-banking etc. Existing approaches have focused on binary detection i.e. either the URL is malicious or benign. Very few literature is found which focused on the detection of malicious URLs and their attack types. Hence, it becomes necessary to know the attack type and adopt an effective countermeasure. This paper proposed a methodology to detect malicious URLs and the type of attacks based on multi-class classification. In this work, we proposed 42 new features of spam, phishing and malware URLs. These features are not considered in the earlier studies for malicious URLs detection and attack types identification. Binary and multi-class dataset is constructed using 49935 malicious and benign URLs. It consists of 26041 benign and 23894 malicious URLs containing 11297 malware, 8976 phishing and 3621 spam URLs. To evaluate the proposed approach, state-of-the-art supervised batch and online machine learning classifiers are used. Experiments are performed on the binary and multi-class dataset using the aforementioned machine learning classifiers. It is found that, confidence weighted learning classifier achieved the best 98.44% average detection accuracy with 1.56% error-rate in the multi-class setting and 99.86% detection accuracy with negligible error-rate of 0.14% in binary setting using our proposed URL features.

1 Introduction

Day by day the World Wide Web becomes victim of Web attacks like spamming, phishing and malware. When the innocent user unknowingly visits the URL, it becomes the victim of the attacks. For example, phishers send an email to the victim seems that it is from trustworthy sender containing an URL. When the innocent user clicks that URL, landed to the vulnerable Website. According to the recent Symantec monthly Internet Security Threat Report 2017 [1],

- The email malware rate increased slightly in August to one in 347 emails. This is the highest rate of activity seen since December 2016.
- The global spam rate crossed the 55% threshold in August, reaching 55.3%. This is the highest spam rate seen since March 2015.
- The phishing rate decreased in August, down to one in 2,493 emails.
- The number of web attacks blocked increased slightly in August, up from 1,159,000 per day to 1,241,000 per day.

* Corresponding author.
Email addresses: drpatil@rcpit.ac.in (D. R. Patil), jbpatil@rcpit.ac.in (J. B. Patil)
ISSN: 2008-2045

- Fake offers topped social media scams, comprising almost 47% of scams in August.

Also, according to the recent APWG Phishing Activity Trends Report analysis, there is a constant stream of phishing and confirmed attack sites in the first half of 2017 [2],

- Phishing attacks occurred most frequently in the payment, financial, and Webmail sectors.
- There has been an increase in the number of phishing attacks using free hosting providers or website builders.
- Several hundred companies are being targeted regularly.

According to the above two Security Threat reports, it is clear that there is a constant increase in Web attacks in the form of spams, phishing and malware distribution, which causes serious impact on business, banking, social networks and innocent users. In [3], the authors have presented a detail survey on various types of malicious Webpage attacks like, drive-by downloads, clickjacking, plug-ins and script-enabled, phishing, social networks, cross-site scripting (XSS), SQL injection, third-party Web apps, JavaScript obfuscation etc. To address Web-based attacks detection, researchers all over the world have taken great efforts in the last few years. Following are the common approaches for Web-based attacks detection [4–13],

- Blacklisting-based.
- Static analysis.
- Dynamic analysis.
- Heuristic-based.

A common practice is to use a blacklist of malicious URLs, which can be constructed from various sources, particularly human feedbacks which are highly accurate. The drawback of blacklisting-based approach is, the unknown URLs are not detected. According to [14, 15], they performed static analysis of the URLs and the JavaScript code in the Webpages for the detection of malicious URLs and JavaScripts. According to them, experimental results show that static analysis is a fast analysis technique with promising results, but still it cannot cope up with todays ever-changing dynamic nature of Web attacks. Although the above approaches are promising in the malicious URLs detection, but still focused to binary detection only i.e. the URL is either malicious or benign.

Importance of knowing Web attack type: Detection of attack types is useful since the knowledge of potential threat allows us to take a proper action as well as effective countermeasure against the threat. For example, we may ignore spam emails in our mailbox but immediately react if the browser flash malware warning by applying countermeasure against the attack. In other words, today it becomes necessary to identify attack types to protect and create awareness in innocent Web users. Very few literature have focused on the detection of malicious URLs and their attack types [16].

This paper proposed a methodology to detect malicious URLs and the type of attacks based on multi-class classification techniques. The set of discriminative features are adopted related to URLs, word-based, domain name, Webpage source and short URLs. We used 117 static and dynamic features, among which 42 are novel features. A binary and multi-class dataset of 49935 malicious and benign URLs is constructed. It consists of 26041 benign and 23894 malicious URLs containing 11297 malware, 8976 phishing and 3621 spam URLs. To evaluate our approach, we have used the supervised batch and online machine learning classifiers. Experiments are performed on our binary and multi-class dataset using the aforementioned machine learning classifiers. It is found that, confidence weighted learning classifier achieved the best 98.44% average detection accuracy with 1.56% error-rate in the multi-class setting and 99.86% detection accuracy with negligible error-rate of 0.14% in binary setting.

The major contributions of this paper are as follows:

- We proposed 42 new features of spam, phishing and malware URLs like URL Features, URL Source Features, Domain Name Features and Short URLs Features. These features improved the detection performance and attack-type identification of malicious URLs of main attack types including spamming, phishing and malware distribution. As per our knowledge these features are not proposed yet.
- The proposed methodology outperforms the work by [16] in the sense that attack type identification, micro TP and macro TP measures are improved using multi-class confidence weighted learning classifier on our multi-class URL dataset.
- The dataset is constructed including short URLs which is not considered in the earlier studies.

The remainder of this paper is organized as follows. Section 2 gives a brief related work. Section 3 describes the methodology with feature extraction and supervised multi-class batch and online machine learning algorithms. Section 4 describes the experimental results and discussion. Section 5 gives limitations of our system. We present our conclusions in Section 6.

2 Related Work

Various approaches which have been proposed for malicious URLs detection. In this section, we will review few state of the art approaches briefly. Following are some of the approaches researchers have used for the malicious URLs detection, which are described below.

- Machine Learning-based Approaches
- Non-machine Learning-based Approaches
- Neural Network-based Approaches
- Behaviour-based Detection Approaches

2.1 Machine Learning-based Approaches

In [17], the authors have proposed a method of phishing website detection that utilizes a meta-heuristic-based nonlinear regression algorithm together with a feature selection approach. To validate the proposed method, they used a dataset comprised of 11055 phishing and legitimate webpages and selected 20 features to be extracted from the mentioned websites. They used two feature selection methods: decision tree and wrapper to select the best feature subset and obtained the detection accuracy rate as high as 96.32% using wrapper method. After the feature selection process, they implemented two meta-heuristic algorithms to predict and detect the fraudulent websites like, harmony search (HS) which was deployed based on nonlinear regression technique and support vector machine (SVM). According to them, the nonlinear regression approach was used to classify the websites, where the parameters of the proposed regression model were obtained using HS algorithm. The experimental analysis show that, the nonlinear regression based on HS led to accuracy rates of 94.13 and 92.80% for train and test processes, respectively. The comparative performance analysis show that, the nonlinear regression-based HS results in better performance compared to SVM.

In [18], the authors have proposed a set of 58 new Webpage hybrid features and refine them as few, maximum relevant, minimum redundant, and robust features as possible. These features, they have specifically extracted from two different sources: webpages URL and content. Thus, they have taken two feature categories into consideration through the experiments. The first feature category is a group of 48 features mostly including cross site scripting and embedded objects features, whereas the second feature category is a group of 10 URL features extracted from webpages URL. To identify a optimal feature subset for effective phishing detection, they used a specific criterion *i.e.* mRMR. According to them, mRMR removes irrelevant and redundant features simultaneously over a high dimensional feature space. They used SVM machine learning classifier and assessment criteria like TP, FP, FN, Precision, Recall and F-measure to evaluate their approach. The experimental demonstration shows that, their approach could be used to optimize a phish detection model for any anti-phishing scheme in the future.

A study [16] has proposed a method using machine learning to detect malicious URLs of all the popular attack types like spam, phishing, malware etc. and to identify the nature of attack a malicious URL attempts to launch. They have used features like lexical, link popularity, Webpage content, DNS, DNS fluxiness and network traffic. They have collected real-life data from various sources like benign URLs from DMOZ Open Directory Project, Yahoo!s directory, Spam URLs from jwSpamSpy, Web spam dataset, Phishing URLs from PhishTank and Malware URLs from DNS-BH. They have used three machine learning algorithms like the Support Vector Machine (SVM) to detect malicious URLs, RAkEL and ML-kNN learning algorithms for multi-label classification problem to identify attack type. They have evaluated their method on 40,000 benign URLs and 32,000 malicious URLs and achieved the accuracy of 98% in detection of malicious URLs and 93% in identification of attack types.

In [19] the authors have presented the formal formulation of Malicious URL Detection as a machine learning task, and categorized and reviewed the contributions of literature studies that address different dimensions of this problem (feature representation, algorithm design, etc.). They have provided a timely and comprehensive survey for a range of different audiences, not only for machine learning researchers and engineers in academia, but also for professionals and practitioners in cyber security industry, to help them understand the state of the art and facilitate their own research and practical applications. They also discussed practical issues in system design, open research challenges and pointed out some important directions for future research.

In [5] the authors have proposed a filter, called Prophiler that uses static analysis techniques to quickly examine a Web page for malicious content. They have used features derived from the HTML contents of a page, from the associated JavaScript code and from the corresponding URL. They used different machine learning algorithms like Random Tree, Random Forest, Naive Bayes, Logistic, J48 Bayes Net and Logistic regression for evaluation. According to them, their filtering approach is able to reduce the load on a more costly dynamic analysis tools i.e. Wepawet by more than 85%, with a negligible amount of missed malicious.

An approach based on automated URL classifica-

tion, using statistical methods to discover the lexical and host-based properties of malicious Web site URLs is proposed by [4]. They have extracted the Lexical features and Host-based features. The host-based features include IP address properties, WHOIS properties, domain name properties and geographic properties. They have used machine learning algorithms like Naive Bayes, Support Vector Machine (SVM) and Logistic Regression for evaluation. According to them, the resulting classifiers obtain 95-99% accuracy, detecting large numbers of malicious Web sites from URLs, with only modest false positives.

Monarch, a real-time system that crawls URLs as they are submitted to Web services and determines whether the URLs direct to spam is presented by [20]. They show that Monarch can provide accurate, real-time protection. They have used Monarchs feature collection infrastructure over the course of two months to crawl 1.25 million spam email URLs, roughly 567,000 blacklisted Twitter URLs and over 9 million non-spam Twitter URLs. They have evaluated the accuracy of Logistic Regression with L1-regularization classifier and its run-time performance. Their experimental results show that they can identify Web service spam with 90.78% accuracy and 0.87% false positives, with a median feature collection and classification time of 5.54 seconds.

