URL-based Phishing Detection using Entropy of Non-Alphanumeric Characters

A Thesis Submitted to the Department of Computer Science and
Communications Engineering,
the Graduate School of Fundamental Science and Engineering
of Waseda University
in Partial Fulfillment of the Requirements for the Degree of
Master of Engineering

Submission Date: July 22nd, 2019

Eint Sandi Aung
(5117FG17-5)

Advisor: Prof. Hayato Yamana
Research guidance: Research on Parallel and Distributed
Architecture

ABSTRACT

Cyber phishing is a theft of personal information in which phishers, also known as attackers, lure users to surrender sensitive data such as credentials, credit card and bank account information, financial details, and other behavioral data. The ones who commit such crimes are called ‘Phishers’ or ‘Attackers.’ Phishers act as if they are reliable sources to lure users to gain access/control to their system. Phishing detection is becoming a crucial research area, attracting increased focus as the number of phishing attacks grows, since e-commerce and internet transactions are growing rapidly.

Furthermore, because phishers are innovating various techniques, phishing detection has become a primary concern of developers. Moreover, as long as phishers are innovating their schemes, researchers have no way except to tackle every possible detection technique. Detection mechanisms come along with a vast variety of techniques since no one can be sure which techniques phishers are trying to come up with. Therefore, it is still an interesting yet challenging issue.

We focus on URL (Uniform Resource Locator) -based phishing detection techniques since URL is a significant criterium in preventing phishing attacks without accessing to webpage directly. Hypothesis is that phishers create fake websites with less content information on the webpage as possible – showing only a few words in the webpage. When phishers rarely show content information in a webpage, we cannot retrieve enough features from the webpage by using detection approaches such as content and visual similarity-based. To overcome the limitation of those approaches, we focus on URL-based detection since we can extract features by analyzing URLs only, without accessing to the webpage.

Since previous works extract features of specific special characters, we assume that non-alphanumeric (NAN) characters distribution highly impact phishing URLs. Our contribution is to propose a new feature called entropy of NAN characters and compare with the previously used features, which are from previous researches. To be noted, those previous features are not from only one specific work but are applied on several works. We also emphasize on features engineering because selecting features (NAN characters in our work) affects the most on performance. As it is difficult to gather exactly same datasets used by previous works, we work on our datasets and compare with our contributed feature. We work on two datasets (balanced and imbalanced) and perform feature selection and hyperparameter tuning. We achieved 96% of ROC_AUC with balanced dataset and 89% with imbalanced dataset, which outperforms 87% in balanced and 84% in imbalanced datasets, respectively. Then, we summarize our findings and suggestions for better outcome of phishing detection.

LIST OF CONTENTS

ABSTRACT	i
LIST OF CONTENTS.....	ii
LIST OF FIGURES	iv
LIST OF TABLES.....	v
CHAPTER 1	1
INTRODUCTION	1
1.1. What is Phishing?	1
1.2. Overview of URL-based Phishing Techniques	1
1.3. Problem Statement	1
1.4. Motivation	2
1.5. Contribution.....	2
1.6. Thesis Organization.....	3
CHAPTER 2	4
RELATED WORK.....	4
2.1. Overview of Phishing Detection Architecture	4
2.2. Whitelist-based approach	4
2.3. Blacklist-based approach.....	5
2.4. Content-based/Heuristic-based approach	6
2.5. Visual Similarity-based approach	6
2.6. URL-based approach	7
2.7. Other approaches	7
CHAPTER 3	11
METHDOLOGY	11
3.1. Our URL-based Detection System	11
3.2. Proposed Feature – Entropy	11
3.3. Machine Learning.....	12
3.3.1. Supervised Learning.....	13
3.4. Machine Learning Techniques as Building Blocks.....	13
3.4.1. Naïve Bayes.....	13
3.4.2. Support Vector Machine	13
3.4.3. Random Forest	14
3.5. Feature Selection and Hyperparameter Tuning.....	14
CHAPTER 4	16
DATASETS, FEATURE SELECTION AND EXTRACTION.....	16

4.1.	Datasets	16
4.1.1.	Datasets Nature.....	16
4.2.	Validate Active URLs for legitimate sites	16
4.3.	Non-Alpha Numeric (NAN) Character Selection	16
4.4.	Features Description and Extraction	17
4.4.1.	Feature Description	17
4.4.2.	Feature Extraction	17
CHAPTER 5		20
EXPERIMENTAL EVALUATION AND SUMMARY		20
5.1.	Evaluation Metrics	20
5.2.	Feature Selection	21
5.3.	Experiment with Balanced and Imbalanced Datasets	21
5.4.	Comparison between Previous and Current Evaluations with Hyperparameter Tuning	22
5.5.	Comparison between Feature Sets based on Frequency Probability and Entropy	25
5.6.	Summary and Discussion	26
CHAPTER 6		27
CONCLUSIONS		27
ACKNOWLEDGEMENTS.....		28
REFERENCES		29
PUBLICATION.....		33
APPENDIX		34

LIST OF FIGURES

FIGURES	Pages
2.1. Email spoofing and Fake Website -----	4
2.2. Phishing Detection Schemes -----	4
2.3. Whitelist-based detection -----	5
2.4. Blacklist-based detection -----	5
2.5. Content-based detection -----	6
2.6. Visual similarity-based detection -----	7
3.1. URL-based Phishing -----	11
3.2. URL-based Detection System -----	11
5.1. Feature Importance Score -----	21
5.2. Experimental Evaluation on balanced and imbalanced dataset -----	23
A.1. Xml file sample of legitimate dataset	34
A.2. Csv file sample of phishing dataset	34

LIST OF TABLES

TABLES	Page
2.1. Some of the Approaches and its Limitations	7
2.2. Comparison of Existing URL-based Phishing Detection -----	9
4.1. Datasets Nature -----	16
4.2. Data Sources -----	16
4.3. List of NAN Character	17
4.4. Feature Description -----	17
5.1. Evaluation Results for three Classifiers on balanced and imbalanced datasets using previous and entropy features -----	22
5.2. Comparison of with and without entropy -----	22
5.3. Comparison between Previous and Current Evaluations after Hyperparameter Tuning -----	24
5.4. Comparison between Feature Set1 (Percentage of NAN) and Feature Set2 (Entropy of NAN)	25

CHAPTER 1

INTRODUCTION

Cyber phishing is no longer an unfamiliar topic since the Internet became an essential daily basis. People can do almost everything staying at home and sitting in front of their PCs – from groceries shopping to health support services. It was reported that the number of internet users significantly increased from nearly 0.15 billion in 1998 to 4.32 billion in 2018, which is a growth of approximately 30 times [1]. This indicates that people strongly depend on internet on daily basis. At the same time, enormous amount of information makes some people allured to steal information to access to a system and committing crimes. To analyze the reason of crimes, phishers commit not only for money, but for fame, acknowledgement or out of curiosity. Thus, phishing detection draw attention from researchers.

