Detecting Malicious URLs Using Lexical Analysis

Mohammad Saiful Islam Mamun$^{(\boxtimes)}$, Mohammad Ahmad Rathore, Arash Habibi Lashkari, Natalia Stakhanova, and Ali A. Ghorbani

University of New Brunswick, Fredericton, NB, Canada
{msi.mamun,mahmad.rathore,a.habibi.l,natalia,ghorbani}@unb.ca

Abstract. The Web has long become a major platform for online criminal activities. URLs are used as the main vehicle in this domain. To counter this issues security community focused its efforts on developing techniques for mostly blacklisting of malicious URLs. While successful in protecting users from known malicious domains, this approach only solves part of the problem. The new malicious URLs that sprang up all over the web in masses commonly get a head start in this race. Besides that Alexa ranked trusted websites may convey compromised fraudulent URLs called *defacement* URL. In this work, we explore a lightweight approach to detection and categorization of the malicious URLs according to their attack type. We show that lexical analysis is effective and efficient for proactive detection of these URLs. We provide the set of sufficient features necessary for accurate categorization and evaluate the accuracy of the approach on a set of over 110,000 URLs. We also study the effect of the obfuscation techniques on malicious URLs to figure out the type of obfuscation technique targeted at specific type of malicious URL.

Keywords: Malicious URLs · Lexical features · URL obfuscation · Machine learning

1 Introduction

With ubiquitous use of Internet technology, the concern with security comes to the forefront. The web has been used as a hub for a variety of malicious activities from malware hosting and propagation to phishing websites' tricking users to provide their personal user information. Malicious URLs are intended for malicious purposes. Visitors of such URLs are under the threat of being victim to certain attacks [9]. According to the latest Google Safe browsing report, Google search blacklisted over 50,000 malware sites and over 90,000 phishing sites monthly [1].

Blacklisting is a typical approach to deal with malicious websites which is simple and provide better accuracy. This technique is effective only when lists are timely updated and websites are visited extensively for finding malicious webpages. Unfortunately, it falls short for providing timely protection of online

© Springer International Publishing AG 2016

users. Since blacklists (IP addresses and URLs information) are extracted from expensive and sometimes complex filtering technologies, companies would not sell their updated list to market for free. Moreover, due to the cloaking applied on webpages and attackers recurrently change the URL and IP of the malicious webpages, new webpages are less likely to get checked against the blacklists. Besides that trusted websites may host fraudulent or hidden URL known as a *defacement* URL that contains both malicious and legitimate web pages. These URLs cannot be reached by crawling the legitimate web site within up to the third level of depth. Detection of these malicious URLs are effective when they perform in real time, detect new URLs with high accuracy, and specially recognize specific attack type (e.g., phishing URL).

Heuristic-based technique in [18] can identify newly created malicious websites in real-time by using signatures of known attack payloads. However, this approach would fail to detect novel attacks that result in zero-day exploits and signature detection is often evaded by attackers using change in patterns and obfuscation techniques.

Machine learning techniques are used to classify malicious websites through features taken from URLs, web content and network activity. The detection methods and tools which adopt the approach of patrolling web content may consume more computation time and resource. Therefore, URL based detection techniques for malicious URL detection are largely limited to classification of URLs in general or any specific attack i.e. spam [3,6,20]. Meanwhile research shows that the characteristics of malicious URLs differ with the type of technique used for exploitation (e.g., spam, adware, phishing, drive-by-downloads etc.) [19].

In this study, we adapted machine learning techniques to the detection and categorization of the malicious URLs. We look at four types of malicious use of URLs such as spam URLs, phishing URLs, website URLs distributing malware, and defacement URLs where pages belong to the trusted but compromised sites and identify a set of significant lexical features that can be used in recognizing the types of URL attack.

Obfuscation techniques used by the attacker to evade static detection in malicious URLs. Since obfuscation based features have been widely used for phishing attacks [3,23], we also study the effect of the obfuscation techniques on different type of malicious URLs to determine which attack type is mostly affected with what kind of obfuscation technique.

We select 79 features related to lexical analysis and 4 features related to obfuscation techniques for primary analysis. After applying the feature selection algorithm on the dataset, we end up with five sets of mostly *relevant* features for multi label and multi-class classification for any type of malicious URL detection.

We evaluated this approach on around 110,000 URLs taken from different sources, and achieved a prediction accuracy (with low false positive rate) of nearly 99 % in detecting URLs of the attack type and approx. 93–99 % in identifying attacks with multi-class classifier. It appears that selective lexical features can find the better accuracy for identifying the different types of URL attacks. Although obfuscation techniques are widely used in the literature as a part of

Malicious URL detection (e.g. [23]). However, from our experiment, we found no significant impact on statistical analysis.

2 Related Work

Using lexical features with a proposed classification engine gives high accuracy in classifying the URLs. Domains used for phishing purpose have shown to have different lengths and diverse location. McGrath et al. in [14] analyze the differences between benign and phishing URLs using features such as URL and domain length.