A novel approach for detection and analysis of malicious JavaScript code that leads to perform drive-by-download attack on the victim's machine is proposed by [21]. They have combined anomaly detection with emulation to automatically identify malicious JavaScript code. They have extracted features like redirection and cloaking, deobfuscation, environment preparation and exploitation. They have used htmlunit browser for rendering of the Web pages. They have developed a system that uses a number of features and machine learning techniques to establish the characteristics of normal JavaScript code. The evaluation results show that it is possible to reliably detect malicious code by using emulation to exercise the behavior of the code and comparing this behavior with a model of normal JavaScript code execution.

A lightweight approach, called BINSPECT that combines static analysis and emulation is presented by [6]. They have used supervised learning techniques in detection of malicious Web pages that may launch drive-by- download, phishing, injection and malware distribution attacks. They have extracted features like URL features, page- source features and social-reputation features. They have collected a malicious dataset of 71,919 URLs from the malware and phishing blacklist of Google, Phishtank database and the malware and injection attack URL list of MalwareURL. The benign dataset of 414,000 benign URLs is collected from three popular sources like the Alexa Top sites, the Yahoo random URL generation service and the DMOZ directory. According to their experimental evaluation, BINSPECT achieved 97% accuracy with low false signals.

A machine learning based approach is proposed by [7] to detect phishing Web pages. They have used many novel content based features and applied cutting-edge machine learning techniques such as 6 batch learning algorithms, Random Forests, Support Vector Machines (SVM) with rbf linear kernels, Naive Bayes, C4.5, Logistic Regression (LR) and a set of 5 online learning algorithms: updatable version of Naive Bayes (NB-U), updatable version of LogitBoost (LB-U), Perceptron, Passive- Aggressive (PA) and Confidence-Weighted (CW) algorithms. They have used 179 Web page features such as lexical based features, keyword based features, search engine based feature and reputation based features to demonstrate their approach. To conduct all the experiments, they used WEKA and CW libraries. The experimental results show that their proposed approach can detect phishing Webpages with an accuracy of 99.9%, false positive rate of as low as 0.00% and false negative rate of 0.06%.

In the recent study by [22], the authors have proposed and analyzed a novel set of features including HTML, JavaScript (jQuery library) and XSS attacks. They have evaluated the proposed features on a data set gathered by a crawler from malicious web domains, IP address and black lists. They used a number of machine learning algorithms for the purpose of evaluation. According to them, the experimental results show that by using the proposed set of features, the C4.5-Tree algorithm offers the best performance with 97.61% accuracy, and F1-measure has 96.75% accuracy. Also, they performed ranking of the features. According to their ranking results, they suggested that nine of the proposed features are among the twenty best discriminative features.

2.2 Nonmachine Learning-based Approaches

ADSandbox, an analysis system for malicious Websites that focuses on detecting attacks through JavaScript is presented by [23]. They have employed a novel concept of a client-side JavaScript sandbox. According to them, this approach combines generality with usability, since the system is executed directly on the client running the Web browser before the Web page is displayed. According to them, the experimental results show that, they can achieve false positive rates close to 0% and false negative rates below 15%

with a performance overhead of only a few seconds.

In [24], the authors have provided a new method to detect malware distribution network (MDN) from the secondary URLs and redirect chains recorded by a high-interaction client honeypot. Also, they have proposed a novel drive-by download detection method. According to them, instead of depending on the malicious content, their algorithm first identifies and then leverages the URLs of the MDN's central servers, where a central server is a common server shared by a large percentage of the drive-by download attacks in the same MDN. They have generated signatures for the drive-by-downloads using regular expressions. According to them, the experimental results demonstrated the effectiveness of their method, where the detection rate is 96% with low false positive rate.

A suspicious URL detection system for Twitter, "WARNINGBIRD" have been proposed by [25]. They considered correlated redirect chains of URLs in a number of tweets. According to them, attackers have limited resources and thus have to reuse them i.e a portion of their redirect chains will be shared. Hence, they focused on these shared resources to detect suspicious URLs. They have collected a large number of tweets from the Twitter public timeline and trained a statistical classifier with features derived from correlated URLs and tweet context information. The experimental results show that, they have achieved high accuracy and low false-positive and false negative rates.

In another recent study by [26], the authors have proposed a supervised feature extraction method, called weighted feature line embedding, to solve the problems of phishing website detection. According to them, the proposed method virtually generates training samples by utilizing the feature line metric. The experimental results show that, the features extracted by their method improves the performance of phishing website detection specially by using small training sets.

The authors of [27] have provided a multilayer model to detect phishing, PhiDMA (Phishing Detection using Multi-filter Approach). According to them PhiDMA model incorporates five layers: Auto upgrade whitelist layer, URL features layer, Lexical signature layer, String matching layer and Accessibility Score comparison layer. They have implemented a prototype of PhiDMA model for persons with visual impairments. According to their experiment results, they shows that the model is capable to detect phishing sites with an accuracy of 92.72%.

2.3 Neural Network-based Approaches

In the recent work by [28], the authors have evaluated various deep learning architectures specially recurrent neural network (RNN), identity-recurrent neural network (I-RNN), long short-term memory (LSTM), convolution neural network (CNN), and convolutional neural network-long short-term memory (CNN-LSTM) architectures for the task of malicious URLs detection. They have conducted various experiments with various configurations of network parameters and network structures, to find the optimal parameters for deep learning architecture. All the experiments run till 1000 epochs with a learning rate in the range [0.01-0.5]. According to them, deep learning mechanisms outperformed the hand crafted feature mechanism. Specically, LSTM and hybrid network of CNN and LSTM have achieved highest accuracy as 0.9996 and 0.9995 respectively.

In [29], the authors have proposed a novel framework which combines a neural network with reinforcement learning to detect phishing attacks in the online mode for the first time. According to them, the proposed model has the ability to adapt itself to produce a new phishing email detection system that reflects changes in newly explored behaviors, which is accomplished by adopting the idea of reinforcement learning to enhance the system dynamically over time. This proposed model solves the problem of limited dataset by automatically adding more emails to the offline dataset in the online mode. Their experimental demonstration show that, the proposed technique can handle zero-day phishing attacks with high performance levels achieving high accuracy, TPR and TNR at 98.63%, 99.07%, and 98.19% respectively. In addition, they obtained low FPR and FNR, at 1.81% and 0.93% respectively. Also, they performed comparison with other similar techniques on the same dataset and shows that the proposed model outperforms the existing methods.

2.4 Behaviour-based Detection Approaches

Another recent work by [30], the authors have presented an in-depth empirical study conducted based on 1,529,433 malicious URLs collected over the past two years. They have analyzed attackers' tactical behavior regarding URLs and extracted common features and divided them into three different feature pools to determine the level of compromise of unknown URLs. To leverage detection rates, they employed a similarity matching technique. According to them, new URLs can be identified through attackers' habitual URL manipulation behaviors. The experimental analysis show that, they have achieved accuracy up to 70% with attributes of URLs to be exam-

ined. They show that, this model can be utilized during preprocessing to determine whether input URLs are benign, and as a web filter or a risk-level scaler to estimate whether a URL is malicious.

Considering the classification based malicious URLs detection techniques, Table 1 gives the comparative study of previous work done on malicious URLs detection. Table 2 gives the list of acronyms used in Table 1.

A comparative study of previous work for malicious URLs detection based on key features like URL features, URL source features, domain name features and short URLs features and their advantages and limitations are given in Table 3.

3 Methodology

3.1 Framework of our proposed multi-class malicious URLs detection system

The framework of our proposed multi-class malicious URLs detection system is shown in Figure 1. Our methodology consists of three steps like, feature extraction, preprocessing and labeling, training based on multi-class batch and online learning algorithms and detection and attack type identification.

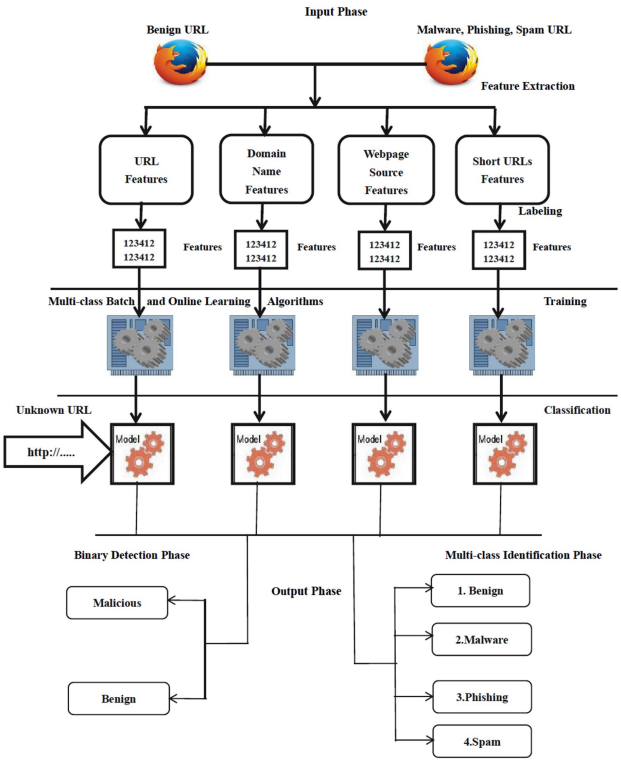

Figure 1. Framework of our proposed multi-class Malicious URLs Detection System

3.2 Feature extraction, preprocessing and labeling

The raw malware, phishing, spamming and benign URLs from benchmarks sources are fed to the feature extraction module written in Java. We have extracted 117 static and dynamic features of the benign, malware, phishing and spamming URLs. These are numeric and binary features. In binary dataset preparation malicious URLs labeled as +1 and benign URLs labeled as -1. In multi-class dataset preparation, we have labeled the benign URLs as 1, malware URLs as 2, phishing URLs as 3 and spam URLs as 4.