1.1. What is Phishing?

Phishing is a cyber threat in which attackers take advantage of users by mimicking legitimate authentic websites in order to steal sensitive information such as passwords and bank statements. Phishing is performed through different mediums: internet, short message service and voice. Their targeted vectors can be email, instant messaging, smishing (short message phishing), vishing (voice phishing) and websites [2]. In our research, phishing refers to web phishing through the Internet. Although phishing can be protected against by: (1) user awareness, and (2) technology-based approaches, the former cannot be completely trusted since it relies on humans—not all of whom are aware of phishing.

1.2. Overview of URL-based Phishing Techniques

URL-based phishing attacks are mainly performed by embedding sensitive words or characters in a link that:

- Mimic similar but misspelled words.
- Contain special characters for redirecting.
- Use shortened URLs.
- Use sensitive keywords which seem reliable.
- Add a malicious file in the link and so on.

1.3. Problem Statement

Anti-Phishing Working Group (APWG) [3] addressed that the total number of unique phishing websites detected is approximately 785,000 in 2018, which is a significant number in the area. They also said that use of web page redirects to hide actual phishing sites to make victims misunderstand as legitimate sites has increased. When potential victims click on the links, they are redirected to phishing sites via several numbers of intermediate sites. Then, phishers request for credential information or install malware in the victims' computer. "This obfuscation technique is an effort by the phishers to hide the phishing URL – most notably from detection via web server log referrer field monitoring," said Stefanie Ellis, Anti-Fraud Product Marketing Manager at MarkMonitor [4].

Moreover, according to PhishLabs [5], HTTPS phishing sites became popular in the past two years ago. At the end of 2016, there were less than 5% of phishing sites found on HTTPS infrastructure. Later in Q4 of 2017, it raised up to 33%. Phishers target users' poor awareness of green lock smbc.co.jp, acting as if it is secured. Actually, the

green lock only shows that it uses SSL certificates, but not the security of the site. In addition, phishers mimic websites which are similar to legitimate ones, without providing much information on the content, which are later analyzed by researchers for detection purpose. They hide suspicious information as less as possible to lure users.

To sum up, we can conclude the problem statement as follow:

- Users being tricked by URLs based on users' lack of knowledge
- Showing information of a webpage's content as less as possible

1.4. Motivation

Regarding the problem statements mentioned above, we need a detection scheme which 1) alerts users before clicking the link and 2) classifies phishing or not by analyzing URLs, before accessing their content themselves. Thus, we assume that detecting phishing URLs can be effective even in which there is no or less content information on a webpage, compared to other approaches such as content and visual similarity-based. According to [3, 5], we notice that phishers have an increasing interest in URL-based attacks to gather sensitive information. Moreover, URL-based phishing detection can reduce unnecessary access to the phishing page, compared to other approaches such as blacklisted websites used in google safe browsing, content and visual similarity. Thus, although there are a wide range of phishing detection schemes such as blacklist-based detection, whitelist-based detection, content-based detection and visual similarity-based detection, we emphasize URL-based phishing detection in our research. Previous URL-based phishing detection such as [7, 8] have used feature such as URL length, which could be manipulated by phishers and also bias to the dataset when we do not have path information of a URL for legitimate websites – Alexa's top 1M dataset provide domain names only, and Alexa ranking – free hosting services (000webhost.com) ranked high in Alexa which are also used by phishers. [7] performed feature extraction of specific special characters (such as '.', '-', '=', '_', '?', '/'). Thus, we assume that distribution of NAN characters effects on phishing detection and propose entropy calculation of NAN as an important feature.

1.5. Contribution

Because of the problems of Alexa-ranking feature and URL length feature, in our work, we do not deploy these features. According to [7, 8], we assume that participation of NAN characters greatly affects phishing detection because phishers create fake URLs with NAN characters such as :

- 1) extra unnecessary dots,
- 2) '//' to redirect to a completely different domain,
- 3) '-' in the domain to mimic similar, yet different website
- 4) unnecessary symbols

However, instead of directly using the frequencies of NAN characters found in URL, we propose a new feature to compute distribution of special characters. Our contribution is to propose a new feature called **entropy** of non-alphanumeric characters (NAN) for phishing detection only by analyzing their URLs.

Although we know that 12 features in total in our system is not sufficient in the real-world system, our objective is to confirm how effective our proposed entropy of NAN is in the system. We contribute a new feature which is useful in URL-based phishing detection whenever less or no information is available in phishing websites.

1.6. Thesis Organization

Our thesis is organized as introduction in chapter 1 with its problem statement and motivation. Related work including phishing detection architecture in chapter 2, and then in chapter 3, we present our phishing detection system, machine learning methods as building blocks, feature selection method applied in our work. Then, we describe the dataset we work on and feature selection and extraction with detailed explanation in chapter 4 and later, our experimental evaluations and summary on chapter 5. Eventually, we conclude our thesis in chapter 6.

CHAPTER 2

RELATED WORK

2.1. Overview of Phishing Detection Architecture

Although there are various medium and vector to be categorized for phishing – internet as medium, email or instant messaging as vector and social engineering as technical approach [2], we can mainly classify the attacks into two according to social engineering perspective [6]: fake websites (Figure. 2.1 (a) – compromised domain) and email spoofing (Figure. 2.1 (b) – via email).

Figure 2.1. Email Spoofing (a) Fake Website (b)

We survey phishing detection approaches for better understand how it works. Phishing detection schemes are categorized differently in various papers[9,10,11,12]. Thus, we categorize them in general as shown in Figure 2.2.

Figure 2.2. Phishing Detection Schemes

To explain briefly, whitelist and blacklist schemes use its periodically updated database. Whitelist saves login information and alerts users if abnormal login happens while the latter checks if current URL is in periodically updated blacklisted data (e.g. Google Safe Browsing). Content based detection comes with several techniques such as text mining (e.g. keyword extraction and search engine), html and JavaScript analysis (e.g. login form or iframe detection) and so on. Meanwhile, visual similarity scheme solely extracts signatures of webpages for detection. URL-based detection consists of mining keywords in URL for target domain, domain-based detection or hybrid detection (keywords extraction from domain and content).

2.2. Whitelist-based approach

Kang et al. [13] proposed an approach based on white-listed sites in 2007. They performed a URL similarity check to distinguish phishing sites from otherwise for local pharming– changing host file in the local computer and a mechanism comparing with Domain Name System (DNS) query to overcome DNS pharming attacks which is DNS record of the URL has been spoofed – i.e. problem for relying on DNS from previous

works. Although whitelist-based methods seem effective for phishing detection, there is a limitation on obtaining all legitimate sites on the web. An abundant list of reliable websites is necessary for a robust system with high accuracy; otherwise, false positive rates increase due to a lack of white-listed websites information, which is practically impossible to collect all legitimate sites in the world.

In 2008, Cao et al. [14] also presented an automated individual whitelist approach (AIWL), in which the system maintains a user's previous login and warns when unfamiliar access has occurred. They proposed trusted Login User Interfaces (LUIs) features. It stored legitimate website list where users submitted sensitive information. Their system alarms users whenever sensitive information is sent to any webpage that are not in their list. However, AIWL warns an alert whenever any information is sent to any other page in the list although the page is genuine.

