Towards Fighting Cybercrime: Malicious URL Attack Type Detection using Multiclass Classification

1st Tariro Manyumwa
College of Computer Science and Technology
Zhejiang University
Hangzhou, China
tariro@zju.edu.cn

2nd Phillip Francis Chapita
School of Sciences and Big Data Science
Zhejiang University of Science and Technology
Hangzhou, China
francisphil2020@gmail.com

3rd Hanlu Wu
College of Computer Science and Technology
Zhejiang University
Hangzhou, China
wuhanlu@zju.edu.cn

4th Shouling Ji
College of Computer Science and Technology
Zhejiang University
Hangzhou, China
sji@zju.edu.cn

2020 IEEE International Conference on Big Data (Big Data) | 978-1-7281-6251-5/20/$31.00 ©2020 IEEE | DOI: 10.1109/BigData50022.2020.9378029

***Abstract*—Malicious Uniform Resource Locators (URLs) remain one of the most common threats to cybersecurity. They are commonly spread through phishing, malware and spam. One popular way to detect malicious URLs is through blacklists. Blacklists maintain records of previously known malicious URL reputations. These lists are however shortcoming when there is need to detect newly generated malicious URLs. For that reason, modern research has resorted to training machine learning algorithms to detect malicious URLs. In this paper, we contributed towards the detection of malicious URLs using URL based features in a multiclass classification setting. We focused on three popular URL attack types which are phishing, spam and malware. Our work can be used as a supplementary tool in new or existing anti-phishing, anti-spam and anti-malware detection platforms. We compared the performance of the following ensemble learners: Extreme Gradient Boosting (XGBoost), Adaptive Boosting (AdaBoost), Light Gradient Boosting (LightGBM) and Categorical Boosting (CatBoost). We evaluated the performance of some URL features that we referred to as our features. These included priority features like Kullback-Leibler Divergence (KL divergence), bag of words segmentation and other word-based features. Results showed that our features performed better when compared to experiments we conducted without our features. We trained these algorithms on 126 983 URLs from benchmark datasets and all four learners returned an overall accuracy above 0.95.**

***Index Terms*—Multiclass Classification, Ensemble Learners, KL divergence, Malicious URLs, Attack types**

I. Introduction

Most business enterprises rely on the Internet services available on the World Wide Web (WWW). These services are however susceptible to cyberttacks. Most cyberattacks usually occur when users click on malicious URLs. URLs are used on the WWW to access legitimate resources and when they are used for other reasons other than that they pose threats to data availability, controllability, confidentiality and integrity. The 2019 APWG fourth quarter Phishing Activity Trends Report analysis as listed below, states that July through October was the worst period for phishing ever seen in a period of 3 years[1].

- Webmail, payment, financial institutions, and Software-as-a-Service sectors continue to be targeted the most by Phishing attacks. While social Media targets grew and doubled over the course of the year.
- Phishing sites now use SSL protection. Therefore, users cannot rely on SSL alone to determine the safety of a site.
- The number of unique phishing report e-mails went up from 112,162 in the second quarter to 132,553 in the fourth quarter.

Also, according to the 2019 semantic monthly Internet Security Threat Report [2],

- The number of Web Attacks went up by 56% while the number of attack groups using destructive malware rose by 25%. One in ten URLs are malicious.
- In 2018, small organization employees were more likely to fall victim to spam, phishing and email malware as compared to large scale organizations.
- The phishing rate decreased from 1 in 2,995 in 2017 to 1 in 3,207 in 2018 for emails.
- The email spam rate went up to over 54.5% in a month's time period.

Evidently, there is a continuous increase in web attacks launched through phishing, spam and malware. Phishing URLs are usually used to gain unauthorised access to personal information for example banking details, passwords and names.

[1]https://docs.apwg.org/reports/apwg/trends/
[2]https://www-west.symantec.com/content/dam/symantec/docs/reports/

When attackers obtain this information, they proceed to stealing money or logging into private institutions like government websites, school websites and hospital medical report databases. Spam URLs normally intend to perform unauthorized advertisements especially using well known brands like Amazon, Taobao, Alibaba and PayPal. Malware URLs usually spread malicious software that once downloaded will infect the host. Malicious software is spread with the intention to launch silent attacks on a user's computer by installing malicious programs. Other web attacks can be spread through drive-by downloads, cross site scripting (XSS), JavaScript obfuscation and clickjacking as surveyed in [1]. Researchers in the security industry continue to invest in various approaches towards the detection of malicious URLs. A well-known detection mechanism is the use of blacklists. Blacklists maintain records of malicious URLs. However, they fail to detect newly generated ones. Over 90% of users click on malicious links before they have been added to blacklists. In addition, blacklists rely on human feedback and maintenance which can be unreliable and labor intensive [2]. Some studies used machine learning approaches to solve malicious URL detection tasks. Using existing lists of known malicious and legitimate URLs, they extracted URL features then trained classification algorithms to learn patterns and attributes that could differentiate between these URLs. The quality of detection results relies tremendously on the amount, processing and quality of data samples [3].

In this paper, we address the detection of malicious URLs as a multiclass classification task and study the performance of four well-known ensemble learners: Extreme Gradient Boosting (XGBoost), Adaptive Boosting (AdaBoost), Light Gradient Boosting (LightGBM) and Categorical Boosting (CatBoost). Furthermore, we examine and compare the results with and without our features. Our features perform better especially when using AdaBoost and XGBoost. The important features in our work are obtained through Kullback-Leibler Divergence (KL divergence), bag of words segmentation and word-based features. We also observe that KL divergence returns better results when used with the exclusion of Shannon entropy features so we add Shannon entropy to the "without our features part of the experiments". The results with our features do not include Shannon entropy as a feature. All features used in our work are extracted directly from the URL string without utilising external features from third-party sources. It is imperative to be able to detect URL attack types as this will provide insights on the types of features and models that best suit specific requirements. It also allows organisations to assign higher priority to high-risk attack types. An organization may be reluctant when dealing with spam URLs but instead take immediate action when it comes to malware and phishing URLs. Fig.1 shows some of the techniques used by cyberattackers to disguise malicious URLs as legitimate ones.