Based on the behavior of domains in phishing websites, Ma et al. in [6], manage to identifying suspicious URLs by using lexical and blacklisted host based features. They believe that there are certain red flag *keywords* tend to appear in malicious URLs such as appearing *ebayisapi* for spoofing *ebay* web pages. However, in the following works, Ma et al. in [15] use supervised learning across both lexical and host-based features. In this case, authors argue that their approach is complementary to both blacklisting and system based evaluation such as site content and behavior analysis where one cannot predict seeing only the status of previously unseen URLs. Moreover, this requires visiting potentially dangerous sites. They figure out that using appropriate classifiers, it is possible to identify most predictive features for automatically classifying malicious or benign URLs.

In [5], authors use lexical, host and page-content based features for identifying malicious URLs collected from spam messages in twitter and emails. Choi et al. in [8], present a machine learning method to detect malicious URLs and identify attack types such as spamming, phishing, and malware. Their studies include multiple types of malicious URLs using features from six different areas namely lexicon, link popularity, web page content, DNS, DNS fluxing and network traffic. However, their result shows that using lexical features yield *lower* accuracy for spam and malware URL dataset.

Authors in [21] use descriptive features of URL to complement lexical features. They combine the lexical information and static characteristics of URL string to classify malicious URLs. Without host and content based analysis, in this experiment, they were able to deal with two million URLs in five minutes and their proposed method misses around 9 % of malicious instances.

The effectiveness of machine learning based phishing detection with known protected websites has been studied by Chu et al. in [10]. Based on only lexical and domain features authors propose several highly effective features with detection rate over 91 %. In [22], authors study hidden fraudulent URLs which are embedded to the trusted URLs and also defacement URLs which are legitimate pages belonging to trusted but compromised web sites. They provide a dataset that can be useful for evaluating the performance of a classifier for aforementioned malicious URLs.

Obfuscation techniques are commonly used by spammers and scammers in order to obscure (hide and confuse) any malicious URL. It's often appeared in unsolicited emails, ad-related URLs and web-site owner intending to evade recognition of a linked address. Obfuscated URL parts can help the malicious URL parts to evade the detection system. In this work, our target was to see the obfuscation techniques used by Malicious, Phishing and Spam URLs separately. Gerara et al. in [23] mentioned several URL obfuscation techniques used in malicious URLs. Le et al. in [3] identified four of them that are commonly used in domain are obfuscating with IP address, another domain, large host names or, unknown and misspelled. Ma et al. [6] mentioned three prominent obfuscation types for avoiding detection. They are benign tokens in the URL, free hosting services used, sites with international TLD.

However, the malicious owner can also use these obfuscation techniques to evade detection and tempt users to hit. Therefore, Su et al. in [2] proposes to dissect URLs to several portions and use logistic regression for detection. Apart from the work of Choi et al. in [8] where authors evaluates the performance of a classifier for discriminating three types of malicious URLs (Spam, Phishing, Malware), most of the related works focus on either *malicious* URLs *in general* or any specific type of URLs (e.g. Phishing). Regarding feature selections, most of the works depend on various kind of features: lexical, content, obfuscation, DNS etc. In this work, we consider four different types of malicious URLs (Defacement, Spam, Phishing, Malware) for experiments. By applying only static lexical features for classifying URLs (to achieve high performance), our malicious URL detector produces a promising as well as competitive performance to some existing works.

3 Background

3.1 Lexical Analysis

Lexical features are the textual properties of URL such as length of hostname, URL length, tokens found in the URL etc. Due to lightweight computation[1], safety[2] and high classification accuracy lexical features become one of the most popular sources of features in machine learning [3].

Features collected from URLs are not dependent on any application like email, social networking websites, games etc. Since many malicious URLs have short life span, lexical features remain available even when malicious webpage are unavailable [10].

In this research, we study five main components of the URLs to be inspected for analysis: URI, domain, path, argument and file name. Following are the brief description of all the features considered for analysis.

[1] Using web content as features requires downloading and analysis of page contents. Moreover inspecting millions of URL and its contents per unit of time may create a bottleneck.

[2] Access to malicious webpage may cause risk since such webpages may contain malicious content such as Javascript functions.