Figure 2.3. Whitelist-based detection

2.3. Blacklist-based approach

Web browsers – such as Google Safe Browsing [15] – defend against phishing attacks by updating a list of black-listed sites. It uses user application to verify whether a given URL is blacklisted. Currently, it implements two blacklist services ‘goog-phish-shavar’ (phishing) and ‘googmalware-shavar’ (malware).

In 2008, Sharifi et al. [16] proposed a new blacklist generator technique to solve the common issues of maintaining an up-to-date list.

PhishNet [17] also predicts phishing attacks based on a blacklist scheme. It solves the problem of exact matching. It uses five heuristics – top level domain, IP address, directory structure, query string and brand name– for combinations of blacklists to predict new phishing sites. Although it cannot detect zero hour phishing sites – newly created phishing sites, it achieves 95% true positive rate and 3% false positive (FP) rate over large datasets.

Generally, blacklist approach has lower FP rate than heuristic approaches.

Figure 2.4. Blacklist-based detection

Since blacklist-based system relies on third-party services (like Google) for searching domain name to compare top results, it results in poor performance. Furthermore, blacklist approaches encounter the major issue of zero hour phishing attacks because newly created phishing sites are not in the list, although services such as Google Safe Browsing use periodically updated blacklist. Google Safe Browsing [15] also fails to detect phishing if blacklisted domain's IP address is changed. PhishNet[17] has a problem that if a URL is a slightly changed version of blacklisted one, it remains undetected.

2.4. Content-based/Heuristic-based approach

SpoofGuard [18] is a browser plug-in for Internet Explorer (IE) using a set of heuristics for anomalies detection in a webpage. It checks if the given URL is similar to whitelisted URLs and also detects if a hidden attribute – text attribute is different from actual one, is present. It cannot detect if URLs are not defined properly according to its rules.

PhishGuard [19] – also a browser plug-in – performs detection based on HTTP authentication. It sends same user ID, but different random passwords. If the response is HTTP 200, it detects as phishing. PhishGuard [19] leads to credential theft if the websites reply unauthorized.

Zhang et al. [20] presented a novel approach, so-called CANTINA in 2007. Their work is based on Term Frequency – Inverse Document Frequency (TF-IDF) information retrieval algorithm used to detect phishing websites. CANTINA alone resulted in a high false positive rate due to limitations on the number of search engine results. It caught 97% phishing sites with 6% false positive rates, which is not optimal. Thus, they used several heuristics to reduce the false positive rate and improve accuracy. Their approach achieved a better outcome compared to popular anti-phishing toolbars, achieving 90% of phishing sites detected with only 1% false positive rate.

In 2011, Xiang et al. [21] further improved CANTINA, calling it CANTINA+, which is regarded as the most comprehensive feature-rich approach in content-based phishing detection. It achieved a better 0.4% false positive rate with 10% training phish and over 92% true positive rate on unique testing phish.

Since both CANTINA and CANTINA+ use search engines and third-party services, DNS compromising became a challenging threat.

Moghimi et al. [22] in 2016 proposed a rule-based method to detect phishing internet banking by extracting two feature sets from the content of webpage; page resource access protocol feature set and page resource identity feature set. They used approximate string matching algorithms to determine the relationship between the content and the URL of a webpage. They achieved accuracy of 99.14% TP and 0.86% FN. However, their system completely depends on the webpage content. If phishers use efforts and redesign the page, then it will lead to high FP rate. Moreover, their features are extracted by tracing DOM. Thus, if phishers use flash media or image of legitimate webpage instead, they cannot classify correctly.

Similar works can be found in [23-25].

Figure 2.5. Content-based detection

2.5. Visual Similarity-based approach

Wenyin et al. [26] proposed a simple visual-similarity-based approach in 2005. Their system performed phishing detection on three levels of similarity matrices; (i) block level similarity, (ii) layout-similarity and (iii) overall-style similarity. However, the most representative work on visual similarity was later presented by Fu et al. [27] in 2006 using the Earth Mover Distance (EMD). EMD was used to calculate the signatures of two images for visual similarity. Their method performed well in accuracy with 89% true positive and 0.71% false positive rates. The significant workload

required to process two images has a performance drawback compared to other approaches.

Chen et al. [28] introduced a heuristic anti-phishing system to model perceptual similarity. They employed a logistic regression algorithm for normalizing page content features. Although the proposed method achieved 100% true positive rate, it had 0.74% false positive rate, which is higher than [27]. There are many similar works based on visual similarity including [29-39].

Figure 2.6. Visual similarity-based detection

2.6. URL-based approach

Aburrous et al. [40] proposed an intelligent phishing detection system for e-banking using fuzzy data mining in 2010. The experiment was performed based on fuzzy logic with data mining algorithms. They showed how effective URL-based approaches are for phishing detection.

Yuan et al. [8] in 2018 proposed to extract features from URLs and webpage links. Their work is based on hybrid approach (consist of both URL-based features and content-based features). However, they performed statistical (such as mean, median between features) based on URLs along with lexical features (such as title and textual content). They achieved 98.3% of TP and 2.6% of FP, however, the datasets are relatively small i.e. 3305 legitimate and 2892 phishing websites are used. Furthermore, they used URL length feature and Alexa ranking feature, in which URL length feature can be manipulated by phishers and Alexa ranking feature ranks free hosting domain such as 000webhost.com, which are commonly used by phishers.

In general, although URL-based schemes perform well without knowing content information of webpage, they highly depend on feature extraction. Furthermore, performance can vary depending on which features we apply. Thus, selecting features matters in URL-based detection.

Overall, URL-based methods perform faster than any other approaches, including content and visual similarity-based approaches. More importantly, they work well on zero hour phishing attacks, which are becoming a major concern in modern anti-phishing society. In upcoming sections, we further discuss details of URL-based detection.

2.7. Other approaches

A variety of alternative techniques are used by researchers in phishing detection. Such techniques include heuristic [41], hybrid [42], machine learning [43], DNS-based and others [44]. Additionally, several surveys regarding different schemes are performed by researchers [45-46].

We listed some of the phishing detection approaches and limitations in Table 2.1.