Paper Organization: The following section outlines our Motivation. Section III gives an overview of previous related work in malicious URL detection. Data Collection and classification techniques are covered in Section IV. The experimental results and evaluation are discussed in section V and the conclusion is outlined in section VI together with future work.

https://accnt-fomrs-subcrlptions.gq/PrefixLoginAuthPrivacy
Actual words: account, forms, subscriptions
https://noreply.paypal.com.service.paypol.co/app/index
Actual word: paypal
https://secur3noticeal3rtinf0rtmat1onlimit3d.seiugbiesugbes.org/signin/
Actual words: secure, alert, information, limited
https://wwvv-paypal.serveirc.com/? some
Actual words: www, service
https://psycheforce.com/site/customer_center_logim/.user174896/payment.php
Actual word: login

Fig. 1. An example of ways in which some attackers manipulate URLs through obfuscation, use of digits and look-alike brand names

II. MOTIVATION

This research aims to contribute towards the detection of malicious URL attack types. We use URL based features extracted from the bag of words segmentation, word-based features and KL divergence. KL divergence is used to calculate the character distributions between URL attack types in our datasets. [4] formulated a weakly-supervised approach using KL divergence. It was also used by [5] in their approach to compare standard URLs and phishing URLs. They used the final KL divergence value as a feature. Researchers in [6] also used KL divergence for their web spam detection. They applied it to measure the two text units of the source and target pages. [7] and [8] also used it in their work. Our main goal is to use only the URL string for feature extraction without having to access external services like the WHOIS database, page source and URL Geographical Locations. The framework of our multiclass system is shown in Fig. 2. Relying entirely on the URL string improves detection speed and efficiency. Other reasons we only pay attention to URL-based features are as follows:

1) Extraction of these features is faster than trying to extract from external servers. Most external servers may be unavailable when trying to query them.
2) Some URLs only last for a very short period of time. Therefore, servers might not have them registered and stored in their databases. Training models to learn from the URL string itself then becomes a better alternative.
3) Some servers tend to return missing values for some URL requests. For example, we attempted to query the WHOIS server and some details like dates and addresses returned missing values.

We also contribute by using ensemble learners in a multiclass attack type detection setting and compare their performances. As far as we know, our work is the first to compare four ensemble learners using bag of words segmentation, word-based features and KL divergence. Apart from these features, we also add existing URL features from previous work to our feature sets. Some research did use ensemble learners like AdaBoost and XGBoost classifiers but mainly for binary classification and in most scenarios phishing URL detection. Our

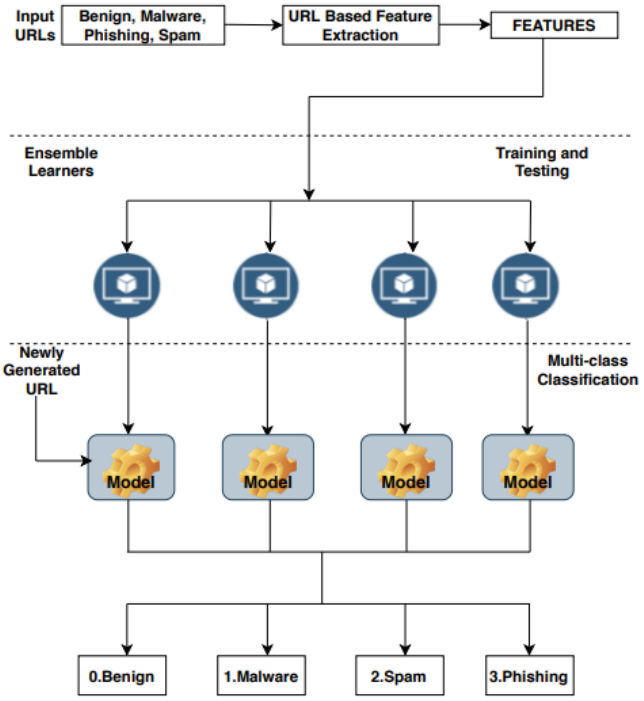

Fig. 2. Framework of our URL Multiclass Classification System

work compares the performance of these ensemble learners on phishing, spam and malware. We also compare their performances with and without our features. In [9], the authors found that phishing URL analysis was significantly different from spam website detection through URL and domain analysis. Studies in [10] and [11] explored malicious URL types and used multilabel and multiclass classification.

III. Related Work

In this section we explore previous literature that used machine learning algorithms to detect malicious URLs specifically for phishing, spam and malware.

A. Phishing URL Detection

Cyberattackers normally use phishing URLs to target financial institutions, government affairs offices and medical records. They intend to gain unauthorized access to personal details such as passwords, usernames, ID and account numbers. Researchers in [12] conducted their experiments using the Random Forest and Support Vector Machine algorithms and reported up to 0.85 precision and 0.87 recall as well as 0.90 precision and 0.88 recall for the respective algorithms. They used URL features and considered all forms of domain squatting in their work. Firstly, they discussed types of phishing including spear-phishing and pharming. They also explored ways in which malicious links are spread such as through shortened links, spoofing and all kinds of squatting. In [13], the authors used online learning to detect unknown or zero hour phishing URLs. Precisely, they trained a confidence weighted algorithm using surface level URL features. PhishDef is a phishing URL detection technique that was proposed by authors in [14]. They used URL lexical features and showed that these features were sufficient for detection. They also compared several classification algorithms then evaluated the performance of batch-based and online algorithms. PhishDef could detect phishing URLs on-the-fly with a 0.97 accuracy. [15] focused their work on identifying relevant features that distinguish between legitimate and phishing websites. Using associate rule mining and predictive apriori, they were able to determine the effectiveness of their selected URL features. Important features in their work included keywords, top level domains and transport layer security. A novel concept of oversampling in URL data space was introduced by [16]. They generated synthetic URLs using text generative adversarial networks on a dataset collected from phishtank.com. They used a 80-20 split ratio for their train and test sets. Their work also relied on features extracted directly from the URL string. In their study, they used k-means clustering and Euclidean distance to generate synthetic URLs.

In [5], the authors used Kolmogorov-Smirnov test on four different real datasets. Their method presented novelty through their statistical measures. They recorded high accuracy while using automatic feature extraction of URL character distribution and a few other features. Conducting their experiment in WEKA, the authors examined Random Forest, J48, PART, SMO, Logistic Regression and used Naive Bayes as a baseline for comparison. A fast feature extraction system based on URL analysis was proposed in [9]. Their work targeted obfuscation techniques used by cyberattackers when constructing phishing URLs. Using character n-grams as their features and online learning algorithms, they investigated their work on balanced and imbalanced datasets. They also made comparisons between Phishing and Spam domains. They concluded that phishing URL analysis is significantly different from spam detection through domain analysis. Their work motivated us to take a step further and investigate URL multi-class detection, experimenting on domains as well as whole URLs from spam, malware and phishing.