– *Entropy Domain and Extension:* Malicious websites often insert additional characters in the URL to make it look like a legitimate. e.g, CITI can be written as CIT1, by replacing last alphabet I with digit 1. English text has fairly low entropy i.e., it is predictable. By inserting characters the entropy changes than usual. For identifying the randomly generated malicious URLs, alphabet entropy is used.
– *CharacterContinuityRate:* Character Continuity Rate is used to find the sum of the longest token length of each character type in the domain, such as abc567ti $= (3 + 3 + 1)/9 = 0.77$. Malicious websites use URLs which have variable number of character types. Character continuity rate determine the sequence of letter, digit and symbol characters. The sum of longest token length of a character type is divided by the length of the URL [21].
– *Features related with Length Ratio:* The length ratio of the parts of URL is computed to find the abnormal parts [21]. The combination of URL part consist of argument, path, domain and URL such as *argPathRatio* (Ratio of argument and path), *argUrlRatio* (Ratio of argument and URL), *argDomainRatio* (Argument divided by domain), *domainUrlRatio* (Domain divided by URL), *pathUrlRatio* (Path divided by URL), *PathDomainRatio* (Path divided by Domain).
– *Features related to count of Letter, Token and Symbol:* The frequency of characters in the URL are calculated in the form of letters, tokens and symbol [5,10,18]. These characters are categorized and counted from these components of URLs:
  • *Symbol Count Domain:* A dictionary of delimiters such as ://.:/?=,;()]+ are calculated from domain. Phishing URLs e.g. have more dots compared to benign ones [15,17].
  • *Domain token count:* Tokens are taken from the URL String. The Malicious URLs use multiple domain tokens. Number of tokens in the domains are calculated.
  • *Query Digit Count:* Number of digits in the query part of the URL.
  • *tld:* Some phishing URL use multiple top level domain within a domain name.
– *Number Rate of Domain, DirectoryName, FileName, URL, AfterPath:* Number rate calculate the proportion of digits in the URL parts of directory name, domain, filename, URL itself and part after the path. [21].
– *Features related to Length:* Length of URL gets longer due to addition of variables or redirected URL. [11,18]. such as, Length of URL (url Len), domain (domain Len) and file name (file Name Len), Arguments' LongestWordLength[3], Longest Path Token Length [8], Average length of path token [10] (avgpathtokenlen).
– *ldl getArg:* In phishing URLs masquerading is done by adding digits in the letters. For detection of these deceiving URLs, sequence of letter digit letter in URL and path is calculated [21].

[3] URLs which originating from pages that are written in server side scripting languages, often have arguments [3]. The longest variable value length from arguments of URL is calculated.

– *spcharUrl:* URLs use special characters which are suspicious such as // and they have higher risk of redirection [11].

Table 1. Feature selection for lexical analysis

Dataset	Features
Spam (CfsSub + Best First)	Domain token count, tld, ldl getArg, Number of Dots in URL, delimiter path, Symbol Count Domain
Phishing (CfsSub + Best First)	Domain token count, tld, url Len, domain length, file Name Len, dpath url Ratio, Number of Dots in URL, Query Digit Count, Longest Path Token Length, delimiter Domain, delimiter path, Symbol Count Domain, Entropy Domain
Malware (CfsSub + Best First)	Domain token count, tld, url Len, arg Doman Ratio, Number of Dots in URL, Number Rate Domain, Symbol Count Domain, Entropy Domain, Entropy Extension
Defacement (CfsSub + Best First)	Domain token count, avgpathtokenlen, tld, ArgUrlRatio, NumberofDotsinURL, Arguments LongestWordLength, spcharUrl, delimeter Domain, delimeter path, NumberRate DirectoryName, SymbolCount Domain, Entropy Domain
All (Infogain + Ranker)	Entropy Domain, argPathRatio, ArgUrlRatio, ArgDomanRatio, pathurlRatio, CharacterContinuityRate, NumberRate FileName, domainUrlRatio, NumberRate URL, PathDomainRatio, NumberRate AfterPath, avgpathtokenlen

3.2 Obfuscation

Obfuscation is used as a common method for masking malicious URLs. An attacker intending to evade static analysis on lexical URL features use obfuscation techniques so that malicious URLs become statistically similar to the benign ones [11]. The obfuscation techniques on URLs is analyzed for the intent of malicious activity in this research. We analyzed mainly two type of URL obfuscation techniques used by attackers:

– Type I: Obfuscating the hostname:
  - Obfuscating the domain with IP (IP_obfus): In this type of attack host name is obfuscated by the IP address instead of domain name. Use of IP address as a domain name of the URL alludes the owner is tempting to access private information of the user [16,18].

  - Obfuscating domain with legitimate name (Prefix_obfus): In this type of attack the domain is a prefix with a legitimate domain such as brand name. The purpose of brand name prefix is to find the URL which use legitimate, benign domain in their prefix. Therefore, the user might be tempted to click on the URL through the brand prefix. Legitimate domain name such as domain name from Alexa is used in variation to make the malicious URL look like a legitimate one [12]. For example, in the following legitimate URL http://widget.b2b.com/relationship/, *b2b* is a benign first level domain name known as a *brand name*. However, in case a custom domain name that has name or brand in it, http://detail.b2b.hc360.com/detail/ or http://detail.b2b1.com/detail/, although *hc360* or *ab2b* is not a benign domain, a brand name *b2b* is used as a *second level* domain or a *prefix* to distract users.