Table 2.1. Phishing Detection Approaches

Approach	Solution	Pros	Limitation/Remark
WhiteList (AIWL)[14]	<ul style="list-style-type: none"> • Maintain own individual whitelist • Alert users for possible attack if credential to 	<ul style="list-style-type: none"> • Perform well on detection (LUI authentication) by storing entire LUI 	<ul style="list-style-type: none"> • Warn whenever any information is sent to any other trusted page, that are not in the whitelist since AIWL maintains previous successful LUI of websites

	Login User Interface (LUI) not in the white-list is submitted.	information rather than only URL. <ul style="list-style-type: none"> Efficiently defend pharming attacks by alerting users when legitimate IP address is maliciously changed (Anti-Pharming, which cannot be detected by SpoofGuard [18]) 	<ul style="list-style-type: none"> Difficult to defend against local machine Trojan Horse and viruses since whole AIWL is installed in local PC
BlackList (google safe browsing) [15]	<ul style="list-style-type: none"> Block if URL is in blacklist 	<ul style="list-style-type: none"> Privacy: use hashed URL to preserve the actual URL queried by user 	<ul style="list-style-type: none"> Cannot detect zero-day attack and when IP is changed
Content/Heuristic (PhishGuard)[19]	<ul style="list-style-type: none"> Identify phishing website by submitting random credentials in a login process before submitting actual credentials. 	<ul style="list-style-type: none"> Can detect phishing websites with two possibilities: 1)always respond with authentication failure, 2)always respond with authentication access, based on the fact that phishing sites only store the credentials for future use and do not verify them. 	<ul style="list-style-type: none"> Credential theft if unauthorized access is replied in case of getting http 401 response in submission of correct credentials after several 401 responses. e.g. http 401 have two meanings: 1) wrong password error, 2) website indicates/alerts failed authentication in default. Authentication failure alert but “HTTP 200 OK” response, instead, along with a new page in some legitimate websites (such as ICICI bank)
Visual Similarity [35]	<ul style="list-style-type: none"> Detect phishing by finding similarities between phishing and legitimate websites based on text attributes, hidden image and overall visual appearance 	<ul style="list-style-type: none"> Integrate visual similarity detection with open source tool AntiPhish [47] to overcome the problem of reuse of legitimate credentials warns as suspicious in [47] 	<ul style="list-style-type: none"> Cannot distinguish if text is replaced with image of same appearance
URL [8]	<ul style="list-style-type: none"> Statistical [mean, median etc.] and lexical [title, text content] features are used 	<ul style="list-style-type: none"> Work regardless of webpages in different languages. Work well on zero-hour phishing attack 	<ul style="list-style-type: none"> Alexa rank feature gives phishing domain in high rank (e.g. 000webhost.com)

Table 2.2. Comparison of Existing URL-based Phishing Detection

PAPER	PERFORMANCE							DATA SET					URL FEATURES	ALGORITHMS	YEAR
	ACC	TP	FP	Prec	R	F1 SCORE	AUC	DATASET SOURCE		DATASET SIZE		DATASET TYPE			
								LEGITIMATE	PHISHING	LEGITIMATE	PHISHING				
[48]							99.29	VIRUSTOTAL	VIRUSTOTAL	16M	0.9M	IMBALANCED	PRIMARY DOMAIN FEATURE, PATH FEATURE, FILE EXTENSION FEATURE (*3)	CNN	2017
[49]	99.09							ALEXA+ PAYMENT GATEWAY+ TOP BANKING WEBSITE	PHISHTANK+ OPENPHISH	1600+ 66+ 252	1528+ 613		NO. OF DOTS, SPECIAL SYMBOLS, URL LENGTH, SPECIAL WORDS, POSITION OF TLD, HTTP COUNT, BRAND NAME, DATA URI (*8)	RF SVM NN LR NB	2017
[50]	98.76			98.60	98.93	98.76	99.91	COMMON CRAWL	PHISHTANK	1M	1M	BALANCED	PATH LENGTH, URL ENTROPY, LENGTH RATIO, '@' AND '-' COUNT, PUNCTUATION COUNT, TLDS COUNT, IP ADDRESS, SUSPICIOUS WORDS COUNT, EUCLIDEAN DISTANCE, KOLMOGOROV-SMIRNOV STATISTIC (*10)	RF LSTM	2017
[51]	95.80							ALEXA+ SEARCH ENGINE	PHISHTANK	500+ 500	1000	BALANCED	URL SIZE, NO. OF HYPHENS, NO. OF DOTS, NO. OF NUMERIC CHARACTERS, IP ADDRESS, SIMILARITY INDEX (LEVENSHTEIN, JARO WINKLER, NORMALIZED LEVENSHTEIN, LONGEST COMMON SUBSEQUENCE, Q GRAM, HAMMING) (*6)	NB BAYES SVM	2018
[52]	95.00								PHISHTANK				IP ADDRESS, REDIRECTION OF PAGE USING “/”, ADDING PREFIX OR SUFFIX SEPARATED BY “-”, SUBDOMAIN AND MULTI-SUBDOMAIN, URLS HAVING @ SYMBOL (*6)	IG RANKER METHOD	2018
[53]		99.70	0.40	99.70	99.70	99.70	1.00	DIGG58.COM+ GITHUB	PHISHTANK+ GITHUB	16516+ 37,667	12483+ 24,905	SLIGHTLY IMBALANCED	IP ADDRESS, NO. OF DOTS, NO. OF “/”, SPECIAL CHARACTERS, ABNORMAL LENGTH OF DOMAIN, CHARACTER DISTRIBUTION (*36)	RF MLP NB LR J48 SVM	2018
[54]							70.10-EBAY 71.01-PAYPAL	EBAY+ PAYPAL+ BANK OF AMERICA + SORIO ET AL.	EBAY+ PAYPAL+ BANK OF AMERICA+ SORIO	18800+ 17572+ 9408+ 82101	8529+ 9690+ 4610+ 6562	IMBALANCED	NO. USAGE OF DOMAIN NAME, URL LENGTH, DOMAIN SEPARATED BY “-”, MULTIPLE SUBDOMAINS, USAGE OF “@” SYMBOL,	SVMSMOTE BSMOTE2 RMR ADASYN	2018

							70.10-BOA 97.65-SORIO		ET AL.				NO. OF TLD IN THE PATH, NO. OF SUSPICIOUS WORDS, NO. OF PUNCTUATION SYMBOLS USED, DIGITS IN DOMAIN, ENTROPY, KULLBACK-LEIBLER DIVERGENCE, NO. OF “-” IN PATH, VOWEL/CONSONANT RATIO, DIGIT/LETTER RATIO, USAGE OF BRAND NAMES, LONG HOSTNAMES, SHORT HOSTNAMES, NO. OF “.” IN HOSTNAME (*18)		
[55]	96.89							SEARCH ENGINE+ COMMON CRAWL+ TWITTER STREAM API	PHISHTANK	456300			FEATURES FROM [11][48]	ANN LSTM	2018
[56]	D1: 99.47 D2: 99.92								PHISHTANK+ OPENPHISH+ MALWARE DOMAIN LIST+ MALWARE DOMAINS	DATA SET1: 90101 DATA SET2: 26000				CNN CNN-LSTM BIGRAM	2018
[7]	99.44							ALEXA	PHISHTANK + MALWARE DOMAIN LIST+ SPAM DOMAIN LIST JWSPAMSPY	26041	26041	BALANCED	117 STATIC AND DYNAMIC FEATURES	J48 SIMPLE CART RF RT ADTREE REPTREE MAJORITY VOTING	2018
[8]	97.70	98.30	2.60					ALEXA+ NETWORK SECURITY CHALLENGE	PHISHTANK	3305	2892		IP ADDRESS, SUSPICIOUS CHARACTERS, NETWORK PROTOCOL, ALEXA RANKING, LENGTH OF ENTIRE URL, LENGTH OF HOSTNAME, LENGTH OF MAIN DOMAIN NAME, NO. OF DOTS IN HOSTNAME, NO. OF DOTS IN URL PATH, URL TOKEN COUNT, HOSTNAME TOKEN COUNT, SEARCH ENGINE RESULT (*12)	KNN LR RF DT GBDT XGBST DF	2018

CHAPTER 3

METHDOLOGY

In this section, we first present overview of URL-based detection system in section 3.1 and then describe our proposed feature – entropy for Non-Alpha Numeric (NaN) characters in section 3.2. Then, we mention about binary classification for two classes of legitimate and phishing in our work in section 3.3 and the machine learning techniques which we use as building blocks in 3.4. We conclude this section with feature engineering and hyperparameter tuning that are used in preprocessing stage and classification stage.