B. Spam URL Detection

Unlike malware and phishing, spam is usually intended for commercial advertising to a large number of recipients. However, some spam can also be used for fraudulent activities like selling counterfeit products, illegal drugs or even fake job opportunities. The work done by [1] used the Support Vector Machine (SVM) classifier to classify links as spam or non spam. They used open source datasets from WEBSPAM-UK2006 and WEBSPAM-UK2007. KL divergence was also used by the authors to measure the differences between two text units of the source and target pages. [17] also used the same dataset as [1] to identify web spam through the analysis of language models. Using Natural Language Processing (NLP) and KL divergence, they identified the differences between two web pages. They obtained anchor text, surrounding anchor text and URL terms from the source pages as well as title, content page and meta tags from the target page. Their

work reported an improvement of near 0.06 and 0.02 for F-measure on the two datasets respectively.

A real-time URL spam filtering service (Monarch) was proposed by the authors in [18]. 11 million URLs drawn from email and Twitter were used in their work. They found that there was a difference between spam targeting email and that targeting Twitter. The authors demonstrated that Monarch could achieve a traffic of 638,000 URLs per day with a 0.91 overall accuracy and 0.87 false positives. They estimated that it would cost $22 751 a month to deploy Monarch with the potential of processing 15 million URLs per day. Researchers in [19] introduced some measurements to enable the detection of spam by analysing the misuse of short URLs. Using one of the popular URL shortening services Bitly, they investigated on 641,423 URLs with meta-data such as info, clicks, countries and referrers. They also investigated patterns in click traffic of spam short URLs. They conducted their experiments in the WEKA software package. Of all the classification algorithms they used, Random Tree achieved the best performance with 0.9081 accuracy. BEAN, a behavior analysis approach was also proposed by [20]. Deriving their approach from a Markov chain, they achieved a 0.91 precision and 0.88 recall. Their work analyzed URL spam in Twitter trending topics.

In [21], the authors proposed a new approach for detecting spam URLs. Their approach leveraged URL redirection properties. The novelty of their work lies in three characteristics: domain-independence, adversarial robustness and semi-supervised detection. They used both supervised and semi-supervised detection, SVM, Logistic Regression (LR) and Decision Tree(DT). Their semi-supervised method achieved over 0.96 recall, 0.70 precision and a false positive rate below 0.07. [22] investigated spam URL behavioral signals. They used the Bitly click API in their spam detection process and obtained an accuracy of 0.86. Their work emphasized using only the behavioral signals. Crowdsourcing for URL spam detection (CUD), that can be used as a supplement of existing detection tools was introduced by [23]. They employed sentiment analysis based on the human submitted comments about the URLs. Supervised machine learning, specifically the Repeated Incremental Pruning to Produce Error Reduction algorithm was used in their work. They obtained an 0.868 true positive rate and a 0.9 false positive rate from their experiments.

C. Malware URL Detection

Malware URLs spread malicious software that can potentially damage computer systems. In [24], the authors addressed blacklist detection challenges by training a SVM classifier using discriminative lexical features of malware URLs. Their real-time anomaly based classification involved a manual identification and evaluation of malware URLs from existing blacklist sources. They identified 12 lexical features and conducted an empirical analysis to verify the effectiveness of the features. They found that there was a consistency in the way malware URLs are crafted. A 0.9695 accuracy was recorded in their work. A presentation by [25] also did an empirical analysis of the factors affecting malware URL detection with the hypothesis that characteristics of malware URLs or blacklisting techniques influence detection performance. They collected data samples from publicly available malware domain lists and scanned them through Virus total that searches through several databases for URL statuses. LR and Cox proportional hazard models were used to verify their hypothesis. Researchers in [26] used a semi-supervised approach to classify benign and malware URLs. They trained an algorithm based on results they had collected over a period of 1000 hours. Their model served as a network-based (URL Blocker) filter to aid detection in a real-time stream of URLs. 123 boolean features were extracted from URL components (schema, host name, primary domain, top level domain, path and query) as well as from external services. In [27] MineSpider, a browser emulator that uses a URL extractor targeting drive-by download attacks was proposed. Their objective was to design a model that extracts rather than detects malicious URLs. They extracted code relevant to redirections and analyzed JavaScript containing browser fingerprinting code. Their method extracted 30 000 new URLs in a few seconds as compared to conventional techniques. It also reduced the number of false negatives of malicious websites.

There is also extensive work done in binary classification detection techniques whereby researchers did not discriminate between the attack types. In most cases, they combined all the malicious URLs into one set and trained algorithms that classify whether a URL is malicious or benign. In [11], majority voting and DT were used to classify malicious and benign URLs. They used a combination of 117 dynamic and static features, 44 of them were novel. A balanced dataset of 26 041 benign and malicious URLs was used to evaluate 6 decision tree classifiers. Their features included URL features, domain name features, web page source features and short URL features. An approach that used static lexical features extracted from the URL string was used in [28]. They proposed this approach based on the assumption that the distribution of URL features is different for malicious and benign URLs. Their work confirmed that malicious URLs are significantly different from benign URLs in their lexical distribution. Researchers in [29] conducted a binary classification experiment using lexical features, network and host features. RF and SVM returned the highest accuracy in their work. 80:20 was reported to be the best split ratio. Binary classification was also explored by [30], [31], [32] and [33]. Some researchers explored short URL detection. The popularity of short URLs especially on Online Social Networks (OSN) has resulted in the spread of more malicious URLs. Work done by [34] investigated whether URL shortening services are an effective and reliable tool that can be used to hide malicious URLs. They did so by analysing existing detection techniques used by well-known shortening services. Their work observed 622 URL shortening services and collected 24 953 881 short URLs over a period of 2 months. [35] used URL shorteners to compare phishing and malware attacks. Over 7 000 malicious Bitly short URLs la-

beled as malware and phishing were collected. Through phish click analytics, they found most phishing clicks to be from the USA and Russia. From their findings, detection techniques against phishing attacks were stronger than efforts against malware attacks. Short URL detection was also investigated in [36], [37] and [38].

IV. Data Collection

Data collection and representation is crucial in building an effective detection model. We collected benign, malware, phishing and spam URLs from publicly available sources. Most of these sources have also been used by recent URL detection studies [9], [39], [40], [32] and [41]. Below, we discuss the datasets collected and their properties. Table I shows our datasets distribution. A total of 126 983 URLs was used in our work.

DMOZ Dataset: We collected 58 132 benign URLs from the parsed DMOZ[3] dataset from the Harvard dataverse website. Benign URLs are considered legitimate which pose no risk for the internet users when clicked. DMOZ used to be an open human edited directory. Although it ceased its operations, a lot of researchers still rely on crawling URLs from this site.