- Type II: Obfuscating the directory:
  - Obfuscation with encoding (Encoding_obfus): In this attack type string of characters are obfuscated by the use of alternate encoding schemes i.e., Unicode, Base64, Hexadeimal, Decimal characters [7]. Unicode encoding allow the characters to be referenced and saved in multiple bytes. UTF-8 is a common encoding format. It preserves the full ASCII character code range. The standard character can be encoded in longer escape-code sequence.
  - Obfuscating using redirected URLs (RedirectURL_obfus): The content coming between protocol name and the '@' character is ignored which allow the addition of obfuscated URLs. URLs for redirection are embedded in the links. These links deviate the user to link which has no link to the actual page [18]. These redirected URL are attached in malicious links by attackers [13].
  - Obfuscating using hexadecimal form (Hex_obfus): Characters of a path is represented by their corresponding numbers in the hexadecimal form where numbers are preceded by a "%" symbol to recognize a hexadecimal representation of the character. For instance, "%63ur%65" is the hexamdedimal form of "cure". It is mainly used to use spaces and special characters in the URL. However, the same techniques can be used to inject malicious items.
  - Authentication type obfuscation (AuthString_Obfus): This type of obfuscation is used for automatic authentication when login name or password is required for accessing a web page. But if the site requires no authentication, the additional authentication text will have no effect e.g. http://www.xyz.com/index.htm.

4 Experiment

4.1 Dataset

Around 114,400 URLs were collected initially containing benign and malicious URLs in *four* categories: Spam, Malware, Phishing and Defacement. Four single-class datasets by mixing benign and malicious URLs and one multi-class dataset

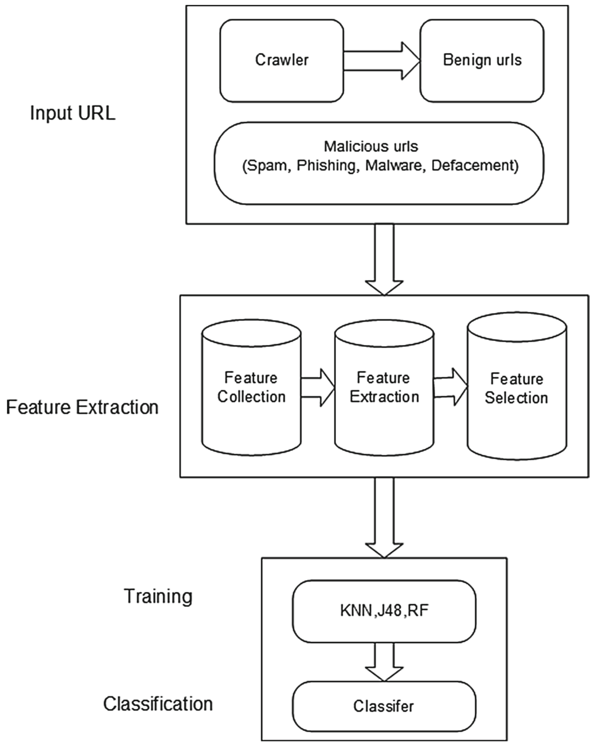


Fig. 1. URL classification architecture

by combining all four malicious URLs and benign URLs were generated for experiment (See Fig. 1).

- Benign URLs: Over 35,300 benign URLs were collected from Alexa top websites. The domains have been passed through Heritrix webcrawler to extract the URLs. Around half a million unique URLs are crawled initially and then parsed to remove duplicate and domain only URLs. Later the extracted URLs have been checked through virustotal to filter the benign URLs.
- Spam URLs: Around 12,000 spam URLs were collected from publicly available web spam dataset in [24].
- Phishing URLs: Around 10,000 phishing URLs were taken from OpenPhish website [26] which is a repository of active phishing sites.
- Malware URLs: More than 11,500 URLs related to malware websites were obtained from DNS-BH [25] which is a project that maintain list of malware sites.
- Defacement URLs: In [27], authors select 2500 URLs provided by Zone-H [28] and extend the lists by adding URLs of pages reached by crawling the compromised sites up to the third level. After necessary filtration (e.g. URLs whose path is empty or equal to index.html, URLs whose domain is an IP address),

they labelled 114,366 URLs as *Defacement*. However, for our experiment we randomly choose 45,457 URLs.

4.2 Features Selection, Test and Validation

The motive of evaluation and search strategy is to find the features which are significant and contribute most in the analysis. In this paper, we used CFSSubsetEval and Infogain as feature selection algorithms.

CfsSubsetEval evaluates the worth of a subset of features by considering the individual predictive ability of each feature along with the degree of redundancy. Infogain searches the space of feature subsets by greedy hill-climbing strategy augmented with a backtracking facility. Later a ranker ranks features by their individual evaluations.

Initially 79 features were selected from components such as URL, domain, path, file-name and argument. After applying feature selection algorithm on each dataset, different sets of features have been chosen (See Table 1) for further experiment.

Classification of data is done through two groups of algorithms. K-Nearest Neighbours algorithm (KNN) [30], a pattern recognition method used for classification by assigning weight to the neighbours according to the their contributions. Euclidean function was used as a distance function for KNN. Tree based classifiers namely C4.5 [29] and RandomForest [31] used to present results as a tree. To evaluate the quality of the classifiers, we used two common metrics: Precision (Positive Predictive value) and Recall (Sensitivity).

$$\text{Precision}(\texttt{Pr}) = \frac{TP}{TP + FP} \qquad \text{Recall}(\texttt{Rc}) = \frac{TP}{TP + FN}$$

During training we tuned up several parameters of Random Forest to achieve better and efficient model with less error. For example, the number of trees to be generated is set to 80 for Spam, 100 for Malware and Phishing, 150 for Defacement, 120 for multi-class datasets.