We have extracted four types of static and dynamic URL features like, URL features, domain name features, URL source features and short URLs features. We have implemented a URL feature extractor in Java. The URL feature extraction is implemented based on the URL class of Java and the features are collected by lexical scanning of the URL string. The domain name features extraction is implemented based on the domain name extraction and scanning of the domain name. The URL source features are collected by visiting the page via Selenium WebDriver [31] and an instance of Firefox browser so as to capture the run time details of what is rendered (HTML) using a feature extraction engine implemented in Java. For each URL visit for feature extraction, a fresh instance of the Firefox browser is launched to ensure a unique session for each URL. The short URLs features are extracted by checking the domain names containing the major URL shortening services like *bit.ly, goo.gl, tinyurl.com, owl.ly, deck.ly, su.pr, bit.do etc. The expanded URLs are obtained by making query to the URL shortening services. After getting the original URL from URL shortening services, we have set a threshold value of 30 for the length of URLs i.e. if the length of the returned URL is >=30, it is marked as malicious. Also, we have checked the lexical properties of the returned URL string for deciding it as benign or malicious. We have checked the returned URL string for containing suspicious lexical characters like, _, =, (,), %, & and @.*

3.2.1 URL Features

We have extracted 63 URL features of the URL string. Among these features 47 are from the literature [4–10, 16, 18] and 16 are new features. These are the lexical properties of the URLs. Lexical features are the textual properties of the URL itself. The URL features are given in Table 4 and 5. In addition to the lexical features, we have checked the presence of suspicious words in the URLs. These are numeric and binary features. These features are important to differentiate malicious URLs from benign ones.

Table 1. Comparative study of prior work

Related Work	Classification/Regression Technique	Feature Selection Techniques	Detected Classes
[28]	RNN, I-RNN, LSTM, CNN,CNN-LSTM	NA	Malicious/Benign URLs
[30]	Fuzzy	NA	Malicious/Benign URLs
[17]	HS, SVM	DT, WFS	Phish/Benign Websites
[29]	NN	NA	Phish/Benign Websites
[18]	SVM	mRMR	Phish/Benign Websites
[16]	SVM, RAkEL, ML-kNN	NA	Malicious/Phishing/Spam/Benign URLs
[5]	RT, RF, NB, J48, Bayes Net, LR	NA	Malicious/Benign Websites
[27]	String matching, Scoring Algorithm	NA	Phish/Benign Websites
[4]	NB, SVM, LR	NA	Malicious/Benign URLs
[20]	LR	NA	Spam/non-spam URLs
[21]	libAnomaly	NA	Malicious/Benign JavaScript Code
[6]	J48, RT, RF,NB, Bayes Net, SVM, LR, CW Majority Vote	NA	Malicious/Benign Websites
[7]	RF, SVM, NB, C4.5, LR, NB-U, LB-U,Perceptron, PA, CW	NA	Phish/Benign Websites
[25]	statistical classifier	NA	Malicious/Benign URLs
[22]	Multi-layer perceptron, NB, SVM, KNN, AD-Tree, BFTree,C4.5	Entropy Based, Gain Ratio, correlation coefficient square, TOPSIS	Malicious/Benign Websites
[26]	WFLE	NA	Phish/Benign Websites
[8]	LR	NA	Phish/Benign Websites
[10]	P, AP,PA, CW, AROW	NA	Malicious/Benign URLs
[12]	Gradient Boosting	NA	Phish/Benign Websites
[13]	NB, RF, SVM,LR	NA	Malicious/Benign URLs

Following is the example of phishing URL, which depicts some of the above mentioned URL features like, the multiple occurrence of suspicious symbols like %, -, &, digits in the URL, URL without "www" and presence of words like .php, login which generally not appear in benign URLs.

`http://unblocking-fb.site/contact-us/ref/index%20-%201.php?=10065877425?fb_source=bookmark_apps&ref=bookmarks&count=0&fb_bmpos=login_failed`

- *Entropy of URLs*

To demonstrate the randomness factor in URLs, we used Shannon Entropy as a measure: higher the entropy, higher is the randomness of the instance under consideration [10]. We calculated the entropy measure of each benign and malicious URL separately. The Shannon entropy of the URL string is calculated using following equation,

$$H(x) = -\sum_{i=0}^{n} p(x_i) \log_b p(x_i) \quad (1)$$

where,
$H(x)$ is the Shannon entropy of string x,
b is the base of the logarithm used and
$p(x)$ is the probability mass function.

From the Table 6 it is clear that, malicious URLs have high entropy as compare to benign URLs. It shows that there is more randomness factor in malicious URLs, to mark it as malicious.

- *Suspicious Word Based features of the URLs*

We added 7 new suspicious words in the URL feature set. The word-based features are binary. We tested if the given word is present or absent in a URL. We have used string matching algorithm by Knuth-Morris-Pratt (KMP) to find the presence or absence of the suspicious word in the URL [32].

Table 2. List of Acronyms used in Table 1

RNN: Recurrent Neural Network
I-RNN: Identity-Recurrent Neural Network
LSTM: Long Short-term Memory
CNN: Convolution Neural Network
CNN-LSTM: Convolutional Neural Network-long Short-term Memory
HS: Harmony Search
SVM: Support Vector Machine
DT: Decision Tree
WFS: Wrapper-based Feature Selection
NN: Neural Network
mRMR: Maximum Relevant Minimum Redundant
RAkEL: RAndom k-labELsets
ML-kNN: Multi-label K-nearest neighbor
RT: Random Tree
RF: Random Forest
NB: Naive Bayes
J48: J48 Decision Tree
LR: Logistic Regression
CW: Confidence Weighted
NB-U: Naive Bayes Updatable
LB-U: LogitBoost Updatable
PA: Passive Aggressive
KNN: K-nearest neighbor
AD-Tree: Alternating Decision Tree
BFTree: Best First Tree
TOPSIS: Technique for Order Preference by Similarity to Ideal Solution
WFLE: Weighted Feature Line Embedding
AP: Averaged Perceptron
AROW: Adaptive Regularization of Weight Vectors

Motivation behind categorizing some words as suspicious: Malicious URLs contain some suggestive word tokens such as *secure, account, webscr, login, ebayisapi, signin, banking, confirm, blog, logon, signon, login.asp, login.php, login.htm, .exe, .zip, .rar, .jpg, .gif, viewer.php, link=, getImage.asp, plugins, paypal, order, dbsys.php, config.bin, download.php, .js, payment, files, css, shopping, mail.php, .jar, .swf, .cgi, .php, abuse, admin, .bin, personal, update, verification.* We checked the presence of these security sensitive words and included as the binary features in our dataset. Following are the examples of URLs which depicts the presence of such one or more suspicious words.

The attackers include these suspicious words in the URLs to deceive legitimate users, such as including *login, login.html, login.php, signin, logon, signon*, or *.exe, .zip, .rar, .jpg, .js, .gif,. bin, .jar* to carry malware, or *.php, .js* for sending spams and other illegal activities.

- *Malware URL*

`http://files.appsapi.info/updates/ts/ts_5/update1.exe?random=57933`

- *Phishing URL*

`http://unblocking-fb.site/contact-us/ref/index%20-%201.php?=10065877425?fb_source=bookmark_apps&ref=bookmarks&count=0&fb_bmpos=login_failed`

- *Spam URL*

`http://labor.vermont.gov/LinkClick.aspx?link=http://workforprofit.net/jobs/?wwvxo`

In Table 7, we given the frequency distribution of our proposed security sensitive (suspicious) words in benign and malicious URLs. It indicates that the frequency of suspicious words in the malicious URLs is higher than that of benign URLs. Hence, these features help to identify malicious URLs from benign URLs.

3.2.2 URL Source Features

For the effective detection of malicious URLs, we used the URL source features. We rendered the URLs with the help of Selenium WebDriver [31] and an instance of Firefox browser. We have written a script in Java and Selenium WebDriver, which extracts the URL source features. We have extracted 34 such features among which 21 features are taken from literature [5–8, 11, 16, 18, 22] and 13 are new features. These are numeric, binary and real value features. These features are given in Table 10

The attackers also modify the URL source code to hide their identity and embed suspicious tags, for example *iFRame, img, input, applet* etc., sensitive words like *pay, free, access, bonus, click* etc. to deceive the innocent users. According to [33], $iframes$ are used for loading dynamic content in a Webpage from other source. In $iframe$ based attacks attackers need not have to compromise the servers serving advertisements or other Web contents. Attackers may successful to redirect the users using malicious $iframe$ embedded in a Webpage. Hence attackers usually embed hidden $iframes$ into the page to deceive a user redirect to a malicious page.

Table 3. Comparison of Different Feature Types

Feature Type	Advantages	Limitations
URL features [4], [10], [13], [14], [16], [28], [30], [20], [25]	• Easy feature extraction • Fast processing • Less system overhead • Better for static analysis of URLs • Widely used in the literature	• Not able to detect todays ever-changing attacks • Not able to identify other attacks only suitable for phishing attacks • Not able to detect run-time behavior of attacker
URL source features [5], [6],[7], [8], [12], [13],[15],[17],[29], [18], [27], [21], [22], [26]	• Deep analysis of webpages possible • Suitable for malware and embedded code detection • Attacker behavior detection possible • Able to detect attacks on the fly • Widely considered in the prior work	• Require more time for webpage rendering and processing • Code obfuscation and injection becomes challenging • System design becomes cumbersome
Domain Name Features [4], [5], [6], [16],[8], [7]	• Easy feature extraction • Useful in different attack detection • Widely considered in the prior work for attack types identification	• Prone to typosquatting or URL hijacking • Domain name resolution may becomes cumbersome • Domain registration period play an vital role because malicious domains have very short life-span
Short URLs Features [13]	• High comprehensiveness	• Time consuming extraction process • Most of the URLs were offline

The examples of hidden iframes is given in Table 8, 9. The first $iframe$ is invisible because its HTML attribute visibility is set as false. The second $iframe$ has zero width and height.

3.2.3 Domain Name Features

We have used 18 domain name features, among these 7 are taken from the literature [4–8, 16, 18] and 11 are new features. We have extracted the domain names from the URL string using a script written in Java. These are numeric, binary and real value features. The domain name features are given in Table 11.

- *Entropy of Domain Name*

We have used Shannon Entropy to demonstrate the randomness factor in domain names of malicious and benign URLs. High entropy indicates the more suspicious nature of the URL. The Shannon entropy of the domain name string is calculated using Equation (1). Table 12 show the average entropy of malicious and benign domain names and longest domain tokens used in our dataset.

It is clear that, the entropy of domain names and longest tokens in domain names of malicious URLs is higher than benign URLs. It indicates that there is more randomness factor in malicious URLs, to mark it as malicious.