3.1. Our URL-based Detection System

Figure 3.1. Our URL-based Detection System

Figure 3.2. shows how URL phishing is performed. When phishers mimic reliable sites, users submit credential information to the attackers without knowing the website is faked.

Figure 3.2. URL-based Phishing

3.2. Proposed Feature – Entropy

We know that in URL-based detection, performance greatly depends on the types of features and the number of features because we can access URLs alone in detection. We use 12 features in total, in which 10 out of 12 are previously used features in various

papers called “old features” and 2 of them are our new applied features called “new features”. The reason of choosing the features is that accessing to the webpage is not required to perform feature extraction. However, our objective is to confirm how effective our entropy feature of NAN is. The key idea for our contribution of computing entropy on NAN is to find out how they are distributed on each URL. This is because previous works [7, 8] extract frequencies of specific special characters such as “-”, “/”, “_”, “.” (NAN characters in our work) in each URL, and our hypothesis is that the number of times phishers use special characters are greater than that of times legitimate websites use.

Thus, instead of measuring frequencies of specific special characters and representing as several features, we compute entropy to measure the frequentist probability distribution between phishing and legitimate websites and represent as single feature. For example: instead of measuring Number of “.” as a feature $F1$, Number of “-” as feature $F2$, Number of “@” as $F3$ and representing a vector for a URL as $[F1, F2, F3]$, we compute entropy of NAN characters as feature $F1$ representing the vector as $[F1]$ where $F1 = \text{entropy}(\text{frequentist probability distribution of NAN characters})$. Since the presence of special characters in phishing and legitimate websites are different, we assume that measuring their distribution results a precise value for a better classification. We use entropy because it is a measure of disorder (as well as a measure of purity) and it is computed based on frequentist probability distribution. High level of disorder means low level of purity – if probabilities between special characters are not much different (i.e. $4/7$ and $3/7$), it has high entropy or low level of purity. Otherwise, low entropy or high purity.

We define entropy of NAN characters as follows:

$$\text{Entropy} = - \sum_{i \in T} (P_i \log P_i) \quad (3.1)$$

,where, i = the i^{th} NAN character in T i.e., $T(i)$

$T = \{\text{list of NAN characters}\}$

P_i = the frequentist probability of the i^{th} NAN character

To be noted, we use logarithmic base 2 in calculation.

A high entropy is considered a high level of disorder (i.e. low level of purity). For example: Assume we have 5 different NAN characters {“-”, “.”, “_”, “@”, “&”} whose appearance probabilities are same as 0.2, then we have $-\sum_{i=1}^5 (0.2 \log 0.2) = (-2.321928)$. Note: Entropy is generally measured between 0 and 1, however, depending on the number of elements in dataset, entropy can be greater than 1. But, it means the same.

We consider that NAN characters play a huge impact on classification of phishing detection. Thus, we use entropy to find out how each NAN characters are equally distributed on each URL after feature selection of those characters are performed. Entropy feature is applied after important features of NAN characters are selected. We consider probability distribution of NAN characters found in each URL, not distribution over entire datasets.

3.3. Machine Learning

Machine Learning is an automated learning in which we make computers process a program itself from past knowledge available without the need of external help. It learns the past experience by itself and converts it to knowledge. In machine learning, input is the training data while output is the knowledge it learned from the previous experience. We need machine learning when the tasks we want to perform are complex

and we need a well-fitted program that adapts to the input which varies in time. We can categorize machine learning into supervised and unsupervised learning in general although it has other categories such as semi-supervised and reinforcement learning.

3.3.1. Supervised Learning

Supervised learning is a learning which already has a ground truth to check if a result is correctly predicted. In supervised learning, each training data has its label (i.e. ground truth/class) and all of them are fully labelled. Supervised learning can solve two problems, which are classification and regression. Regression is used for problems in which output values are numerical or continuous, for example, stock price prediction. However, classification is applied for prediction in which output values are categorical or discrete, for example, spam detection. In phishing detection area, we apply classification to analyze if the URL vector belongs to phishing or legitimate and output a discrete result. In our case, we apply binary classification, phishing (-1) and legitimate(1).

3.3.1.1. Binary Classification

Binary classification is also a type of supervised learning. It identifies a given data into one of two classes. Binary classification is commonly applied in problems such as whether a patient has a certain disease, or if a student passes exam. In our research, we apply binary classification since we only consider two labels for the input URL: phishing and legitimate. Some of the commonly used methods in binary classification are Logistic Regression, random forest (RF), Bayesian (which can be naïve bayes NB or gaussian naïve bayes GNB, support vector machine (SVM), Neural Networks and Decision Trees.

3.4. Machine Learning Techniques as Building Blocks

We first apply three algorithms such as NB, SVM and RF in our research to find out which algorithm fits well. To be noted that we use machine learning techniques as building blocks. We listed them as follows.

3.4.1. Naïve Bayes

Naïve Bayes (NB) is a simple yet effective classifier used in numerous applications. In NB classifier in our case, we define conditional probability of our feature vector $P(x|y)$ as follows:

$$P(y = -1|x) = \frac{P(x|y=-1)}{P(x|y = -1)+P(x|y=1)} \quad (3.2)$$

Where, x = a feature vector

$y \in \{1,-1\}$ = class label of either a phishing(-1) or legitimate website(1)

$P(x|y)$ = the conditional probability of a feature vector given its label.

3.4.2. Support Vector Machine

Support Vector Machine (SVM) is a discriminative classifier defined by the optimal separating hyperplane between labels. SVM outputs an optimal hyperplane by dividing a line between class labels in two dimensional space. Here, since we don't have linearly separable training data in the real-world, kernel trick is necessary. It maps non-linearly separable data into higher dimensional space, such that, an optimal hyperplane can be found to separate samples. In our work, we deploy SVM classifier from sklearn by applying our extracted feature vectors with respected class labels (phishing -1 and legitimate 1).

3.4.3. Random Forest

Random forests (RF) is built with random attribute selection using bagging. RF employ a divide and conquer approach (ensemble mechanism) for improving performance. In RF, the mechanism combines various random subsets of trees. The overall result is calculated based on the average, or weighted average, of the individual results. RF improves the problem of overfitting of the decision trees. We use RF not only for classification with hyperparameter tuning but also for feature selection.