We divided our experiment in testing and validation, therefore, split the datasets accordingly (80 % for test - 20 % for validation) using pre-processing function (resample with noReplacement) in Weka.

5 Analysis and Results

As mentioned earlier two set of features have been selected as part of the analysis: (i) Analyzing lexical features to recognize benign and malicious URLs based on selected set of features in Table 1, (ii) Analyzing Obfuscation techniques against different attack types.

Table 2. Classification results (single-class)

Dataset	**Algorithm**	**Result**	
		Pr	Re
Spam	C4.5	0.98	0.98
	KNN	0.98	0.98
	RF	**0.99**	**0.99**
Phishing	C4.5	0.97	0.97
	KNN	0.97	0.97
	RF	**0.99**	**0.99**
Malware	C4.5	0.98	0.98
	KNN	0.98	0.98
	RF	**0.99**	**0.99**
Defacement	C4.5	0.99	0.99
	KNN	0.99	0.99
	RF	**0.99**	**0.99**

(a) Lexical Features

Dataset	**Algorithm**	**Result**	
		Pr	Re
Spam	C4.5	0.752	0.388
	KNN	0.753	0.388
Phishing	C4.5	0.382	0.844
	KNN	0.383	0.844
Malware	C4.5	0.857	0.398
	KNN	0.857	0.398
Defacement	C4.5	0.795	0.741
	KNN	0.795	0.741

(b) Obfuscation Features

Table 3. Classification results based on lexical features (multi-class)

Dataset	Labels	C4.5		KNN		RF	
		Pr	Rc	Pr	Rc	Pr	Rc
Multi class	Spam	0.96	0.971	0.96	0.97	0.962	0.986
	Phishing	0.92	0.856	0.92	0.85	0.926	0.928
	Malware	0.96	0.97	0.96	0.97	0.979	0.983
	Defacement	0.93	0.97	0.93	0.97	0.969	0.973
	Average	0.94	0.94	0.94	0.94	**0.97**	**0.97**

5.1 Lexical Analysis

Table 2a shows the results of lexical analysis on single-class datasets, the accuracy of the all datasets with selected set of features are higher than 97 %. For example in Spam and Malware datasets the accuracy were more than 98 % with 6 and 9 features or in the Defacement dataset, accuracy was 99 % with 13 features. The same analysis has been done in the multi-class dataset which was the combination of all four different types of malicious URLs with Benign ones. As Table 3 shows, the average of accuracy in all ML algorithms are more than 95 %.

We observe that tree based classifiers, with Random Forest yields highest accuracy among the classifiers tested. While efficient in identifying certain type of URL individually (≈99 %), Random Forest has also outperformed as a multiclass classifier (≈97 %). Among other classifiers examined, KNN and C4.5 classifiers have approximately the same performance (≈94 %) for multiclass classifer with the worst accuracy for phishing (around 80 %). Since Random Forest appears

Table 4. Soft pediction based on SD (C0 = Benign, C1 = Dataset class)

Dataset	Filters									
	Yellow ($0 < \texttt{SD} \leq .1$)		Green ($.1 < \texttt{SD} \leq .2$)		Orchid ($.2 < \texttt{SD} \leq .3$)		Blue ($.3 < \texttt{SD} \leq .4$)		Red ($\texttt{SD} > .4$)	
	C0	C1	C0	C1	C0	C1	C0	C1	C0	C1
Defacement	847	640	341	247	119	95	50	27	17	21
Phishing	2077	725	810	534	308	295	192	285	72	125
Malware	1793	1496	997	804	349	288	142	111	75	43
Spam	3974	94	124	177	54	218	39	158	2	2

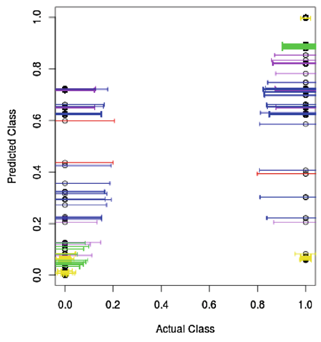
to be effiencient in both *binary* and *multiclass* classification, we select Random Forest classifier for further investigation.

Confidence interval of a prediction known as *prediction interval*, a well-defined concept in statistical inference, estimates the prediction interval based on member decision tree scores. At the time of prediction Random Forest produces a *hard decision* based on the maximum votes of the individual trees for a class to get elected [4]. However, a *soft prediction* can also be determined from the individual trees' voting which provides a confidence score for the prediction. This confidence score can be used for *hard decision* once it exceeds a threshold value. For this experiment, we train a regression-type Random Forest model for the datasets. For all binary classifiers (Spam, Phishing, Malware, Defacement), Benign is labelled as 0 and any respected class (e.g. Spam) is labelled as 1.