3.2.4 Short URLs Features

Today Online Social Networks (OSN) like Twitter, Facebook, WhatsApp etc., are widely used by millions of users all over the world for communication. Due to the text limitation on OSN, URL shortening services like *bit.ly, goo.gl, tinyurl.com, owl.ly, deck.ly, su.pr, bit.do* etc. are widely used; however they are not free from risks [13]. It is also applicable to the Webpages. To deceive the legitimate users attackers often use such types of URL shortening services to hide their original identity. Considering this in mind, we have extracted two features of short URLs i.e. length of expanded URL and is URL is malicious?. We have written an URL expander script in Java, once we get the short URL with above URL shortening services; our expander script returns the original

Table 4. URL Features

Feature Name	Type
Features used in the literature	
Length of URL	numeric
Presence of IP address in Hostname	numeric
Length of Query string in URL	numeric
Number of Tokens in URL	numeric
Number of Dots (.) characters	numeric
Number of Hyphens (-) sign characters	numeric
Number of Underscore (_) sign characters	numeric
Number of Equal (=) sign characters	numeric
Number of Forward slash (/) sign characters	numeric
Number of Question Mark sign (?)characters	numeric
Presence of “secure” word in URL string	binary
Presence of “account” word in URL string	binary
Presence of “webscr” word in URL string	binary
Presence of “login” word in URL string	binary
Presence of “ebayisapi” word in URL string	binary
Presence of “signin” word in URL string	binary
Presence of “banking” word in URL string	binary
Presence of “confirm” word in URL string	binary
Presence of “blog” word in URL string	binary
Presence of “logon” word in URL string	binary
Presence of “signon” word in URL string	binary
Presence of “login.asp” word in URL string	binary
Presence of “login.php” word in URL string	binary
Presence of “login.htm” word in URL string	binary
Presence of “.exe” word in URL string	binary
Presence of “.zip” word in URL string	binary
Presence of “.rar” word in URL string	binary
Presence of “.jpg” word in URL string	binary
Presence of “.gif” word in URL string	binary
Presence of “viewer.php” word in URL string	binary
Presence of “link=” word in URL string	binary
Presence of “getImage.asp” word in URL string	binary
Presence of “plugins” word in URL string	binary
Presence of “paypal” word in URL string	binary
Presence of “order” word in URL string	binary
Presence of “dbsys.php” word in URL string	binary
Presence of “config.bin” word in URL string	binary
Presence of “download.php” word in URL string	binary
Presence of “.js” word in URL string	binary
Presence of “payment” word in URL string	binary
Presence of “files” word in URL string	binary
Presence of “css” word in URL string	binary

Table 5. URL Features

Presence of “shopping” word in URL string	binary
Presence of “mail.php” word in URL string	binary
Presence of “.jar” word in URL string	binary
Presence of “.swf” word in URL string	binary
Presence of “.cgi” word in URL string	binary
Proposed Features	
Number of Semicolon (;) sign characters	numeric
Number of Open Parenthesis (() sign characters	numeric
Number of Close Parenthesis()) sign characters	numeric
Number of Mod Sign (%) sign characters	numeric
Number of Ampersand Sign (&) sign characters	numeric
Number of At the Rate Sign (@) sign characters	numeric
Number of Digits in the URL	numeric
Entropy of URL string	real
Presence of “.php” word in URL string	binary
Presence of “abuse” word in URL string	binary
Presence of “admin” word in URL string	binary
Presence of “.bin” word in URL string	binary
URL without “www”	binary
Presence of “personal” word in URL string	binary
Presence of “update” word in URL string	binary
Presence of “verification” word in URL string	binary

Table 6. Average Entropy of benign and malicious URLs used in our dataset

Benign URLs	Malicious URLs
3.87	4.14

Table 7. Frequency Distribution of Word Based Features in URLs

Feature Name	Benign (%)	Malicious (%)
“.php” word in URL	0.03	35.66
“abuse” word in URL	0.01	5.51
“admin” word in URL	0.04	6.45
“.bin” word in URL	0.08	0.13
“personal” word in URL	0.03	0.19
“update” word in URL	0.15	2.2
“verification” word in URL	0.00	0.72

Table 8. Examples of hidden iFrame malwares [33]

< iframe width=“0” height=“0” frameborder=“0” src=“http://facebook.cn” visibility=“true” >

Table 9. Examples of hidden iFrame malwares [33]

< iframe width= “700” height=“500” frameborder=“0” src=“http://facebook.cn” visibility =“false”>

Table 10. URL Source Features

Feature Name	Type
Features used in the literature	
Number of Blank Lines in a Web Page	numeric
Number of Blank Spaces in a Web Page	numeric
Number of Words in a Web Page	numeric
Average Length of Words in a Web Page	real
Number of iFRames in a Web Page	numeric
Number of JavaScript in a Web Page	numeric
Number of embed Tag in a Web Page	numeric
Number of object Tag in a Web Page	numeric
Number of meta Tag in a Web Page	numeric
Number of div Tag in a Web Page	numeric
Number of body Tag in a Web Page	numeric
Number of form Tag in a Web Page	numeric
Title Tag present? in a Web Page	binary
Number of anchor Tag in a Web Page	numeric
Number of Hidden elements in a Web Page	numeric
Number of External JS Files in a Web Page	numeric
Number of Links in a Web Page	numeric
Number of Internal Links in a Web Page	numeric
Number of External Links in a Web Page	numeric
Number of applet Tag in a Web Page	numeric
Number of input Tag in a Web Page	numeric
Proposed URL Source Features	
Number of image Tag in a Web Page	numeric
Number of span Tag in a Web Page	numeric
Number of CSS styles in a Web Page	numeric
Number of audio Tag in a Web Page	numeric
The size of Webpage	numeric
Credit card number word present?	binary
“log” word present?, in a Web Page	binary
“pay” word present?, in a Web Page	binary
“free” word present?, in a Web Page	binary
“access” word present?, in a Web Page	binary
“bonus” word present?, in a Web Page	binary
“click” word present?, in a Web Page	binary
Entropy of Webpage	real

URL. We have set the threshold of >=30 characters

Table 11. Domain Name Features

Feature Name	Type
Features used in the literature	
Length of Domain Name	numeric
Domain Name contains IP address?	binary
Is Domain is TLD?	binary
Number of Sub-Domains	numeric
Number of Yahoo links for domain	numeric
Number of Bing links for domain	numeric
Alexa Rank of domain	numeric
Proposed Features	
Domain Name is Valid?	binary
Entropy of Domain Name string	real
Number of tokens in Domain Name	numeric
Length of Longest Domain Token	numeric
Entropy of Longest Domain token	real
Average length of domain token	real
Number of tokens in Path	numeric
Length of Longest Path Token	numeric
Average length of path token	real
Domain Name contains suspicious https?	binary
Domain Name contains suspicious www?	binary

Table 12. Average Entropy of benign and malicious URL domain names and longest domain tokens

Benign URL Domain Name	Malicious URL Domain Name	Longest Domain token in Benign URL	Longest Domain token in Malicious URL
3.25	3.37	2.52	2.89

for the length of the URL. Also, to decide the URL is malicious or benign we have retrieved the lexical features i.e. is URL contains suspicious characters like,, $=, (,), \%, \&, @$. These are numeric and binary features and given in Table 13.

Motivation behind setting threshold of 30 characters for length of URLs: As per our dataset of benign and malicious URLs, we checked the length of both types of URLs. It is found that the average length of benign URLs is 23.49 characters and that of malicious URLs is 48.54 characters. Benign URLs are validated meaningful URLs created for the ease of user understanding, but in case of malicious URLs the attackers intention is to deceive the legitimate users, hence they use various encoding and obfuscation techniques to code the URL. It will result increase in the length of the URLs. But it is not the thumb rule applicable for every benign URL, hence we combined the lexical properties of the URLs to mark it as malicious,

Table 13. Short URLs Features

Feature Name	Type
Length of expanded URL	numeric
Is URL is malicious?	binary

means if the length of the expanded URL is greater than or equal to threshold *i.e* 30 characters and contains above mentioned suspicious symbols, then it is classified as malicious.

3.2.5 Feature Representation

We used the classification algorithms like OVA L1-reg L2-loss SVM (OVA SVM), OVO L1-reg L2-loss SVM (OVO SVM) and multi-class confidence weighted learning(MC-CW), which requires us to store the URL features as sparse vectors. Much of data in machine learning is sparse, that is mostly zeros and often binary. We implemented a feature pre-processing module in Java, which remove all the zero value features from the dataset. Due to this technique, the reduced feature subset noticeably improved the training time. We combined the four feeds of URLs like benign, malicious, phishing and spam, with an average 4:3:2:1 ratio of benign-to-malicious-to-phishing-to-spam URLs. Figure 2 shows our feature vector for multi-class URL dataset.

```
1 1:21 4:8 5:2 9:2 48:10 52:70 53:108 54:1 56:1170 57:1141 58:51 59:1 60:14 63:1 64:150 65:1
66:1 67:1 68:49 71:49 73:49 81:3.56 91:2.65 92:3 93:6 94:1.92 95:3.33 101:1 102:47 103:11
104:12 109:1 114:1 115:5.6
1 1:23 4:8 5:2 9:2 48:12 52:82 53:118 54:3 56:65 57:65 58:83 60:5 63:8 67:1 70:1 81:3.76 91:3.02
92:3 93:8 94:2.75 95:4 101:4 102:11 103:32 104:1 109:1 115:5.25
1 1:22 4:8 5:2 9:2 48:11 52:76 53:103 54:2 55:124 56:2974 57:2583 58:9 59:1 60:12 63:3 64:53
65:1 66:4 67:1 68:23 70:2 71:23 73:23 81:3.66 91:3.1 92:3 93:7 94:2.52 95:3.66 101:3 102:60
103:7 104:2 109:1 114:1 115:5.17
1 1:20 4:8 5:2 9:2 48:9 52:80 53:104 54:6 57:1 59:5 60:22 63:15 64:213 65:1 66:1 67:1 68:105
70:9 71:105 73:105 81:3.45 91:2.64 92:3 93:5 94:1.92 95:3 101:23 102:80 103:7 104:7 107:8
2 1:21 4:8 5:2 9:2 48:10 52:57 53:54 55:10 56:83 57:59 58:25 59:1 60:12 63:7 64:162 65:1 67:1
68:140 70:7 71:140 73:140 81:3.78 91:3.12 92:3 93:6 94:2.25 95:3.33 101:25 102:448 107:95429
115:5.71
2 1:17 4:6 5:1 9:2 48:10 52:57 53:55 55:10 56:83 57:59 58:25 59:1 60:12 63:7 64:154 65:1 67:1
68:140 70:7 71:140 73:140 81:3.73 86:1 91:3.12 92:3 93:6 94:2.25 95:3.33 101:25 102:448
107:95429 115:5.71
2 1:20 4:6 5:1 9:2 48:13 52:59 53:59 55:86 56:5152 57:4587 58:4 60:3 64:5 65:1 67:1 81:3.82
86:1 91:3.39 92:3 93:9 94:2.95 95:4.33 109:1 110:1 111:1 112:1 114:1 115:4.83
3 1:39 4:14 5:3 6:1 9:3 48:22 51:1 52:71 53:58 54:9694989 55:7 56:521 57:424 58:4 63:1 64:1
65:1 67:1 68:1 71:1 73:1 81:4.41 82:1 86:1 91:4.1 92:5 93:9 94:2.95 95:4.4 96:4 97:5 98:2.5 104:1
109:1 110:1 111:1 114:1 115:5.18
3 1:87 4:29 5:2 6:4 9:9 14:1 48:26 51:1 52:71 53:51 54:150079 55:2 56:168 57:148 58:5 63:3 64:2
65:1 67:1 68:3 71:3 73:3 80:13 81:4.66 86:1 91:3.67 92:5 93:12 94:3.25 95:5.19 96:13 97:16
98:4.07 101:1 107:866 109:1 115:4.85
4 1:23 4:8 5:2 9:2 48:12 52:55 53:52 55:86 56:5152 57:4587 58:4 60:3 64:5 65:1 67:1 80:4 81:3.5
91:2.75 92:3 93:8 94:2 95:4 109:1 110:1 111:1 112:1 114:1 115:4.83
```

Figure 2. Feature vector for multi-class URL dataset

3.3 Characteristics of New Features

In this work, we used 42 new URL features for malicious URLs detection and attack types identification. As given in Section 2 and Section 3.2, the researchers have used several URL features for the detection task, still due to the ever-changing nature of attackers, there are many open issues. To cope up with issues, we proposed following new URL features given with there characteristics.