3.5. Feature Selection and Hyperparameter Tuning

In the feature engineering field of phishing detection, researchers apply several features depending on their detection techniques. As our research focuses on URL solely, it is crucial to decide which features are applied.

3.5.1. Feature Selection

Irrelevant or partially relevant features may bring negative effect on performance of a model. The data features which are used to train machine learning models have a great influence on the performance. Feature selection is the very first and most important step of model designing. Feature selection can reduce overfitting and improve accuracy. Thus, we consider feature selection is essential before applying our entropy contribution. As for our case, we assume feature selection as identifying negatively affected NAN characters on the dataset and removing them before applying entropy of NAN as a feature. Thus, we perform selection in order to drop unnecessary NAN characters, which have less or no contribution over model's performance.

3.5.1.1. Feature Importance

We perform feature selection from feature importance scores among 21 NAN characters, in which our feature vector contains frequencies of each NAN character in a URL with its class label. We get feature importance scores for each feature from feature importance property of RF model. Since feature importance is an inbuilt class among decision tree based classifiers, we use RF Classifier to extract the top 10 NAN characters from the dataset. The higher the score is, the more important or relevant the feature is towards the output.

3.5.2. Hyperparameter Tuning with Random Forest

Grid Search takes in as many hyperparameters as we want. However, we are considering `n_estimators`, `bootstrap`, `cv` and `criterion`. It tries possible combination of the hyperparameters with cross-validation we apply. It is a good way to determine the best hyperparameters values to use although it is time consuming. Hyperparameter is a parameter of the model which is predefined before starting the learning process. Although different models have different hyperparameters set, we use RF classifier for hyperparameter tuning.

n_estimators: It specifies the number of trees in the forest of the model. The default value for this parameter is 100 in version 0.22. 100 different decision trees will be constructed in RF. However, we set `n_estimators` in range from 100 to 200 with increase of 10 in each time, and 300 to 700 with increase of 100 in each time.

bootstrap: we specify whether bootstrap samples are used when building trees. It has True and False values.

criterion: It is a function to measure the quality of a split. For Gini impurity, 'gini' criterion and Information Gain 'entropy' are used.

cross-validation (CV) : We apply K-Fold cross-validation which is the most common method. We set 10 folds CV in our tuning.

CHAPTER 4

DATASETS, FEATURE SELECTION AND EXTRACTION

In our work, we discuss about the preprocessing of Non-alphanumeric (NAN) features. We also work on both balanced and imbalanced dataset. Here, balanced dataset means for which class distribution are equal and imbalanced for unequal class distribution. To be noted that we consider two classes; phishing and legitimate. Our hypothesis in this work is that the number of phishing data cannot be compared to that of legitimate websites in the real world, and we cannot guarantee the same effect on classification of phishing with balanced and imbalanced dataset. In this section, we mention about datasets in section 4.1, and preprocessing and NAN selection in 4.2 and 4.3. In the last section of chapter 4, we describe feature descriptions and extractions in details.

4.1. Datasets

In our system, we collect phishing data from PhishTank [57] and legitimate data from DMOZ [58] (aka Curl). We use two datasets, which are balanced and imbalanced on RF, to find out the best-fitted algorithm among SVM, Gaussian NB and RF without adjusting anything. Then, we apply the two datasets along with hyperparameter tuning with RF.

4.1.1. Datasets Nature

Table 4.1. Datasets Nature

<i>Dataset</i>	<i>Size</i>		<i>Type</i>
	<i>Legitimate</i>	<i>Phishing</i>	
D1	5,000	5,000	Balanced
D2	95,754	10,473	Imbalanced

Table 4.2. Data Sources

<i>Dataset</i>	<i>Source</i>	
	<i>Legitimate</i>	<i>Phishing</i>
D1	DMOZ	PhishTank
D2	DMOZ	PhishTank

We collected phishing data from April 6th to April 8th 2019. For imbalanced dataset, the ratio of the number of legitimate sites to that of phishing sites is nearly 9:1. We perform our experiments on different datasets (balanced and imbalanced) to verify whether the performance of the proposed feature depends on the dataset size. Our idea is that in the real-world, we cannot expect that the number of legitimate sites and that of phishing sites are same. Thus, we also use imbalanced data to verify whether our proposed entropy still works on class imbalanced data. We illustrate the nature of our datasets in the Appendix.

4.2. Validate Active URLs for legitimate sites

Active check: We collect only active legitimate datasets. We check if they are still valid. We assume link is valid if we get HTTP response from requested link. We perform validation to ensure that the datasets we work on are up-to-date. As for phishing, we apply latest updated URLs as possible.

4.3. Non-Alpha Numeric (NAN) Character Selection

NAN character selection: We perform feature selection with its importance according to their feature importance. We perform the selection for two times, one for balanced dataset D1 and other for imbalanced dataset D2, because later, we compare experimental evaluation in both D1 and D2. In both selections, we split the dataset to 70% train and 30% test then perform the feature selections with training datasets, D1-

train for balanced and D2-train for imbalanced, respectively. Our hypothesis is that we do not know what kind of characters phishers are going to use and we do not want to define them randomly. Thus, it is better to use NAN characters selection before we apply entropy calculations for phishing detection.

Table 4.3. List of NAN Character

NAN CHARACTER	SYMBOL	NAN CHARACTER	SYMBOL
HASH	#	PERCENT	%
DASH	-	UNDERSCORE	_
DOLLAR SIGN	\$	QUESTION	?
ASTERISK SIGN	*	COMMA	,
LEFT PARENTHESIS	(, {	EQUAL	=
RIGHT PARENTHESIS	), }	AMPERSAND	&
SEMICOLON	;	TIDE	~
COLON	:	PERIOD	.
APOSTROPHE	'	PLUS	+
SLASH	/	AT SIGN	@
EXCLAMATION	!		

4.4. Features Description and Extraction

We extract feature vectors in order to apply classification. Features can be binary or nonbinary values. We describe previously proposed features by various papers as “**old features**”, otherwise, “**new features**” in section 4.4.2. Previously used features mean the features applied by previous papers, not our proposed features. We listed feature description in section 4.4.1 and extractions in section 4.4.2.

4.4.1. Feature Description

We consider 12 different features in our work described as follows (Table 4.4).

Table 4.4. Feature Description

FEATURE	FEATURE NAME	DESCRIPTION
F1	IP ADDRESS	CHECK IP ADDRESS IN DOMAIN[53,7,8,22]
F2	EXE	CHECK EXE FILE IN URL[7]
F3	SENSITIVE WORDS	CHECK SENSITIVE WORDS IN URL[53,7]
F4	// REDIRECT	CHECK // IN URL DOMAIN AND PATH[59]
F5	INTERNAL LINK	CHECK INTERNAL LINK (WWW/HTTP) IN URL DOMAIN AND PATH[7]
F6	AGE OF DOMAIN	CHECK DOMAIN AGE (>12 MONTHS)[7]
F7	PORT NUMBER	CHECK PORT NUMBER[8]
F8	- sign	COUNT NUMBER OF '-' SIGN IN URL[7,8]
F9	@ sign	COUNT NUMBER OF '@' SIGN IN URL[7,8]
F10	. sign	COUNT NUMBER OF '.' SIGN IN URL[7,8]
F11	ENTROPY	CALCULATE ENTROPY OF NAN CHARACTERS
F12	FREE HOST	CHECK DOMAIN USING FREE HOST SERVICES

4.4.2. Feature Extraction

We extract features according to the following rules. We split features into two categories; old features and new features.