PhishTank Dataset:14 374 phishing URLs were collected from PhishTank[4]. PhishTank is a benchmark community based anti-phishing site with an open API that facilitates access to anti-phishing data. It is one of the diverse sources for Phishing URLs.

URLhaus Dataset: We obtained a total of 31 851 malware URLs from project abuse.ch[5] that facilitates different projects towards malware detection. URLhaus provides plain-text malware URL lists that are generated every 5 minutes. It aims to share malicious URLs that are being manipulated to spread malware.

WEBSPAM Dataset: We collected 22 626 spam URLs from publicly available datasets (webspam-UK2006 and webspam-UK2006). This site provides spam and non-spam URLs. All URLs from the mentioned datasets are manually checked for incorrect syntax. We also delete all duplicated URLs from our collection. The labels Benign:0, Malware:1, Spam:2 and Phishing:3 are assigned to the URLs.

A. CLASSIFICATION TECHNIQUES

We use the Scikit-Learn python library on the free cloud service Google Colab. Scikit-Learn is a free software that has a variety of well-established machine learning algorithms. Google Colab, colab meaning colaboratory allows writing and execution of python code through the browser. The classifiers used in the classification framework include XGBoost, AdaBoost, LightGBM and CatBoost. These boosting ensembles were initially used for binary classification. However, over time they advanced to multiclass classification tasks.

TABLE I
Dataset for Training and Testing

URL Class	Test-Set	Train-Set	Total
Benign	11 627	46 505	58 132
Phishing	2 875	11 499	14 374
Malware	6 371	25 480	31 851
Spam	4 526	18 100	22 626
Total	25 399	101 584	126 983

[3]https://dataverse.harvard.edu/dataset
[4]https://www.phishtank.com/
[5]https://abuse.ch/

B. XGBoost

The XGBoost algorithm has been praised for winning Kaggle and other data challenge competitions. It provides a gradient boosting framework designed for better computational speed and performance [42]. By implementing the boosting technique, it has the ability to handle missing values and perform better regularization. XGBoost is widely used due to its loss minimization abilities.

C. AdaBoost

This ensemble algorithm is mostly used to improve decision trees on binary classification tasks. AdaBoost also known as meta-learning was developed by [43]. It trains a series of classifiers based on the performance of previous classifiers in the series. The learners are trained based on the results obtained from a weak hypothesis and reweighing each instance according to the output. In a multiclass setting, AdaBoost uses the Stagewise Additive Modeling (SAMME) algorithm that adaptively implements the multi-class Bayes rule by fitting a SAMME model for the task [44]. It also uses the Stagewise Additive Modeling for Real-valued probabilities obtained from weak classifiers (SAMME.R). These algorithms contribute towards the increase in prediction accuracy.

D. LightGBM

LightGBM introduced by Microsoft is a tree based gradient boosting framework. It is considered fast at processing and powerful at computation. LightGBM is however prone to overfitting especially on small data samples. It differs from other boosting techniques in that it splits the decision trees with the best fit at the leaf whereas others split at depth or level. This allows it to reduce more losses than other algorithms. LightGBM serves as an improvement to gradient boosting through efficiency and scalability [45].

E. CatBoost

Developed by Yandex researchers, this boosting algorithm is for gradient boosting on decision trees. [46] introduced the important algorithmic techniques behind CatBoost. It outperforms most boosting algorithms in terms of prediction speed. It also handles categorical features better than most algorithms [47]. The tree splitting structure of CatBoost results in balanced predictors and reduced overfitting.

The above mentioned boosting algorithms all have their own pros and cons. Most of the algorithms have been used in malicious URL binary classification tasks. However, we explore their performance in a multi-class URL classification setting. Boosting techniques prove to be an effective machine learning algorithm strategy.

V. Experimental Results and Evaluation

A. Feature Selection

Feature extraction, representation and application play a vital role in URL classification. Table II lists the features used in our approach. We analyzed the characteristics of the original URL strings. Our goal is to improve detection speed, accuracy and reliability by focusing on URL lexical features. Lexical features are the textual properties of URL components. They can be extracted from the URLs protocol, domain name, host name, top level domain name, path and parameters. The commonly used URL lexical features from existing literature are word-based features, special character features and count-based features [32], [29], [16], [48] and [9]. We follow the same logic and extract lexical properties from the URL samples. In addition, we check for correlated features before we train our models. When working with URL textual properties, it is very common for some of the features to be highly correlated. Highly correlated features have a negative impact on classification results. The following is a summary of the features used in our work:

Word Based Features Word-based features include suspicious words, security sensitive words, suggestive words and obfuscated words. Suspicious words refer to questionable words that are commonly found in URL strings. We combine all these words into one feature that is suspicious words. These include words like *login, online, mail, verify, index, ad-min, security, submit, subscription, loan, student, admission, secure, signin, transaction, e-transfer, update, shopping, mail.php, cinfig, link*. Cyber attackers use word-based feature tactics to mislead unsuspecting users. Other word-based features are found through obfuscation and typosquatting. Some of the ways in which attackers manipulate URLs are shown in Fig.1. To collect these word-based features, we turn to the word segmentation technique. First, the URL string is split and then we segment the URL tokens to extract segmented bag-of-words [49]. Our suspicious words list also contains brand name features that we manually collect from the URLs. Cyberattackers use famous brands to mask their malicious intentions. We check for the presence of these brand names in the domain names and URL path. We further query our datasets for similar looking brand names that mimic well-known brands. These include:*paypell, albnb, paypaal, payapl, instagraam, pavpal, paypel, pay-pal, pay-pal, amavczon, amaczion, facccceeebookk, amaczon, amaevzon, amavezon, michrosoft, xcel, e3bay, paypl, neetflixin, payral, paypall, faceyook*.

Special Character Features Characters in a URL play an important role in differentiating benign from malicious URLs. We use 17 URL and domain name special character based features. These characters include the following special characters{%$_–#&\+_![]/:?*.,@}. Some malicious URLs contain common punctuation and mathematical symbols. Attackers mislead innocent users by adding suffix or prefix separated by a - sign. The "@" symbol is also a commonly used technique to trick users into clicking malicious URLs. Internet users tend to pay less attention to the content in a URL string that comes before this sign. They just focus on the domain that they are familiar with which comes after the @ sign.

Shannon Entropy Features We use Shannon entropy to demonstrate the randomness factor in the URL strings [11]. Malicious URLs usually have a higher randomness factor as compared to benign URLs. We calculate URL, domain name, path and query entropy.(1) denotes the formula for Shannon entropy.

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

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. Malicious URLs report a higher entropy as compared to benign URLs confirming the fact that there is more randomness in malicious URLs as compared to benign URLs.