(a) *New URL Features*

Here, we proposed 16 new URL features including, 1) Presence of lexical symbols like ; , (,), %, &, @ in the URL string. Most of the phishing, malware and spam URLs contains such suspicious lexical symbols to hide the identity and to make URL more cryptic to understand. The main intention of the attacker in this case is to deceive the legitimate user, as given below.

http://internationalogo.com/path/lp.php?trvid=11645&trvx=1ff4e676&hash=1482849787mb\33247007894&PREFIJO=2108&PUBID=0&USERAGENT=Mozilla/5.0%20(Linux;%20U;%20Android%204.0.3;%20de-ch;%20HTC%20Sensation%20Build/IML74K)%20AppleWebKit/534.30%20(KHTML,%20like%20Gecko)%20Version/4.0%20Mobile%20Safari/534.30

2) Entropy of URL string: as given in Table 6 the average entropy of malicious URLs is greater than benign URLs. Entropy measures the randomness factor in URLs, it shows that there is more randomness factor in malicious URLs as compare to benign URLs.

3) Presence of suspicious words in URLs: we manually verified that phishing, malware and spam URLs contains suspicious words like, ".php", "abuse", "admin", ".bin", "personal", "update" and "verification". Here, the main intention of the attacker is to redirect the legitimate user. Table 7 shown the frequency distribution of these suspicious words in benign and malicious URLs. It is clear from Table 7, that the occurence of these suspicious words in malicious URLs is higher than benign URLs.

4) URLs without "www": we manually verified that most of the phishing and malware URLs does not contains "www", as given below,

http://appleid.apple.co.uk

5) No. of Digits in the URL: Our study shows that most of the phishing and malware URLs contains digits as given below,

http://dl.pocolegion.com/n/5476e90e-8990-4e80-b523-19020a000013/FLV˙Media˙Player.exe

(b) *New URL Source Features*

Here, we proposed 13 new URL source features. These features are collected while rendering the webpage, which demonstrates the run-time behavior of the webpage. These features includes,

1) No. of suspicious HTML tags: It includes HTML tags like image, span, CSS styles and audio. These tags are suspicious in the sense that, they allow attacker to embed malicious contents in the webpage.

2) Size of Webpage: we have used webpage size as one of the attribute to classify webpage as malicious or

benign. Most of the malicious and phishing webpages contains some of the suspicious payloads like images, applets and audios, which increases the size of the webpage. Hence, this parameter plays an important role in the malicious URLs detection.

3) Presence of security sensitive words: According to our study, most of the phishing, malicious and spam webpages contains some of the security sensitive words which we have considered, like "credit card number", "log", "pay", "free", "access", "bonus" and "click". These words force legitimate users to provide security sensitive information and redirect them to the attackers location.

4) Entropy of Webpage: As we have seen, entropy measures the randomness factor in the URLs. The same concept we applied for the malicious and benign webpages classification. As per our study, the malicious webpages exhibits more randomness factor than benign webpages.

(c) *New Domain Name Features*

Here, we proposed 11 new domain name features. To deceive the legitimate users, domains have become a key element for the cybercriminals. Phishers may set up webpages that masquerade as trustworthy brands, such as banks and e-commerce sites. Cybercriminals lure victims to fake sites and users are tricked into providing sensitive information such as usernames, passwords and credit card details. To catch the attackers suspicious actions, we proposed these new domain name features including, domain name is Valid?, entropy of domain name string, no. of tokens in domain name, length of longest domain token, entropy of longest domain token, average length of domain token, no. of tokens in path, domain name contains suspicious https? and domain name contains suspicious www?.

(d) *Short URLs Features*

In the recent days short URLs became more popular on social media and other sites. It is a very effective method for hackers and attackers to reach a wide audience, which dramatically increases the percentage of infections. Cybercriminals use short links, created by shortening services, to plant malware and phishing links on social media and other sites. Considering this in mind, we proposed two short URLs features which includes, length of expanded URL and is URL is malicious?. We expand the short URL to obtain the original URL and by extracting its lexical properties we declare it as malicious or benign.

3.4 Multi-class Classification

In machine learning, multi-class classification is the problem of classifying instances into one of the more than two classes. Supervised multi-class classification algorithms aim at assigning a class label for each input example. Given a training data set of the form (x_i, y_i), where $x_i \in R^n$ is the i^{th} example and $y_i \in (1, ..., K)$ is the i^{th} class label, we aim at finding a learning model H such that $H(x_i) = y_i$ for new unseen examples. The problem is simply formulated in the two class case, where the labels y_i are just +1 or -1 for the two classes involved. The existing multi-class classification techniques can be categorized into [34–36],

- Solving several binary problems,
- Considering all data at once and
- Maximum entropy.

We have used the solving several binary problems strategy to design our experiments and the detail is given as following.

- *Solving Several Binary Problems*

According to [34, 36], the multi-class classification problem can be decomposed into several binary classification tasks that can be solved efficiently using binary classifiers. The most successful and widely used binary classifier is the support vector machines. Most of multi-class classification methods are originally proposed to solve a two-class problem. Multi-class classification can be decomposed to several binary classification problems. One-vs-all and one-vs-one methods are two of the most common decomposition approaches.

(1) *One-vs-all (OVA) Support Vector Machine (SVM) method*

According to this method, if there are k classes in the training data, the one-vs-all method [37] constructs k binary classification models. To obtain the m^{th} model, instances from the m^{th} class of the training set are treated as positive and all other instances are negative. Then the weight vector w^m for the m^{th} model can be generated by SVM linear classifier.

Given training data $(y_i, x_i) \in \{-1, +1\} \times R^n$, $i = 1, ..., l$, where y_i is the label and x_i is the feature vector, SVM construct the following decision function [36],

$$d(x) \equiv w^T \phi(x) + b \tag{2}$$

where,
$d(x)$ is the decision function

w is the weight vector,
$\phi(x)$ is the higher dimensional vector
b is called the bias

After obtaining all k models, an instance x is in the m^{th} class if the decision value of the decision function 2 of the m^{th} model is the largest, i.e

$$\text{class of x} \equiv \arg \max_{m=(1,\ldots,k)} w^T m_x \tag{3}$$

(2) *One-vs-one (OVO) Support Vector Machine (SVM) method*

The one-vs-one method solves binary problems. Each binary classifier constructs a model with data from one class as positive and another class as negative. Since there are combination of two classes, weight vectors are constructed: $w_{1,2}$,$w_{1,3}$,. . . , $w_{1,k}$,$w_{2,3}$, . . . , $w_{(k-1),k}$. There are different methods for testing. One method is by voting. To test an instance x, if model (i, j) predicts x as in the i^{th} class, then a counter for the i^{th} class is added by one; otherwise, the counter for the j^{th} class is added. Then x is in the i^{th} class if the i^{th} counter has the largest value. Other prediction methods are similar though they differ in how to use the decision values.

We have used the LibLinear implementation of the above methods for multi-class classification of our multi-class URL dataset [38]. The usage of this tool is given as follows,

- s type: set type of solver for multi-class classification
  - 5 = L1-regularized L2-loss support vector classification
- c = cost: set the parameter C
- B = bias
- w_i = weight: weights adjust the parameter C of different classes
- M = type: type of multi-class classification
  - M = 0: one-versus-all
  - M = 1: one-versus-one

• *Multi-class Online Confidence Weighted Learning*

Online confidence weighted (CW) learning for binary classification is introduced by [39, 40]. According them, in binary case, a distribution is maintained over weight vectors $w \sim \mathcal{N}(\mu, \Sigma)$. Given an input instance x, a classifier draws a weight vector $w \sim \mathcal{N}(\mu, \Sigma)$ and then predicts the label with the maximal score, $arg\ max_z(w.f(x, z))$. As in the binary case, for multi-class learning the prediction rule is defined as a robustness condition and corresponding learning updates. Following the update on round i, the i^{th} instance is correctly labeled with probability at least $\eta < 1$. Among the distributions that satisfy this condition, the one that has been chosen with the minimal Kullback-Leibler (KL) distance from the current distribution [40, 41]. The CW update rule is,

$$(\mu_{i+1}, \Sigma_{i+1}) = arg \min_{\mu,\Sigma} D_{KL}\left(N(\mu, \Sigma)||N(\mu_i, \Sigma_i)\right)$$
$$s.t. Pr[y_i|x_i, \mu, \Sigma] \geq \eta, \tag{4}$$

where,

$$Pr_{[y|\mathbf{x}, \text{-}, \Sigma]} = Pr_{w\sim N(\mu,\Sigma)}[y = arg \max_{z\in y} (w \cdot f(x, z))]$$

w is a weight vectors,
x is an input instance,
$\mathcal{N}(\mu, \Sigma)$ is a the multivariate normal distribution,
μ is a the vector of feature means,
Σ is the diagonal covariance matrix (i.e., the confidence) of the features and
η is the class probability
$f(x, z)$ is a feature function.

According to [39, 41] in the binary case, the confidence weighted (CW) update rule has a closed-form solution. In the multi-class case, there exists no closed-form solution but the solution can be efficiently approximated.