4.4.2.1. Old Features

F1 – IP address (binary): check if URL contains an IP address. Return -1 if found, otherwise, return 1. Our hypothesis is that phishers use IP address instead of domain name to direct users to a phishing page to confuse users since they do not know where

the current link with IP address belongs to when they click it. (for example, <http://67.205.147.248/oft/index.php?produto=722415036>).

F2 – exe (binary): check if URL contains exe file. Return -1 if found, otherwise, return 1. Hypothesis is that phishers use exe files to run malware on background process.

F3 – sensitive word (binary): check if URL contains sensitive words[50] such as “confirm”, “account”, “banking”, “secure”, “login”, “signin” etc. Return -1 if found, otherwise, return 1. Hypothesis is that most of the phishers use those keywords to lure victims as if the site is legitimate.

F4 – // redirect (binary): check if a path of URL contains “//” to redirect users to phishing page. Return -1 if found, otherwise, return 1. Hypothesis is that “//” is mainly used for redirection by ignoring the left side of URL. (for example, <http://redirect.company.com/http://externalsite.com/page>)

F5 – internal link (binary): check if a path of URL contains another link. Return -1 if found, otherwise, return 1. Hypothesis is that phishers add internal URL in a main URL. (for example, <http://www.linkebuy.com.br/linkebuy/parceiro?protocol=http&url=www.google.com>)

F6 – age of domain (binary): check if the age of domain is less than 12 months. We check ‘whois’ property of the domain and get the creation date of the domain if we can successfully access to that domain. We check the creation date of the domain is less than 12 months of the current accessed date. Return -1 if true, otherwise 1. Hypothesis is that phishing links do not exist for a long time and phishers mostly use newly created links.

F7 – port number (nonbinary): check which port number is used. If it is 443 (HTTPS), then return 1. If 80 (HTTP), then 0, otherwise -1. Hypothesis is that previously phishers rarely use HTTPS. However, in 2017 PhishLab report[5], they started using HTTPS. Thus, we differentiate HTTPS with others.

F8 – ‘-’ dash count (nonbinary) : count frequencies of ‘-’ sign in a URL. Return frequencies of ‘-’ in each URL. Hypothesis is that phishers mimic as legitimate websites by adding ‘-’ in URL.

F9 – ‘@’ at count (nonbinary) : count frequencies of ‘@’ sign in a URL. Return frequencies of ‘@’ in each URL. Hypothesis is that when we analyze our phishing datasets, phishers more often use ‘@’ in URL, especially in URL query value than legitimate websites. (for example, <http://shunmas.com/sj/index.php?email=2@2.com>).

F10 – ‘.’ Dot count (nonbinary) : count frequencies of ‘.’ Sign in a URL. Return frequencies of ‘.’ in each URL. Hypothesis is that phishers either use more sub domains than legitimate websites, or unnecessary ‘.’ in URL path. (for example, <https://www.ssproduction.com.pk/spages/verify.php?ga=2.38170595.17086121.1551095253-653443608.1551095253&mail=laurent@tacer.biz..>).

4.4.2.2. New Features

F11 – entropy of NAN characters (nonbinary): Based on NAN character selection, we perform entropy computation. It is calculated per each URL. Hypothesis is that NAN characters are crucial in phishing detection since phishers use them to generate new links. Moreover, we use entropy to know how those NAN characters are distributed over each URL link.

F12 – free host (binary): check if URL uses free hosting domain (for example : 000webhost.com is mostly found in our phishing datasets). We surveyed manually about free hosting domain present in our phishing datasets and found several phishing domains with 000webhostapp.com. Any user can create a website with a domain name of 000webhostapp.com after signing up at free hosting service 000webhost.com. We

use free hosting services surveyed in [60] which are used by phishers, in 2012. Although it is not updated anymore, 000webhost is still popular and currently used among phishers. Hypothesis is that phishers mainly use free hosting URLs. Hosting service, such as 000webhost, ranks high in Alexa.com, however, they are mostly used by phishers. Thus, we target URLs from free hosting. Return -1 if true, otherwise, return 1.

CHAPTER 5

EXPERIMENTAL EVALUATION AND SUMMARY

We first describe evaluation metrics in section 5.1, and later we explain about feature selection in section 5.2. Section 5.3 and 5.4 present about our experimental evaluation results in details. Finally, we provide summary of our work and discussion in section 5.5.

5.1. Evaluation Metrics

Here, N represents the total number of websites and F represents the number of phishing websites and L represents the number of legitimate websites. N is the number of websites.

$F \rightarrow F$ = Phishing websites are correctly classified as phishing websites

$F \rightarrow L$ = Phishing websites are incorrectly classified as legitimate websites

$L \rightarrow L$ = Legitimate websites are correctly classified as legitimate websites

$L \rightarrow F$ = Legitimate websites are incorrectly classified as phishing websites

$N_{F \rightarrow F}$ = the number of correctly classified phishing websites

$N_{F \rightarrow L}$ = the number of phishing websites that are incorrectly classified as legitimate websites

$N_{L \rightarrow L}$ = the number of correctly classified legitimate websites

$N_{L \rightarrow F}$ = the number of legitimate websites that are incorrectly classified as phishing websites

$N_F = N_{F \rightarrow F} + N_{F \rightarrow L}$ = the total number of phishing websites

$N_L = N_{L \rightarrow L} + N_{L \rightarrow F}$ = the total number of legitimate websites

$N_T = N_{L \rightarrow L} + N_{L \rightarrow F} + N_{F \rightarrow F} + N_{F \rightarrow L}$ = the total number of websites

- **True positive rate (TPR)**: the ratio of the number of correctly classified phishing websites ($N_{F \rightarrow F}$) to the total number of phishing attacks ($N_{F \rightarrow F} + N_{F \rightarrow L}$) as shown in Equation(5.1).
- **False positive rate (FPR)**: the ratio of the number of legitimate websites that are incorrectly detected as phishing attacks ($N_{L \rightarrow F}$) to the total number of all existing legitimate websites ($N_{L \rightarrow L} + N_{L \rightarrow F}$) as shown in Equation (5.2).
- **True negative rate (TNR)**: the ratio of the number of correctly classified legitimate websites ($N_{L \rightarrow L}$) to the total number of legitimate websites ($N_{L \rightarrow L} + N_{L \rightarrow F}$) as shown in Equation (5.3).
- **False negative rate (FNR)**: the ratio of the number of phishing websites that are incorrectly classified as legitimate ($N_{F \rightarrow L}$) to the total number of phishing websites ($N_{F \rightarrow F} + N_{F \rightarrow L}$) as shown in Equation (5.4).
- **Precision (Prec)**: the ratio of correctly detected phishing websites ($N_{F \rightarrow F}$) to the total number of websites detected as phishing ($N_{L \rightarrow F} + N_{F \rightarrow F}$) as shown in (5.5).
- **Recall (R)**: equivalent to TP rate as shown in Equation (5.6).
- **Accuracy (ACC)**: the ratio of the sum of correctly classified phishing and legitimate websites ($N_{L \rightarrow L} + N_{F \rightarrow F}$) to the total number of websites ($N_{L \rightarrow L} + N_{L \rightarrow F} + N_{F \rightarrow F} + N_{F \rightarrow L}$). as shown in Equation (5.7).