KL divergence Features The distribution of English alphabetical letters in a URL can indicate its maliciousness or legitimacy. Domain- generation algorithms (DGAs) influenced us to use KL divergence also known as relative entropy to measure the similarity between the English character distributions in malicious URLs and benign URLs. Human constructed URLs normally contain sensible words and a reasonable amount of characters according to the day-to-day English vocabulary. Digitally generated URLs are very random and in most cases are a chain of meaningless strings of characters. Therefore, we use KL divergence to calculate the character distribution between the URLs. We collect the KL divergence for the whole URL string, domain names, path, query as well as a combination of both path and query.

KL divergence is calculated using (2).

$$D_{KL}(P \,||\, Q) = \sum_{i^P i} \log(\frac{pi}{qi}) \tag{2}$$

where, D is the KL-Divergence value, P represents the distribution of characters in standard English, and Q represents the distribution of characters in the URL.

TABLE II
URL BASED FEATURES USED IN OUR WORK

Other Features	Our Features
URL length	URL length
Domain name length	Domain name length
URL Shannon entropy	URL KL divergence
Special characters count	Special characters count
Presence of an IP address	Presence of an IP address
Domain Shannon entropy	Domain KL divergence
Path Shannon entropy	Path KL divergence
Query Shannon entropy	Query KL divergence
Query + Path Shannon entropy	Query + Path KL divergence
Security sensitive words	Sensitive brand-name words
Bag of Words features	Segmented bag of words
Number of signs	Number of signs
Presence of .exe file extension in the URL	Presence of suspicious file extensions
Short URL length	Sub domains count
Digit to letter ratio	Frequency of digits
Short URL features	Short URL features
Presence of (%20)	Top level domains count

Our file extension features include checking the URL string for the presence of the following file extensions *·exe, ·scr, ·vbs, ·js, .xml, .docm, ·xps, .iso, .img, ·doc, .rtf, ·xls, ·pdf, .pub, .arj, .lzh, .r01, .r14, .r18, .r25, .tar, .ace, .zip, .jar, .bat, .cmd, .moz, .vb, .vbs, .js, .wsc, .wsh, .ps1, .ps1xml, .ps2, .ps2xml, .psc1 and .psc2*.

B. Evaluation Metrics

We evaluated the performance of multiclass ensemble boosting techniques on our URL dataset shown in Table I. To evaluate the models, we use the commonly used metrics also used by [11] which are :*accuracy, average accuracy, error rate, micro-precision, micro-recall, micro-fscore, macro-precision, macro-recall and macro-fscore*.

$$AverageAccuracy = \frac{\sum_{i=1}^{l} \frac{tpi+tni}{tpi+fni+fpi+tni}}{l} \tag{3}$$

Average accuracy reflects the average effectiveness of a classifier for each class (3).

$$ErrorRate = \frac{\sum_{i=1}^{l} \frac{fpi+fni}{tpi+fni+fpi+tni}}{l} \tag{4}$$

The error rate shows classification error for each class(4).

$$Micro - Precision = \frac{\sum_{i=1}^{l} tpi}{\sum_{i=1}^{l}(tpi + fpi)} \tag{5}$$

It measures the sum of contributions of all classes that are truly positive based on their precision (5).

$$Micro - Recall = \frac{\sum_{i=1}^{l} tpi}{\sum_{i=1}^{l}(tpi + fni)} \tag{6}$$

The sum of the models predictions on all true positive samples actually predicted as the positive class (6).

$$Macro - Precision = \frac{\sum_{i=1}^{l} \frac{tpi}{tpi+fpi}}{l} \tag{7}$$

It is the measure of the average precision per class. Precision reflects all samples classified as positive being positive (7).

$$Macro - Recall = \frac{\sum_{i=1}^{l} \frac{tpi}{tpi+fni}}{l} \tag{8}$$

This measures the average per-class effectiveness of a class when identifying class labels (8).

$$Micro-Fscore = \frac{(\beta^2+1)MicroPrecision.MicroRecall}{\beta^2 MicroPrecision + MicroRecall} \tag{9}$$

It is used to measure the quality of the classifier based on the fscore of contributions from all classes(9).

$$Macro - Fscore \frac{(\beta^2+1)MacroPrecision.MacroRecall}{\beta^2 MacroPrecision + MacroRecall} \tag{10}$$

This is the relation between actual positive labels ad those returned by the classifier per class (10).

C. Performance Evaluation

Table V displays the detection accuracy per class types obtained from the ensemble learners. The AdaBoost classifier outperformed all the other three learners. In general, all classification tasks returned reasonably high accuracy rates. XGBoost, CatBoost and AdaBoost showed improved accuracy rates when using our features. The overall detection accuracy for benign URLs is over 0.980 for all classifiers with and without our features. This validates the fact that benign URLs are easier to learn and detect as their structure has not been tampered with or altered for malicious activity. They usually contain meaningful words and proper character distributions. We saw a significant improvement on Spam detection. XGBoost, CatBoost and AdaBoost return a 0.0741, 0.0566 and 0.0035 increase respectively. LightGBM however dropped from 0.7736 to 0.7667, a 0.0069 drop in detection accuracy.

Our spam detection accuracy outperformed that of the work done by [11]. Although our feature sets and algorithms are different, we still obtained a higher accuracy for the spam URLs. Their work obtained as low as 0.4202 detection accuracy for spam while using URL source features, domain name features and short URL features. Phishing URLs returned over 0.990 detection accuracy. This may be because we used bag-of-words segmentation and character level KL divergence as part of our features. Phishing URLs usually pose as dodgy, mismatching URLs that may have additional random characters or incorrectly spelled look-alike brand names. By extracting features from the whole URL string, we were able to detect the abnormally long phishing URLs, presence of digits in the URL, additional prefix and suffix thus improving the detection accuracy.

Table III and IV show the precision, recall and F-score obtained from a binary classification setting. In a binary setting, we use precision, recall and F-score for evaluation. Precision refers to the ratio of positive predictions that are actually positive, recall shows the ratio of actual positive classes that are correctly classified as positive and lastly F-score which is a combination measure of both recall and precision.

Here, XGBoost achieved an increase in all of the above mentioned metrics when using our features as compared to without our features. The highest improvement of 0.0017 obtained from recall is seen on the Spam URLs. AdaBoost and LightGBM registered significant improvements on the Phishing URLs. They returned over 0.9800 in all metrics. Overall, we obtained increased and balanced values using our features which indicates unbiased detection on our datasets. A high recall from all our classifiers also indicates that the classifiers were able to classify most malicious URLs as malicious instead of misclassifying them as benign which would be detrimental.