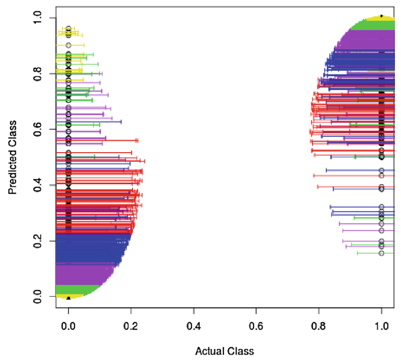
The scatter plot of experiment ("Actual class" versus "Predicted class") are given in the Fig. 2 below. Figure 2 depicts the data points overlaid with error bars. The error bars corresponding to a *soft prediction* is represented by a Standard Deviation (SD) of uncertainty for a certain class. Due to the large number of data points and to achieve a holistic view of data, we filter the result in four steps:

– Yellow-filter contains data points whose SD is greater than 0 but less than or equal to .1
– green-filter contains data points whose SD is greater than .1 but less than or equal to .2
– Orchid-filter subset contains data points whose SD is greater than .2 but less than or equal to .3
– Blue-filter contains data points whose SD is greater than .3 but less than or equal to .4
– Red-filter contains data points whose SD is greater than .4

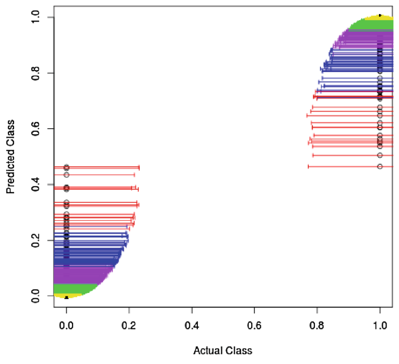
Results of four binary classifiers are given in Table 4. Lifted SD indicates considerable fluctuating among member decision tree scores. Higher lifting can be realized with prospective statistical outliers.



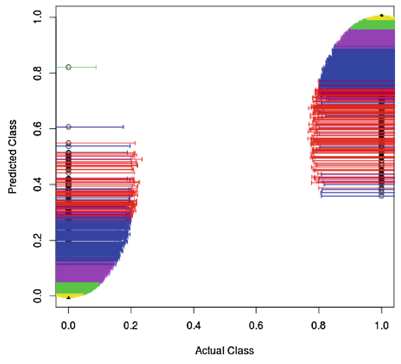
(a) Spam



(b) Malware



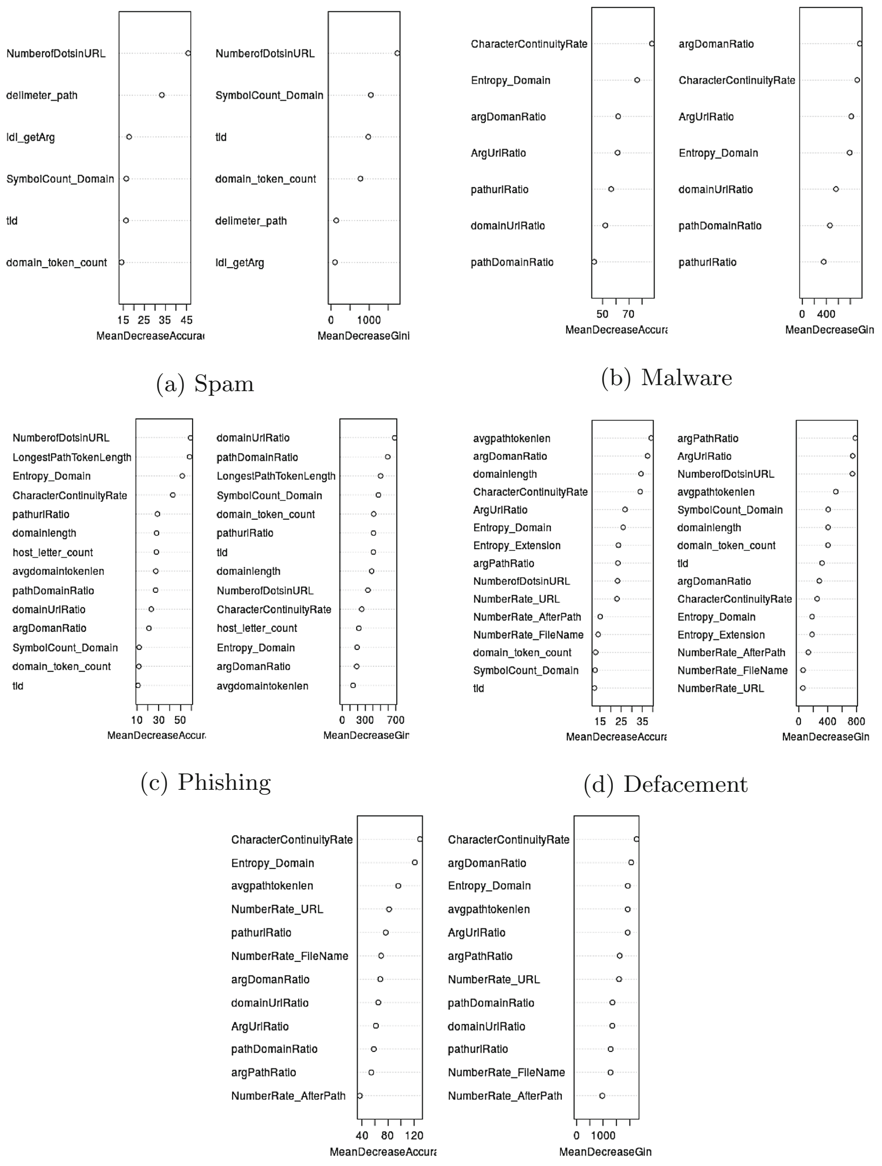
(c) Defacement



(d) Phishing

Fig. 2. Predection interval based on decision tree score (Color figure online)