We have used the multi-class confidence weighted learning implementation by [41, 42], in our experiments to classify our multi-class URLs dataset.

The primary intention to use multi-class classification algorithms like One-vs-all (OVA) SVM, One-vs-one (OVO) SVM and Multi-class Online Confidence Weighted Learning to evaluate the performance of these algorithms on our multi-class URL dataset to detect the class or type of URL. Table 14 gives the characteristics of these multi-class classification algorithms,

4 Experimental Results and Discussion

4.1 Data Source and Dataset

We have collected benign and malicious URLs from the benchmark sources and divided the dataset into a ratio of 66:34 as training and a testing set i.e. 66% for training and 34% for testing. All the mentioned URLs are collected between September 1, 2017 to September 30, 2017. The dataset of benign URLs is collected from the Alexa Top sites [43]. We collected 26041 benign URLs from the above source of benign URLs. For the malicious dataset, we have collected URLs from three benchmark sources, like the malware and phishing blacklist of the PhishTank database of verified phishing pages [44], the malware and injection at-

Table 14. Characteristics of OVA SVM, OVO SVM and Multi-class Online CW Learning

Multi-class Classifier	Characteristics
One-vs-all (OVA) SVM [36]	• Easy implementation • Based on decomposing multi-class task into several binary classification tasks which can be solved efficiently using binary classifier like SVM. • We used SVM binary classifier, which is the most successful and widely used. • The cost for testing an instance is $\mathcal{O}(nk)$ • Better suited for our problem, because our dataset contains only 4 classes.
One-vs-One (OVO) SVM [36]	• Easy implementation • Based on decomposing multi-class task into several binary classification tasks which can be solved efficiently using binary classifier like SVM. • For linear classifiers, one-vs-one method gives better testing accuracy. • It requires $\mathcal{O}(k^2n)$ for storing models and $\mathcal{O}(k^2n)$ cost for testing an instance; both are more expensive than the one-vs-all method, but well suited for our problem, because our dataset contains only 4 classes.
Multi-class Online CW Learning [40]	• Widely used and performed well on many binary NLP classification tasks. • Highly scalable. • Online learning algorithms process one example at a time, yielding simple and fast updates. • The CW algorithm is well-suited for the detection of malicious URLs and attack type identification, because it estimates confidences on the weights of individual features and aggressively update the weights on features.

Table 15. Dataset for Training and Testing

Class of URLs	Training	Testing	Total
Benign	13185	12856	26041
Malware	9888	1409	11297
Phishing	6590	2386	8976
Spam	3295	326	3621
Total	32958	16977	49935

tack URL list of Malware Domain List [45] and Spam domain blacklist by jwSpamSpy [46]. We collected 26041 malicious URLs from the above benchmark sources of malicious URLs including 8976 phishing URLs, 11297 malware URLs and 3621 spam URLs. The breakdown of the dataset is shown in Table 15.

4.2 Experimental Results

4.2.1 Binary Evaluation Measures

The correctness of a binary classification task can be evaluated by computing the confusion matrix shown in Table 16, we have calculated following measures, to evaluate the performance of the classifiers. A binary classifier predicts all data instances of a test dataset as either positive or negative. This classification (or prediction) produces four outcomes true positive (tp), true negative (tn), false positive (fp) and false negative (fn) [47].

Table 16. Binary confusion matrix

		Predicted	
		Positive	Negative
Actual	Positive	tp	fn
	Negative	fp	tn

- tp : It is the correct positive (malicious URL) prediction.
- fp : It is the incorrect positive (malicious URL) prediction.
- tn : It is the correct negative (benign URL) prediction.
- fn : It is the incorrect negative (benign URL) prediction.

The binary performance evaluation measures like $Accuracy, FPR, FNR, Precision, Recall, F-measure, AUC$ are enlisted in Table 17.

4.2.2 Multi-class Evaluation Measures

We have evaluated the performance of multi-class batch and online machine learning classifiers on our

Table 17. Binary Evaluation Metrics

Metrics	Definition	Equation
Accuracy	All correct predictions divided by the total number of instances	$Accuracy = \frac{tp+tn}{tp+tn+fn+fp}$
FPR	The number of incorrect negative (benign URL) predictions divided by the total number of negatives (benign URL)	$FPR = \frac{fp}{tn+fp}$
FNR	The number of incorrect positive (malicious URL) predictions divided by the total number of positives (malicious URL)	$FNR = \frac{fn}{tp+fn}$
Precision	The number of correct positive predictions divided by the total number of positive predictions	$Precision = \frac{tp}{tp+fp}$
Recall	The number of correct positive predictions divided by the total number of positives	$Recall = \frac{tp}{tp+fn}$
F-measure	A measure that combines precision and recall is the harmonic mean of precision and recall	$Fmeasure = \frac{2.Precision.Recall}{Precision+Recall}$
AUC	It captures a single point on the reception operating characteristic curve	$AUC = \frac{1}{2}\left(\frac{tp}{tp+fn} + \frac{tn}{tn+fp}\right)$

Table 18. Multi-class Confusion Matrix

	Classified as Class C_1	$\cdots$	Classified as Class C_i	$\cdots$	Classified as Class C_l
Actual Class C_1	C_{11}	$\cdots$	C_{1i}	$\cdots$	C_{1l}
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
Actual Class C_i	C_{i1}	$\cdots$	C_{ii}	$\cdots$	C_{il}
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
Actual Class C_l	C_{l1}	$\cdots$	C_{li}	$\cdots$	C_{ll}

URL dataset shown in Table 15. The measures for multi-class classification is given as below. For an individual class C_i, the assessment is defined by tp_i, fn_i, tn_i, fp_i, $Accuracy_i$, $Precision_i$ and $Recall_i$ are calculated from the counts for C_i. Quality of the overall classification is usually assessed in two ways,

- A measure is the average of the same measures calculated for C_1, . . . ,C_l (macro-averaging shown with an M index).
- The sum of counts to obtain cumulative tp_i, fn_i, tn_i, fp_i and then calculating a performance measure (micro-averaging shown with l indices).

Macro-averaging treats all classes equally while micro-averaging favors bigger classes. Measures for multi-class classification for many classes C_i: tp_i are true positive for C_i, fp_i are the false positive, fn_i are the false negative and tn_i are the true negative counts respectively. μ and M indices represent micro and macro-averaging. We have used the multi-class confusion matrix given in Table 18 [47, 48].

The positives and negatives are defined through elements of the confusion matrix as follows, where,

- tp_i(true positive) = the number of correctly recognized observations for class C_i.
- tn_i(true negative) = the number of correctly recognized observations that do not belong to the class C_i.
- fp_i(false positive) = the number of observations that were incorrectly assigned to the class C_i.
- fn_i(false negative) = the number of observations that were not recognized as belong to the class C_i.

$$tpi = Cii \tag{5}$$

$$fpi = \sum_{i=1}^{l} cni - tpi \tag{6}$$

$$fni = \sum_{i=1}^{l} cin - tpi \tag{7}$$

$$tni = \sum_{i=1}^{l}\sum_{j=1}^{l} cnj - tpi - fpi - fni \tag{8}$$

Using the above multi-class confusion matrix the measures like error rate, micro-precision, micro-recall, micro-f-score, macro-precision, macro-recall and macro-f-score are calculated as given in Table 19.

4.2.3 Significance of Proposed Features

To verify whether the features we have introduced are important in enhancing the effectiveness of analysis and detection of malicious URLs as well as attack type identification, we compared the binary and multi-class classification measures with and without our newly introduced features on our binary and multi-class URL dataset. In binary settings, the performance of the classifier is evaluated with the help of *accuracy, error rate, fpr, fnr, precision, recall, fscore*

Table 19. Multi-class Evaluation Metrics

Metrics	Definition	Equation
Average_Accuracy	It is the average per-class effectiveness of a classifier	$Average\_Accuracy = \frac{\sum_{i=1}^{l} \frac{tpi+tni}{tpi+fni+fpi+tni}}{l}$
Error_Rate	It is the average per-class classification error	$Error\_Rate = \frac{\sum_{i=1}^{l} \frac{fpi+fni}{tpi+fni+fpi+tni}}{l}$
Micro-Precision	It is the agreement of the data class labels with those of a classifiers if calculated from sums of per-text decisions	$\Pr ecision\mu = \frac{\sum_{i=1}^{l} tpi}{\sum_{i=1}^{l} (tpi + fpi)}$
Micro-Recall	It is the effectiveness of a classifier to identify class labels if calculated from sums of per-text decisions	$Recall\mu = \frac{\sum_{i=1}^{l} tpi}{\sum_{i=1}^{l} (tpi + fni)}$
Micro-Fscore	It is the relations between data positive labels and those given by a classifier based on sums of per-text decisions	$Fscore\mu = \frac{(\beta^2+1)\text{Precision}\mu.\text{Recall}\mu}{\beta^2\,\text{Precision}\mu + \text{Recall}\mu}$
Macro-Precision	It is an average per-class agreement of the data class labels with those of a classifiers	$\Pr ecisionM = \frac{\sum_{i=1}^{l} \frac{tpi}{tpi+fpi}}{l}$
Macro-Recall	It is an average per-class effectiveness of a classifier to identify class labels	$RecallM = \frac{\sum_{i=1}^{l} \frac{tpi}{tpi+fni}}{l}$
Macro-Fscore	It is the relations between data positive labels and those given by a classifier based on a per-class average labels	$FscoreM = \frac{(\beta^2+1)\,\Pr ecisionM.RecallM}{\beta^2\,\Pr ecisionM + RecallM}$

and AUC. In multi-class settings, the performance of the classifier is evaluated with the help of *accuracy, average accuracy, error rate, micro-precision, micro-recall, micro-f-score, macro-precision, macro-recall and macro-f-score.*

- *Performance evaluation on binary URL dataset*

Table 20 shows the overall contribution of new features on the performance of classifiers using binary settings. Here, binary confidence weighted learning classifier achieved 99.86% of accuracy on test set, with increase of 0.05% as compare to without new features. Also, binary L1-reg L2-loss SVM achieved 97.94% of accuracy on test set, with increase of 3.45% as compare to without new features. The error rate of both the classifiers is improved using new features as shown with (↓). Related to confidence weighted learning classifier, the fpr is slightly increased and *precision*, AUC is slightly decreased. For binary L1-reg L2-loss SVM classifier, fpr, fnr shown with (↓) and *precision*, *recall*, $fscore$, AUC shown with (↑), shows improvement in classifier performance using new features.