The class-specific error analysis for the four classifiers is shown in TableVI and VII. Error analysis helps us identify the extent of overlaps between the URL types. There is a huge

TABLE III
CLASSIFICATION RESULTS WITHOUT OUR FEATURES

Classifier	Precision	Recall	F-Score
XGBoost Classifier			
Benign	0.9429	0.9735	0.9660
Malware	0.9533	0.9457	0.9408
Spam	0.8903	0.7750	0.8211
Phishing	0.9845	0.9820	0.9800
CatBoost Classifier			
Benign	0.9400	0.9818	0.9615
Malware	0.9438	0.9490	0.9427
Spam	0.8732	0.7406	0.8116
Phishing	0.9754	0.9850	0.9728
AdaBoost Classifier			
Benign	0.9660	0.9740	0.9660
Malware	0.9510	0.9600	0.9600
Spam	0.9139	0.8444	0.8721
Phishing	0.9764	0.9900	0.9770
LightGBM Classifier			
Benign	0.9533	0.9816	0.9743
Malware	0.9440	0.9419	0.9417
Spam	0.8709	0.7734	0.8237
Phishing	0.9800	0.9820	0.9756

TABLE IV
CLASSIFICATION RESULTS WITH OUR FEATURES

Classifier	Precision	Recall	F-Score
XGBoost Classifier			
Benign	0.9686	0.9820	0.9714
Malware	0.9610	0.9521	0.9555
Spam	0.9440	0.9502	0.9616
Phishing	0.9912	0.9904	0.9945
CatBoost Classifier			
Benign	0.9607	0.9822	0.9727
Malware	0.9523	0.9556	0.9524
Spam	0.8929	0.8109	0.8532
Phishing	0.9987	0.9905	0.9974
AdaBoost Classifier			
Benign	0.9688	0.9904	0.9792
Malware	0.9676	0.9684	0.9655
Spam	0.9198	0.8489	0.8799
Phishing	0.9899	0.9980	0.9990
LightGBM Classifier			
Benign	0.9567	0.9803	0.9711
Malware	0.9459	0.9440	0.9472
Spam	0.8776	0.7760	0.8251
Phishing	0.9900	0.9900	0.9900

TABLE V
DETECTION ACCURACY IN FOR EACH CLASS TYPE

URL type	Benign	Malware	Spam	Phishing
Without Our Features				
XGBoost	0.9826	0.9408	0.7660	0.9870
CatBoost	0.9826	0.9393	0.7534	0.9837
AdaBoost	0.9850	0.9556	0.8387	0.9913
LightGBM	0.9818	0.9408	0.7736	0.9898
With Our Features				
XGBoost	0.9836	0.9510	0.8401	0.9928
CatBoost	0.9836	0.9473	0.8100	0.9909
AdaBoost	0.9855	0.9521	0.8422	0.9922
LightGBM	0.9809	0.9355	0.7667	0.9891

overlap between benign and spam URLs using XGBoost without and with our features. 0.1427 spam URLs were classified as benign. This reduced to 0.0934 when using our features. In comparison, 0.0085 benign URLs were misclassified as spam, it is however an error that can be accommodated since it is better to misclassify benign URLs as malicious than the other way round. There is also a noticeable overlap between spam and malware URLs. 0.0864 and 0.0658 spam URLs were classified as malware without and with our features respectively. This is probably due to some similarities between the features used in our analysis. Spam URLs misclassified as phishing were all below 0.005 with our features. The largest error was 0.0031 obtained from LightGBM. Such a low error rate shows that our features were effective in distinguishing between spam and phishing URLs.

0.1507 and 0.1105 error rates were obtained for CatBoost with and without our features for Spam URLs misclassified as benign respectively. AdaBoost returned a 0.1056 and 0.0944 error rate. Lastly, LightGBM obtained 0.1325 and 0.1297. There was a significant decrease in the error rates for all classifiers pertaining spam URLs misclassified as benign URLs.

Malware URLs misclassified as phishing URLs with and without our features for all four classifiers were as follows:XGBoost 0.0014 and 0.0003, CatBoost 0.0013 and 0.0005, AdaBoost 0.0003 and 0.0002, lastly LightGBM 0.0014 and 0.0011. All classifiers showed an improvement using our URL based features. Malware URLs misclassified as benign also returned a decrease in error rate as compared to that of spam URLs misclassified as benign when using our features. Table IX presents the overall contributions of our features using all four classifiers. LightGBM returned the lowest contribution of 0.0025 while AdaBoost returned the highest contribution of 0.0147. Catboost and XGBoost demonstrated an overall contribution of 0.0141 and 0.0137 respectively.

TABLE VI
CLASS-SPECIFIC ERROR ANALYSIS FOR MULTICLASS CLASSIFIERS WITHOUT OUR FEATURES USING OUR DATASETS

URL type	Benign	Malware	Spam	Phishing
XGBoost Classifier				
Benign	0.9826	0.0081	0.0089	0.0003
Malware	0.0328	0.9408	0.0249	0.0014
Spam	0.1427	0.0864	0.7660	0.0049
Phishing	0.0106	0.0006	0.0017	0.9870
CatBoost Classifier				
Benign	0.9826	0.0078	0.0085	0.0010
Malware	0.0306	0.9393	0.0289	0.0013
Spam	0.1507	0.0916	0.7534	0.0042
Phishing	0.0130	0.0022	0.0011	0.9837
AdaBoost Classifier				
Benign	0.9850	0.0059	0.0089	0.0001
Malware	0.0216	0.9556	0.0224	0.0003
Spam	0.1056	0.0546	0.8387	0.0010
Phishing	0.0073	0.0004	0.0009	0.9913
LightGBM Classifier				
Benign	0.9818	0.0077	0.0095	0.0009
Malware	0.0263	0.9408	0.0314	0.0014
Spam	0.1325	0.0913	0.7736	0.0024
Phishing	0.0078	0.0006	0.0017	0.9898

TABLE VII
CLASS-SPECIFIC ERROR ANALYSIS FOR MULTICLASS CLASSIFIERS WITH OUR FEATURES USING OUR DATASETS

URL type	Benign	Malware	Spam	Phishing
XGBoost Classifier				
Benign	0.9836	0.0074	0.0085	0.0005
Malware	0.0223	0.9510	0.0263	0.0003
Spam	0.0934	0.0658	0.8401	0.0007
Phishing	0.0052	0.0002	0.0017	0.9928
CatBoost Classifier				
Benign	0.9836	0.0068	0.0088	0.0007
Malware	0.0256	0.9473	0.0267	0.0005
Spam	0.1105	0.0783	0.8100	0.0010
Phishing	0.0073	0.0004	0.0013	0.9909
AdaBoost Classifier				
Benign	0.9855	0.0066	0.0076	0.0002
Malware	0.0241	0.9521	0.0235	0.0002
Spam	0.0944	0.0626	0.8422	0.0007
Phishing	0.0063	0.0002	0.0013	0.9922
LightGBM Classifier				
Benign	0.9809	0.0094	0.0089	0.0008
Malware	0.0314	0.9355	0.0320	0.0011
Spam	0.1297	0.0100	0.7667	0.0031
Phishing	0.0080	0.0006	0.0022	0.9891

We saw an overlap between malware and spam URLs with the highest overlap seen on the LightGBM classifier.The URL features for these two attack types might be highly correlated resulting in misclassification.The error rate using our features increased by 0.0006. As shown in Table VIII, all classifiers achieved high micro prec(precision), rec(recall) and F-score. XGBoost micro recall and precision showed a 0.0017 increase using our features.