Most of the URLs are grouped into the closest range of its respective class (0 or 1). For instance, in case of Defacement dataset a total of 13,308 URLs out of 15,712 observed in extremity of the range (aligned with average) while 2404 URLs with any SD and in the worst case Phishing dataset shows a total of 5423 URLs out of 15,368 observed with any SD. In addition there is no URL data point with (SD $> .5$) and the majority of URLs are overlaid with yellow and Green error bars. It is interesting to note that the highest SD (Red filter) corresponding to overestimating/underestimating errors, has very few of either kind of URL, in the range of 2 (Spam) and 75 (Malware) for Benign URL and 2 (Spam) to 125 (Phishing) for any other URLs. This ensures that soft prediction is not uniformly distributed. Some range of *soft prediction* values where the SD is very small for example Yellow, Green, and Orchid filters must be used for an infallible prediction. Considering these facts, we can answer how much risky a URL is? If the soft prediction is closer to 1 with a small threshold value of SD



(a) Spam

(b) Malware

(c) Phishing

(d) Defacement

(e) All

Fig. 3. Random forest variable importance

score (e.g. up to Orchid filter), the URL is risky. In the opposite way, the closer the URL is to 0 with a small SD score, the more secure it is.

Random forest computes various measure of variable importance that can be very informative to understand how much each variable contributing to the accuracy of the model. Figure 3 exhibits two different Variable importance graph for all four binary-class and a multi-class: MeanDecreaseAccuracy that is the mean decrease in accuracy and MeanDecreaseGini that is the Gini index or *Mean decrease in Node Impurity*.

5.2 Obfuscation

As mentioned in Sect. 3.2, we study six obfuscation types (Brand name, AuthString, IP, RedirectURL, Encoding, Hex) for analysis. However, no significant output has been noticed for obfuscation type AuthString and Hex for any of our datasets.

Obfuscation detection rate found on all four dataset is shown in the Table 5. As the result shows, Spam URLs contains the most number of Prefix_obfus (brand name) obfuscation (38 %), that refers using of legitimate domains in the spam URLs to trick the user for visiting the webpage. The phishing URLs have some amount of brand name (13 %) and redirect obfuscation (5 %). The malware URLs have a large number of encoding obfuscation (40 %), which shows that malware URLs use a number of encoding techniques to evade from static detection. A few percentage of URLs shows combination of multiple encoding techniques applied in a URL.

Table 5. Obfuscation detection rate in the malicious URL

Obfuscation technique	Dataset			
	Defacement	Spam	Phishing	Malware
Brand name	0.71 %	**38 %**	**13.33 %**	1.58 %
Redirect	0.64 %	5.20 %	4.61 %	0.08 %
Encoding	**5.57** %	4.48 %	2.04 %	**39.01 %**
IP	0.73 %	-	**1.54%**	0.29 %

Using AttributeSelection with InfoGainAttributeEval as attribute evaluator and Ranker as search method, we found three features Encoding, Brand name, RedirectURL to be useful. We try to use machine learning classifiers on the obfuscation features with no promising result (See Table 5). For instance, C4.5 can distinguish phishing URL with 84 % TP rate (the best result in our experiment). However, the Precision (40 %) and ROC area (.65) value for the same classifier is too low to accept the result.

5.3 Comparison

Our research result is very close to that of the work done by Choi et al. in [8]. Although a major part of our experiment datasets (benign, phishing, malware, a portion of spam) are identical, we have extended our dataset with Defacement dataset. Regarding lexical classification outcomes of Choi et al. (Spam 73 % Phishing 91.6 % and Malware 70.3 %), authors did not mention precisely whether their result stems from applying multi-class or single-class classifier. Note that using multi-class classification with additional dataset must degrade the overall performance and accuracy. However, our Random Forest classifier outperforms their lexical feature results in either case of individual and aggregated (multi-class) classifiers yielding around 99 % and 97 % accuracy respectively even with an addition of Defacement URL dataset.

6 Conclusion

This paper explored an approach for classifying different attack types of URL automatically as benign, defacement, spam, phishing and malware through supervised learning relying on lexical features. This technique is an addon for the blacklist techniques, in which new malicious URLs cannot be identified and efficient for analyzing large number of URLs. Selected feature sets applied on supervised classification on a ground truth dataset yields a classification accuracy of 97 % with a low false positive rate. Our prediction interval filtering experiment can also be helpful to improve classifier accuracy. In addition, it can be extended to calculate risk rating of a malicious URL after parameter adjustment and learning with huge training data. Despite random forest classification accuracy is able to identify approx. 97 % of the malicious or benign URL, by using proper SD filter we could reach up to around 99 % accuracy. As future work we are planning to develop a real time tool for computing SD filter dynamically and detection of malicious URLs.