- *Performance evaluation on multi-class URL dataset*

As shown in Table 21 and Figure 3, the use of new features improved the overall performance of the classifiers, as shown with (↑) for improved average accuracy using multi-class settings. Also, the overall classification accuracy of multi-class confidence weighted learning classifier on the test set is 98.34% with new features, with increase of 0.80% as compare to without new features. Multi-class confidence weighted learning classifier outperforms OVA L1-reg L2-loss SVM (OVA SVM) and OVO L1-reg L2-loss SVM (OVO SVM) in case of average accuracy on test set.

We compared average malicious URLs detection accuracy using features used in the literature and our proposed features and found statistically significant difference (p= 0.057), by using t-student test.

As shown in Table 22, by the inclusion of new features the error rate of the classifiers is decreased. Also, the micro-precision, micro-recall, micro-f-score, macro-precision, macro-recall and macro-f-score is improved. Only the macro-recall of OVA L1-reg L2-loss SVM and OVO L1-reg L2-loss SVM is slightly decreased with the use of new URL features. As shown in Figure 4, the error-rate is decreased using new fea-

Table 20. Detail performance analysis of the binary classifiers in (%) on our binary URL dataset with and without proposed features

Classifier	Accuracy	Error Rate	FPR	FNR	Precision	Recall	Fscore	AUC
Without Proposed Features								
CW	99.81	0.19	0.12	0.26	99.88	99.74	99.81	99.88
L1-reg L2-loss SVM	94.49	5.51	9.65	1.37	91.09	98.63	94.71	90.35
With Proposed Features								
CW	99.86(↑)	0.14(↓)	0.13(↑)	0.14(↓)	99.87(↓)	99.86(↑)	99.86(↑)	99.87(↓)
L1-reg L2-loss SVM	97.94(↑)	2.06(↓)	3.81(↓)	0.32(↓)	96.32(↑)	99.68(↑)	97.97(↑)	96.19(↑)

Table 21. Overall contribution of proposed features on the average accuracy of classifiers using multi-class setting

Multi-class Classifier	Avg. Accuracy Without Proposed Features(%)	Avg. Accuracy With Proposed Features(%)	Change (%)
MC-CW	97.54	98.44	0.90(↑)
OVA SVM	96.23	97.55	1.32(↑)
OVO SVM	96.64	97.46	0.82(↑)

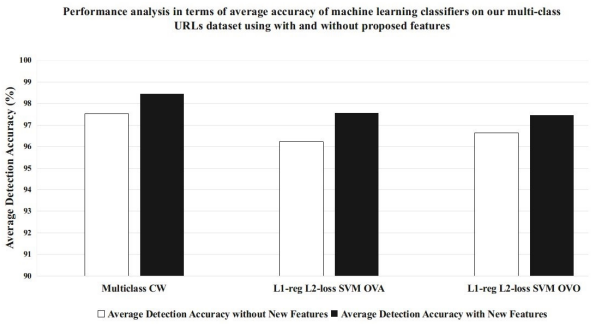

Figure 3. Performance analysis in terms of average accuracy of machine learning classifiers on our multi-class URLs dataset using with and without proposed features

tures for the classifiers. It is the good indication that, new features improves the effectiveness of malicious URLs detection.

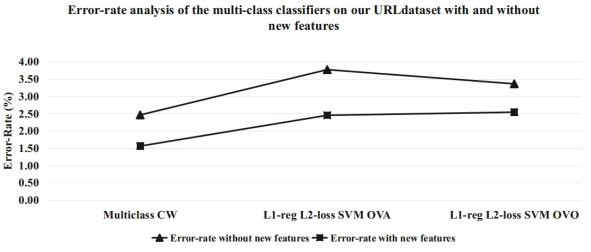

Figure 4. Error-rate analysis of the multi-class classifiers on our URL dataset with and without proposed features

Accordingly, Table 23 shows the detection accuracy of each type of URL class type on our multi-class test set with new features and without new features. It is clear from the table, that there is a significant improvement in the identification of benign, malware and phishing URLs using new feature set except spam URLs for the three classifiers.

Table 24, 25 and 26 shows the class-specific error analysis of the three classifiers. There seems to be a significant overlap between spam and benign URLs with about 13.80% and 15.95% of spam URLs misclassified as benign without new and with new features for multi-class confidence weighted learning, about 37.73% and 58.59% spam URLs misclassified as benign without new and with new features for OVA L1-reg L2-loss SVM and about 41.72% and 58.59% spam URLs misclassified as benign without new and with new features for OVO L1-reg L2-loss SVM. The most probable reason behind high mis-classification of spam as benign is the similarity of features used in our analysis.

Spam URLs misclassified as phishing and malware are close to 0% in most of the cases, indicating that the URL features we have used are effective in differentiating spam from either phishing or malware URLs.

Regarding phishing URLs about 6.83% and 2.77% mis-classified as malware without new and with new features for multi-class confidence weighted learning, about 14.21% and 13.91% mis-classified as malware without new and with new features for OVA L1-reg L2-loss SVM and about 16.89% and 13.91% mis-classified as malware without new and with new features for OVO L1-reg L2-loss SVM. This indicates that there is overlap between malware and phishing URLs as most of the feature values have high correlations.

Finally as shown in Table 27, we have given comparison of our proposed methodology with the similar study of [16]. As can be seen from Table 27, we

Table 22. Detail performance analysis of the multi-class classifiers in (%) on our multi-class URL dataset with and without proposed features

Classifier	Error Rate	Micro-Precision	Micro-Recall	Micro-F-score	Macro-Precision	Macro-Recall	Macro-F-score
Without Proposed Features							
MC-CW	2.46	95.09	95.09	95.09	79.85	92.55	85.73
OVA SVM	3.77	92.47	92.47	92.47	74.15	82.81	78.24
OVO SVM	3.36	93.27	93.27	93.27	75.14	83.10	78.92
With Proposed Features							
MC-CW	1.56(↓)	96.87(↑)	96.87(↑)	96.87(↑)	83.99(↑)	93.95(↑)	88.69(↑)
OVA SVM	2.45(↓)	95.11(↑)	95.11(↑)	95.11(↑)	77.46(↑)	81.95(↓)	79.64(↑)
OVO SVM	2.54(↓)	94.91(↑)	94.91(↑)	94.91(↑)	77.00(↑)	81.60(↓)	79.23(↑)

Table 23. Detection accuracy in (%) for each class type of URL on our multi-class dataset

Classifier	Benign	Malware	Phishing	Spam
Without Proposed Features				
MC-CW	95.72	95.17	92.79	86.50
OVA SVM	94.99	90.19	84.40	61.66
OVO SVM	95.95	93.75	83.19	59.51
With Proposed Features				
MC-CW	97.08(↑)	97.44(↑)	97.23(↑)	84.05(↓)
OVA SVM	96.88(↑)	97.58(↑)	91.32(↑)	42.02(↓)
OVO SVM	96.99(↑)	98.65(↑)	88.73(↑)	42.02(↓)

Table 24. Class-specific error analysis in (%) for Multi-class Confidence Weighted Learning

Class Type	Benign	Malware	Phishing	Spam
Without Proposed Features				
Benign	95.72	0.10	0.10	4.07
Malware	0.00	95.17	4.83	0.00
Phishing	0.29	6.66	92.79	0.25
Spam	13.50	0.00	0.00	86.50
With Proposed Features				
Benign	97.08(↑)	0.05(↓)	0.08(↓)	2.78(↓)
Malware	0.00	97.44(↑)	2.56(↓)	0.00
Phishing	0.04(↓)	2.47(↓)	97.23(↑)	0.25
Spam	15.95(↑)	0.00	0.00	84.05(↓)

achieved improved classification accuracy, attack identification accuracy, micro TP and macro TP. The improved performance measures of our approach is an indication that, the proposed features are more effective in classification and attack type identification. Also the use of online learning classifiers like

Table 25. Class-specific error analysis in (%) for OVA L1-reg L2-loss SVM

Class Type	Benign	Malware	Phishing	Spam
Without Proposed Features				
Benign	94.99	0.49	0.10	4.42
Malware	3.34	90.19	5.90	0.57
Phishing	2.35	12.66	84.41	0.59
Spam	37.73	0.61	0.00	61.65
With Proposed Features				
Benign	96.88(↑)	0.07(↓)	0.19(↑)	2.85(↓)
Malware	0.14(↓)	97.58(↑)	2.27(↓)	0.00(↓)
Phishing	0.29(↓)	8.38(↓)	91.32(↑)	0.00(↓)
Spam	57.98(↑)	0.00(↓)	0.00	42.02(↓)

Table 26. Class-specific error analysis in (%) for OVO L1-reg L2-loss SVM

Class Type	Benign	Malware	Phishing	Spam
Without Proposed Features				
Benign	95.95	0.10	0.04	3.91
Malware	2.20	93.75	3.98	0.07
Phishing	1.01	15.47	83.19	0.34
Spam	40.49	0.00	0.00	59.51
With Proposed Features				
Benign	96.99(↑)	0.11(↑)	0.05(↑)	2.85(↓)
Malware	0.14(↓)	98.65(↑)	1.21(↓)	0.00(↓)
Phishing	0.34(↓)	10.94(↓)	88.73(↑)	0.00(↓)
Spam	57.96(↑)	0.00	0.00	42.02(↓)

confidence weighted learning is more effective than batch learning classifiers.

Table 27. Comparative performance evaluation of our proposed methodology with study by [16]

Measure	Our proposed methodology	Study by [16]
Classification Accuracy (%)	99.86(↑)	98.20
Attack Identification Accuracy (%)	98.44(↑)	93.11
Micro TP (%)	96.87(↑)	91.23
Macro TP (%)	93.95(↑)	89.33

5 Limitations

Considering our approach, it is also not free from limitations. Following are some of the limitations of our multi-class URL classification and attack type identification system,

- There is need to investigate more discriminative spam URL features to differentiate them efficiently from benign URLs.
- Our methodology lacks analysis and detection of obfuscated JavaScripts in the Webpages, which is the major cause behind attacks like drive-by downloads, XSS, malware-delivery etc.
- There is need to investigate more features of short URLs for the effective detection and attack type identification, because it is the most growing trend today on the micro-blogging sites like Twitter, Facebook etc.

6 Conclusions

In this work, we have proposed a methodology to detect malicious URLs and identify attack types. We used 117 various types of discriminative features like URL features, domain name features, URL source features and short URLs features. Significant results are obtained by using proposed 42 novel features. Experimental results on our binary and multi-class dataset show that our methodology is effective for both detection and attack type identification tasks. We have achieved highest accuracy of 99.86% in detection of malicious URLs using binary setting and an average accuracy of 98.44% in identification of attack types using confidence weighted learning classifier in multi-class setting with our proposed URL features. We compared average malicious URLs detection accuracy using features used in the literature and our proposed features and found statistically significant (p= 0.057) increase by using our proposed URL features. Also, we have evaluated the performance of the three learning algorithms in multi-class setting using measures like error rate, micro-precision, micro-recall, micro-f-score, macro-precision, macro-recall, macro-f-score and achieved highly effective results. In future we will concentrate our research activity to improve the Spam URLs identification.