TABLE VIII
MULTI-CLASS CLASSIFICATION MICRO AND MACRO WITH AND WITHOUT OUR FEATURES

Model	Micro Prec	Micro Rec	Micro Fscore	Macro Prec	Macro Rec	Macro Fscore
Without Our Features						
XGBoost	0.9533	0.9533	0.9533	0.9527	0.9462	0.9506
CatBoost	0.9517	0.9520	0.9517	0.9544	0.9312	0.9424
AdaBoost	0.9654	0.9654	0.9655	0.9612	0.9439	0.9502
LightGBM	0.9534	0.9534	0.9534	0.9456	0.9201	0.9315
With Our Features						
XGBoost	0.9697	0.9697	0.9697	0.9458	0.9295	0.9378
CatBoost	0.9683	0.9683	0.9683	0.9447	0.9132	0.9383
AdaBoost	0.9681	0.9681	0.9681	0.9667	0.9405	0.9511
LightGBM	0.9577	0.9574	0.9577	0.9438	0.9237	0.9321

TABLE IX
OVERALL CONTRIBUTION OF OUR FEATURES ON THE AVERAGE CLASSIFICATION ACCURACY

Multi-Class Classifier	Without Our Features	With Our Features	Change in (ratio)
XGBoost	0.9546	0.9683	0.0137
CatBoost	0.9550	0.9691	0.0141
AdaBoost	0.9551	0.9698	0.0147
LightGBM	0.9542	0.9567	0.0025

VI. CONCLUSION AND FUTURE WORK

In this work, we aimed to find the best performing model using our URL based features in a multiclass ensemble classification setting. We have proposed an approach to detect malicious URLs and tested our approach using four datasets containing different URLs. We extracted URL based features such as word-based features, special character features, KL divergence features, bag of words segmentation features, domain name features and short URL features. We have evaluated the performance of four ensemble machine learning classifiers and reported their average accuracy, micro and macro precision, micro and macro recall and F1-Score. We tested the models with and without our features. Malicious URL detection requires an almost instant response considering the ever-changing attack techniques being devised by cyber criminals. In all experiments, Spam detection obtained the lowest scores as compared to other datasets although the score is high as compared to other previous work. In the future, we will concentrate on multiclass classification using other algorithms such as online learners. Our work also follows suggestions offered by previous authors in this domain for example exploring XGBoost when detecting malicious URLs. Overall XGBoost and AdaBoost perform exceptionally well when detecting URL types. We would like to further explore other ensemble learners, deep learning and online learners in our future work and how they will perform when using URL based features. Also explore neural networks as done by [50] and compare performances. There is need to extract more discriminative features that can assist in differentiating spam, malware and phishing URLs from benign ones.

REFERENCES

[1] R. C. Patil and D. R. Patil, "Web spam detection using svm classifier," in *2015 IEEE 9th International Conference on Intelligent Systems and Control (ISCO)*, 2015, pp. 1–4.

[2] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "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*, 2009, pp. 1245–1254.

[3] C. Zhao, Y. Xin, X. Li, Y. Yang, and Y. Chen, "A heterogeneous ensemble learning framework for spam detection in social networks with imbalanced data," *Applied Sciences*, vol. 10, no. 3, p. 936, 2020.

[4] J. Xu, S. Denman, C. Fookes, and S. Sridharan, "Detecting rare events using kullback-leibler divergence: A weakly supervised approach," *Science & Engineering Faculty*, 2016.

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

[6] L. Araujo and J. Martinez-Romo, "Web spam detection: New classification features based on qualified link analysis and language models," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 3, pp. 581–590, 2010.

[7] J. Hong, T. Kim, J. Liu, N. Park, and S.-W. Kim, "Phishing url detection with lexical features and blacklisted domains," *Adaptive Autonomous Secure Cyber Systems*, pp. 253–267, 2020.

[8] X. Zheng, J. Wang, F. Jie, and L. Li, "Two phase based spammer detection in weibo," in *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, 2015, pp. 932–939.

[9] R. Verma and A. Das, "What's in a url: Fast feature extraction and malicious url detection," in *Proceedings of the 3rd ACM on International Workshop on Security And Privacy Analytics*, 2017, pp. 55–63.

[10] M. S. I. Mamun, M. A. Rathore, A. H. Lashkari, N. Stakhanova, and A. A. Ghorbani, "Detecting malicious urls using lexical analysis," in *International Conference on Network and System Security*. Springer, 2016, pp. 467–482.

[11] D. R. Patil and J. B. Patil, "Malicious urls detection using decision tree classifiers and majority voting technique," *Cybernetics and Information Technologies*, vol. 18, no. 1, pp. 11–29, 2018.

[12] O. Christou, N. Pitropakis, P. Papadopoulos, S. McKeown, and W. J. Buchanan, "Phishing url detection through top-level domain analysis: A descriptive approach." in *Proceedings of the 6th International Conference on Information Systems Security and Privacy*, 2020, pp. 289–298.

[13] A. Blum, B. Wardman, T. Solorio, and G. Warner, "Lexical feature based phishing url detection using online learning," in *Proceedings of the 3rd ACM workshop on Artificial intelligence and security*, 2010, pp. 54–60.

[14] A. Le, A. Markopoulou, and M. Faloutsos, "Phishdef: Url names say it all," in *2011 Proceedings IEEE INFOCOM*, 2011, pp. 191–195.

[15] S. C. Jeeva and E. B. Rajsingh, "Intelligent phishing url detection using association rule mining," *Human-centric Computing and Information Sciences*, vol. 6, no. 1, p. 10, 2016.