References

1. Google Safe Browsing Transparency Report (2015). www.google.com/transparencyreport/safebrowsing/
2. Su, K.-W., et al.: Suspicious URL filtering based on logistic regression with multi-view analysis. In: 8th Asia Joint Conference on Information Security (Asia JCIS). IEEE (2013)
3. Le, A., Markopoulou, A., Faloutsos, M.: PhishDef: URL names say it all. In: Proceedings IEEE, INFOCOM. IEEE (2011)
4. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
5. Thomas, K., et al.: Design and evaluation of a real-time URL spam filtering service. In: Proceeding of the IEEE Symposium on Security and Privacy (SP) (2011)
6. Ma, J., et al.: Identifying suspicious URLs: an application of large-scale online learning. In: Proceedings of the 26th Annual International Conference on Machine Learning. ACM (2009)
7. Nunan, A.E., et al.: Automatic classification of cross-site scripting in web pages using document-based and URL-based features. In: IEEE Symposium on Computers and Communications (ISCC) (2012)
8. Choi, H., Zhu, B.B., Lee, H.: Detecting malicious web links and identifying their attack types. In: Proceedings WebApps (2011)
9. Huang, D., Kai, X., Pei, J.: Malicious URL detection by dynamically mining patterns without pre-defined elements. World Wide Web **17**(6), 1375–1394 (2014)

10. Chu, W., et al.: Protect sensitive sites from phishing attacks using features extractable from inaccessible phishing URLs. In: IEEE International Conference on Communications (ICC) (2013)
11. Xu, L., et al.: Cross-layer detection of malicious websites. In: Proceedings of the Third ACM Conference on Data and Application Security and Privacy. ACM (2013)
12. Garera, S., et al.: A framework for detection and measurement of phishing attacks. In: Proceedings of the ACM Workshop on Recurring Malcode (2007)
13. Radu, Vasile: Application. In: Radu, Vasile (ed.) Stochastic Modeling of Thermal Fatigue Crack Growth. ACM, vol. 1, pp. 63–70. Springer, Heidelberg (2015)
14. Kevin, M.D., Gupta, M.: Behind phishing: an examination of phisher modi operandi. In: Proceedings of the 1st Usenix Workshop on Large-Scale Ex-ploits and Emergent Threats (2008)
15. Ma, J., et al.: Beyond blacklists: learning to detect malicious web sites from suspicious URLs. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM (2009)
16. Davide, C., et al.: Prophiler: a fast filter for the large-scale detection of malicious web pages. In: Proceedings of the 20th International Conference on World Wide Web. ACM (2011)
17. Xiang, G., et al.: CANTINA+: a feature-rich machine learning framework for detecting phishing web sites. ACM Trans. Inf. Syst. Secur. (TISSEC) **14**(2), 21 (2011)
18. Abdelhamid, N., Aladdin, A., Thabtah, F.: Phishing detection based associative classification data mining. Expert Syst. Appl. **41**(3), 5948–5959 (2014)
19. Eshete, B., Villafiorita, A., Binspect, K.W.: Holistic Analysis and Detection of Malicious Web Pages. Security and Privacy in Communication Networks. Springer, Heidelberg (2012)
20. Cao, C., Caverlee, J.: Behavioral detection of spam URL sharing: posting patterns versus click patterns. In: IEEE International Conference on Advances in Social Networks Analysis and Mining (ASONAM) (2014)
21. Lin, M.-S., et al.: Malicious URL filtering- a big data application. IEEE International Conference on Big Data (2013)
22. Enrico, S., Bartoli, A., Medvet, E.: Detection of hidden fraudulent urls within trusted sites using lexical features. In: Proceeding 18th International Conference on Availability, Reliability and Security (ARES). IEEE (2013)
23. Garera, S., Provos, N., Chew, M., Rubin, A.: A framework for detection and measurement of phishing attacks. In: Proceedings of the ACM workshop on Recurring malcode, pp. 1–8. ACM (2007)
24. WEBSPAM-UK2007 dataset. http://chato.cl/webspam/datasets/uk2007/
25. Malware domain dataset. http://www.malwaredomains.com/
26. OpenPhish dataset. https://openphish.com/
27. Davanzo, M., Bartoli, A.: Anomaly detection techniques for a web defacement monitoring service. Expert Syst. Appl. (ESWA) **38**(10), 12521–12530 (2011)
28. Zone-h, unrestricted information. http://www.zone-h.org/
29. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers Inc., San Francisco (1993)
30. Keller, J.M., Gray, M.R., Givens, J.A.: A fuzzy k-nearest neighbor algorithm. IEEE Trans. Syst. Man Cybern. **4**, 580–585 (1985)
31. Liaw, A., Wiener, M.: Classification and regression by randomForest. R news **2**(3), 18–22 (2002)