7 Acknowledgments

This work is supported by the financial assistance under the scheme “Rajiv Gandhi Science and Technology Commission (RGSTC), 13-IIST/2014, Government of Maharashtra through North Maharashtra University, Jalgaon, India.

References

[1] Symantec internet security threat report istr 2017. URL https://www.symantec.com/security_response/publications/ monthlythreatreport.jsp. Accessed: September 30, 2017.

[2] Apwg phishing activity trends report analysis 2017. URL http://docs.apwg.org/reports/apwg_trends_report_h1_2017.pdf. Accessed: October 17, 2017.

[3] Patil D. R. and Patil J. B. Survey on malicious web pages detection techniques. *International Journal of U-and E-service, Science and Technology*, 8(5):195–206, 2015. doi: 10.14257/ijunesst.2015.8.5.18.

[4] Justin Ma, Saul L. K., Savage S. and Voelker G. M. Learning to detect malicious urls. *ACM Transactions on Intelligent Systems and Technology*, 3(2):1–24, 2011. doi: 10.1145/1961189.1961202.

[5] Canali D., Cova M., Vigna G. and Kruegel C. Prophiler: a fast filter for the large-scale detection of malicious web pages. In *20th International Conference on World Wide Web (WWW11)*, pages 197–206, 2011.

[6] Eshete B., Villafiorita A. and Weldemariam K. BINSPECT: Holistic Analysis and Detection of Malicious Web Pages. In *SecureComm*, pages 149–166, 2012.

[7] Basnet R. B., Mukkamala S. and Sung, A. H. Detection of Phishing Attacks: A Machine Learning Approach. In *Soft Computing Applications in Industry*, pages 373–383, 2008.

[8] Garera S., Provos N., Chew M. and Rubin A. D. A framework for detection and measurement of phishing attacks. In *2007 ACM workshop on Recurring malcode*, pages 1–8, 2007.

[9] Zhang Y., Hong J. I. and Cranor L. F. Cantina: a content-based approach to detecting phishing web sites. In *16th International Conference on World Wide Web*, pages 639–648, 2007.

[10] Verma R. and Das A. Whats in a URL: Fast Feature Extraction and Malicious URL Detection. In *3rd International Workshop on Security and Privacy Analytics*, pages 55–63, 2017.

[11] Basnet R. B. and Sung A. H. Classifying phishing emails using confidence-weighted linear classifiers. In *International Conference on Informa-*

tion Security and Artificial Intelligence (ISAI), pages 108–112, 2010.

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

[13] Nepali R. K. and Wang, Y. You look suspicious!!: Leveraging visible attributes to classify malicious short urls on twitter. In *49th Hawaii International Conference on System Sciences (HICSS)*, pages 2648–2655, 2016.

[14] Patil D. R. and Patil J. B. Malicious web pages detection using static analysis of URLs. *International Journal of Information Security and Cybercrime*, 5(2):57–70, 2016. doi: 10.19107/IJISC.2016.02.06.

[15] Patil D. R. and Patil J. B. Detection of Malicious JavaScript Code in Web Pages. *Indian Journal of Science and Technology*, 10(19):1–12, 2017. doi: 10.17485/ijst/2017/v10i19/114828.

[16] Choi H., Zhu B. B. and Lee, H. Detecting Malicious Web Links and Identifying Their Attack Types. In *2nd USENIX conference on Web application development(WebApps'11)*, pages 1–12, 2011.

[17] Babagoli, M., Aghababa, M. P. and Solouk, V. Heuristic nonlinear regression strategy for detecting phishing websites. *Soft Computing*, pages 1–13, 2018. doi: https://doi.org/10.1007/s00500-018-3084-2.

[18] Zuhair, H., Selamat, A. and Salleh, M. Selection of robust feature subsets for phish webpage prediction using maximum relevance and minimum redundancy criterion. *Journal of Theoretical and Applied Information Technology*, 81(2):188–205, 2015.

[19] Sahoo D., Liu C.and Hoi, S. C. Malicious URL Detection using Machine Learning: A Survey. *arXiv preprint arXiv:1701.07179.*, pages 1–21, 2017.

[20] Thomas K., Grier C., Ma J., Paxson V. and Song D. Design and evaluation of a realtime URL spam filtering service. In *IEEE Symposium on Security and Privacy (SP)*, pages 447–462, 2011.

[21] Cova M., Kruegel C. and Vigna G. Detection and analysis of drive-by-download attacks and malicious JavaScript code. In *19th International Conference on World Wide Web*, pages 281–290, 2010.

[22] Hajian Nezhad J, Vafaei Jahan M, Tayarani-N M and Sadrnezhad Z. Analyzing new features of infected web content in detection of malicious web pages. *The ISC International Journal of Information Security*, 9(2):63–83, 2017.

[23] Dewald A., Holz T. and Freiling F. C. ADSandbox: Sandboxing JavaScript to fight malicious websites. In *ACM Symposium on Applied Computing*, pages 1859–1864, 2010.

[24] Zhang, J., Seifert, C., Stokes, J. W. and Lee, W. Arrow: Generating signatures to detect drive-by downloads. In *20th international conference on World wide web*, pages 187–196, 2011.

[25] Lee, S.and Kim, J. WarningBird: Detecting Suspicious URLs in Twitter Stream. In *Network and Distributed System Security Symposium (NDSS12)*, pages 1–13, 2012.

[26] Imani M and Montazer GA. Phishing website detection using weighted feature line embedding. *The ISC International Journal of Information Security*, 9(2):49–61, 2017.

[27] Sonowal G. and Kuppusamy K. S. PhiDMA - A phishing detection model with multi-filter approach. *Journal of King Saud University-Computer and Information Sciences*, pages 1–14, 2017. doi: 10.1016/j.jksuci.2017.07.005.

[28] Vinayakumar, R., Soman, K. P. and Poornachandran, P. . Evaluating deep learning approaches to characterize and classify malicious URLs. *Journal of Intelligent & Fuzzy Systems*, 34(3):1333–1343, 2018. doi: 10.3233/JIFS-169429.

[29] Smadi, S., Aslam, N. and Zhang, L. Detection of Online Phishing Email using Dynamic Evolving Neural Network Based on Reinforcement Learning. *Decision Support Systems*, 107:88–102, 2018. doi: https://doi.org/10.1016/j.dss.2018.01.001.

[30] Sungjin Kim, Jinkook Kim and Brent ByungHoon Kang. Malicious URL protection based on attackers habitual behavioral analysis. *Computers and Security*, 2018. doi: 10.1016/j.cose.2018.01.013.

[31] Selenium webdriver 2.39. URL `http://www.seleniumhq.org/download/`. Accessed: March 25, 2017.

[32] Knuth D. E., Morris Jr J. H. and Pratt V. R. Fast pattern matching in strings. *SIAM journal on computing*, 6(2):323–350, 1977. doi: 10.1137/0206024.

[33] Seshagiri, P., Vazhayil, A. and Sriram, P. AMA: Static code analysis of web page for the detection of malicious scripts. In *Procedia Computer Science*, pages 768–773, 2016.

[34] Aly M. Survey on multiclass classification methods. *Neural Networks*, 19:1–9, 2005.

[35] Hsu C. W. and Lin C. J. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002. doi: 10.1109/72.991427.

[36] Yuan G. X., Ho C. H. and Lin C. J. Recent advances of large-scale linear classification. In *the IEEE*, pages 2584–2603, 2012.

[37] Bottou, L., Cortes, C., Denker, J. S., Drucker, H.,

Guyon, I., Jackel, L. D.and Vapnik, V. Comparison of classifier methods: a case study in handwritten digit recognition. In *the IEEE*, pages 77–82, 1994.

[38] Fan R. E., Chang K. W., Hsieh C. J., Wang X. R. and Lin C. J. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[39] Mark Dredze, Koby Crammer and Fernando Pereira. Confidence-Weighted Linear Classification. In *25th International Conference on Machine Learning (ICML)*, pages 264–271, 2008.

[40] Crammer K., Dredze M. and Kulesza A. Multi-class confidence weighted algorithms. In *Conference on Empirical Methods in Natural Language Processing*, pages 496–504, 2009.

[41] Dahlmeier D., Ng H. T. and Ng E. J. F. NUS at the HOO 2012 Shared Task. In *Seventh Workshop on Building Educational Applications Using NLP*, pages 216–224, 2012.

[42] Confidence-weighted (cw) learning. URL http://www.comp.nus.edu.sg/~nlp/software.html. Accessed: September 20, 2017.

[43] Alexa: Alexa top global websites. URL http://www.alexa.com/topsites. Accessed: September 1, 2017.

[44] Phishtank: Join the fight against phishing. URL https://www.phishtank.com/. Accessed: September 1, 2017.

[45] Malware domain list. URL http://www.malwaredomainlist.com/forums/index.php?topic=3270.0/. Accessed: September 1, 2017.

[46] Spam domain blacklist (filtered by jwspamspy). URL http://www.joewein.de/sw/blacklist.htm/. Accessed: September 1, 2017.

[47] Sokolova M. and Lapalme G. A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, 45(4):427–437, 2009. doi: 10.1016/j.ipm.2009.03.002.

[48] Quality metrics for multi-class classification algorithms. URL https://software.intel.com/en-us/daal-programming-guide-quality-metrics-for\-multi-class-classification-algorithms. Accessed: August 20, 2017.

Dharmaraj R. Patil received master of engineering in computer science & engineering from Government College of Engineering, Aurangabad, Maharashtra, India and pursuing Ph.D. in computer engineering from North Maharashtra University, Jalgaon, Maharashtra, India. He is working as an assistant professor in the Computer Engineering Department at R.C. Patel Institute of Technology, Shirpur, Maharashtra, India. He has 14 years of teaching experience. His research interest are web security, intrusion detection and web mining. He has published many papers in international/national conferences and journals.

Jayantrao B. Patil received master of technology in computer science & data processing from IIT, Kharagpur, India and Ph.D. in computer engineering from North Maharashtra University, Jalgaon, Maharashtra, India. He has 29 years of teaching and 2 years of industrial experience. His research interest are Web catching and Web prefetching, Web data mining, text watermarking, Web usage mining, Web personalization, semantic Web mining and Web security. He has published many research papers in international/national conferences and journals.