[16] A. Anand, K. Gorde, J. R. A. Moniz, N. Park, T. Chakraborty, and B.-T. Chu, "Phishing url detection with oversampling based on text generative adversarial networks," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 1168–1177.

[17] J. Martinez-Romo and L. Araujo, "Web spam identification through language model analysis," in *Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, 2009, pp. 21–28.

[18] K. Thomas, C. Grier, J. Ma, V. Paxson, and D. Song, "Design and evaluation of a real-time url spam filtering service," in *2011 IEEE Symposium on Security and Privacy*, 2011, pp. 447–462.

[19] D. Wang, S. B. Navathe, L. Liu, D. Irani, A. Tamersoy, and C. Pu, "Click traffic analysis of short url spam on twitter," in *9th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing*, 2013, pp. 250–259.

[20] D. Wang and C. Pu, "Bean: A behavior analysis approach of url spam filtering in twitter," in *2015 IEEE International Conference on Information Reuse and Integration*, 2015, pp. 403–410.

[21] H. Kwon, M. B. Baig, and L. Akoglu, "A domain-agnostic approach to spam-url detection via redirects," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2017, pp. 220–232.

[22] C. Cao and J. Caverlee, "Behavioral detection of spam url sharing: posting patterns versus click patterns," in *Proceedings of the 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2014, pp. 138–141.

[23] J. Hu, H. Gao, Z. Li, and Y. Chen, "Poster: Cud: crowdsourcing for url spam detection," in *Proceedings of the 18th ACM conference on Computer and communications security*, 2011, pp. 785–788.

[24] O. Morufu, A. M. Taufik, M. Ramlan, and A. Azizol, "Identification and evaluation of discriminative lexical features of malware url for real-time classification," *IEEE Conference Proceedings*, vol. 2016, p. 95, 2016.

[25] M. Vasek and T. Moore, "Empirical analysis of factors affecting malware url detection," in *2013 APWG eCrime Researchers Summit*, 2013, pp. 1–8.

[26] A. D. Gabriel, D. T. Gavrilut, B. I. Alexandru, and P. A. Stefan, "Detecting malicious urls: A semi-supervised machine learning system approach," *2016 18th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, pp. 233–239, 2016.

[27] Y. Takata, M. Akiyama, T. Yagi, T. Hariu, and S. Goto, "Minespider: Extracting urls from environment-dependent drive-by download attacks," in *2015 IEEE 39th Annual Computer Software and Applications Conference*, vol. 2, 2015, pp. 444–449.

[28] A. Joshi, L. Lloyd, P. Westin, and S. Seethapathy, "Using lexical features for malicious url detection – a machine learning approach," *arXiv preprint arXiv:1910.06277*, 2019.

[29] R. Patgiri, H. Katari, R. Kumar, and D. Sharma, "Empirical study on malicious url detection using machine learning," in *International Conference on Distributed Computing and Internet Technology*, 2019, pp. 380–388.

[30] F. Vanhoenshoven, G. Napoles, R. Falcon, K. Vanhoof, and M. Koppen, "Detecting malicious urls using machine learning techniques," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2016, pp. 1–8.

[31] A. Astorino, A. Chiarello, M. Gaudioso, and A. Piccolo, "Malicious url detection via spherical classification," *Neural Computing and Applications*, vol. 28, no. 1, pp. 699–705, 2017.

[32] F. Khan, J. Ahamed, S. Kadry, and L. K. Ramasamy, "Detecting malicious urls using binary classification through adaboost algorithm," *International Journal of Electrical and Computer Engineering*, vol. 10, no. 1, pp. 997–1005, 2020.

[33] V. N and Vinodhini, "Malicious-url detection using logistic regression technique," *International Journal of Engineering and Management Research*, vol. 9, no. 6, pp. 108–113, 2019.

[34] F. Maggi, A. Frossi, S. Zanero, G. Stringhini, B. Stone-Gross, C. Kruegel, and G. Vigna, "Two years of short urls internet measurement: security threats and countermeasures," in *Proceedings of the 22nd international conference on World Wide Web*, 2013, pp. 861–872.

[35] S. L. Page, G.-V. Jourdan, G. V. Bochmann, J. Flood, and I.-V. Onut, "Using url shorteners to compare phishing and malware attacks," in *2018 APWG Symposium on Electronic Crime Research (eCrime)*, 2018, pp. 1–13.

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

[37] Y. Alshboul, R. K. Nepali, and Y. Wang, "Detecting malicious short urls on twitter," in *AMCIS*, 2015.

[38] B. Alghamdi, J. Watson, and Y. Xu, "Toward detecting malicious links in online social networks through user behavior," in *2016 IEEE/WIC/ACM International Conference on Web Intelligence Workshops (WIW)*, 2016, pp. 5–8.

[39] A. E. Aassal, S. Baki, A. Das, and R. M. Verma, "An in-depth benchmarking and evaluation of phishing detection research for security needs," *IEEE Access*, vol. 8, pp. 22 170–22 192, 2020.

[40] B. Sabir, M. A. Babar, and R. Gaire, "An evasion attack against ml-based phishing url detectors." *arXiv preprint arXiv:2005.08454*, 2020.

[41] T. Li, G. Kou, and Y. Peng, "Improving malicious urls detection via feature engineering: Linear and nonlinear space transformation methods," *Information Systems*, vol. 91, p. 101494, 2020.

[42] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.

[43] Y. Freund and R. E. Schapire, "A short introduction to boosting," *AT and T*, 1999.

[44] T. Hastie, S. Rosset, J. Zhu, and H. Zou, "Multi-class adaboost," *Statistics and Its Interface*, vol. 2, no. 3, pp. 349–360, 2009.

[45] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: a highly efficient gradient boosting decision tree," in *NIPS'17 Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 3149–3157.

[46] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," in *NIPS 2018: The 32nd Annual Conference on Neural Information Processing Systems*, 2018, pp. 6639–6649.

[47] A. V. Dorogush, V. Ershov, and A. Gulin, "Catboost: gradient boosting with categorical features support," *arXiv preprint arXiv:1810.11363*, 2018.

[48] F. Tajaddodianfar, J. W. Stokes, and A. Gururajan, "Texception: A character/word-level deep learning model for phishing url detection," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 2857–2861.

[49] H. Tupsamudre, A. K. Singh, and S. Lodha, "Everything is in the name–a url based approach for phishing detection," in *International Symposium on Cyber Security Cryptography and Machine Learning*. Springer, 2019, pp. 231–248.

[50] P. Lison and V. Mavroeidis, "Automatic detection of malware-generated domains with recurrent neural models." *arXiv preprint arXiv:1709.07102*, 2017.