$$TPR = \frac{N_{F \rightarrow F}}{N_{F \rightarrow F} + N_{F \rightarrow L}} \quad (5.1)$$

$$FPR = \frac{N_{L \rightarrow F}}{N_{L \rightarrow L} + N_{L \rightarrow F}} \quad (5.2)$$

$$TNR = \frac{N_{L \rightarrow L}}{N_{L \rightarrow L} + N_{L \rightarrow F}} \quad (5.3)$$

$$FNR = \frac{N_{F \rightarrow L}}{N_{F \rightarrow F} + N_{F \rightarrow L}} \quad (5.4)$$

$$P = \frac{N_{F \rightarrow F}}{N_{L \rightarrow F} + N_{F \rightarrow F}} \quad (5.5)$$

$$R = TPR \quad (5.6)$$

$$ACC = \frac{N_{L \rightarrow L} + N_{F \rightarrow F}}{N_{L \rightarrow L} + N_{L \rightarrow F} + N_{F \rightarrow F} + N_{F \rightarrow L}} \quad (5.7)$$

5.2. Feature Selection

We perform feature selection for two times using feature importance on Random Forest (RF). On both datasets, we extract 21 NAN characters from each URL if presence. Then, we perform feature selection based on feature importance. Figure 5.1 shows the importance of features. We then select the top 10 features as an input to entropy. After feature selection of NAN, we input top 10 features to compute entropy scores of a URL in feature extraction.

Figure 5.1. Feature Importance Score

5.3. Experiment with Balanced and Imbalanced Datasets

After we selected NAN characters as mentioned above, we extract the 12 features described in Table 4.4. Then, we first perform classifications with three classifiers: Gaussian naïve bayes (GNB), support vector machine (SVM) and RF without adjusting any parameters to choose the best fit model. In Table 5.1, we show previous features (Prev) along with our proposed feature (entropy of NAN – Entropy). Note that those previous features are applied in different works, not just in a single one. We compare

experimental evaluation with or without *entropy*. Precision, recall, ACC and ROC_AUC are presented in percentage (%). RF outperforms two other classifiers on both balanced and imbalanced datasets with 89.43% and 72.64% of ROC_AUC score (Receiver Operating Characteristics) respectively. We measure the ROC_AUC score instead of accuracy since the class of highly imbalanced data tends to be biased.

Table 5.1. Evaluation Results for three Classifiers on balanced and imbalanced datasets using previous and entropy features

Classifier	Datasets	Features	Prec	R	TPR	FPR	FNR	TNR	ACC	ROC_AUC
SVM	Imbalanced	Prev+Entropy	0.94	0.67	0.66	0.34	0.33	0.66	93.08	66.28
	Balanced		0.80	0.80	0.80	0.20	0.20	0.80	79.63	79.56
GNB	Imbalanced	Prev+Entropy	0.95	0.64	0.64	0.36	0.36	0.64	92.74	64.20
	Balanced		0.81	0.72	0.72	0.28	0.28	0.72	71.33	71.91
RF	Imbalanced	Prev+Entropy	0.92	0.73	0.73	0.27	0.27	0.73	94.02	72.64
	Balanced		0.89	0.89	0.89	0.11	0.11	0.89	89.41	89.43

Table 5.2. Comparison of Without (Prev) and With (Prev+Entropy) entropy

Classifier	Datasets	Features	Prec	R	TPR	FPR	FNR	TNR	ACC	ROC_AUC
SVM	Imbalanced	<i>Prev+Entropy</i>	0.94	0.66	0.66	0.34	0.33	0.66	93.08	66.28
		Prev	0.94	0.65	0.65	0.34	0.34	0.65	92.98	65.29
	Balanced	<i>Prev+Entropy</i>	0.80	0.80	0.80	0.20	0.20	0.80	79.63	79.56
		Prev	0.80	0.80	0.80	0.20	0.20	0.80	79.50	79.75
GNB	Imbalanced	<i>Prev+Entropy</i>	0.95	0.64	0.64	0.36	0.36	0.64	92.74	64.20
		Prev	0.94	0.64	0.64	0.36	0.36	0.64	92.65	64.18
	Balanced	<i>Prev+Entropy</i>	0.81	0.72	0.72	0.28	0.28	0.72	71.33	71.91
		Prev	0.82	0.71	0.71	0.29	0.29	0.71	72.00	71.30
RF	Imbalanced	<i>Prev+Entropy</i>	0.92	0.73	0.73	0.27	0.27	0.73	94.02	72.64
		Prev	0.90	0.66	0.66	0.34	0.34	0.66	93.22	66.31
	Balanced	<i>Prev+Entropy</i>	0.89	0.89	0.89	0.11	0.11	0.89	89.41	89.43
		Prev	0.82	0.80	0.80	0.11	0.11	0.89	79.50	79.49

Then we compare the evaluation results of ROC_AUC with and without our *proposed entropy* feature in Table 5.2. As RF outperforms the two other classifiers, we focus only on RF for more hyperparameter tuning. To be noted that, our results are measured based on testing datasets of 10-fold cross validation. In the above Table 5.2, we measured experimental evaluations based on 10-fold cross validations without using hyperparameter tuning to select a better model. However, in the following section 5.4, we perform hyperparameter tuning with RF.

5.4. Comparison between Previous and Current Evaluations with Hyperparameter Tuning

As for hyperparameter tuning, we tune parameters on RF as described below. We performed GridSearchCV for parameter tuning to generate a better outcome. GridSearchCV evaluates all combinations we define along with cross validation.

```
param_grid = {'n_estimators':
[100,110,120,130,140,150,160,170,180,190,200,300,400,500,600,700],
'criterion': ['gini', 'entropy'],
'bootstrap': [true, false] }

-----

Number of combinations = 16*2*2 = 64

CV = 10-Fold
```

We show the comparison results of with and without proposed entropy of NAN characters in Table 5.3. To note that, we refit Grid Search with ROC_AUC for a fair classification to avoid bias. The evaluation results mentioned in the table are measured only for testing datasets after 10-fold cross validation. The following Figure 5.2 shows our experimental results after hyperparameter tuning.

Pseudo Code :

```
clf = RandomForestClassifier()
skf = StratifiedKFold(n_splits=10)
param_grid =
{'n_estimators':
[100,110,120,130,140,150,160,170,180,190,200,300,400,500,600,700],
'criterion': ['gini', 'entropy'],
'bootstrap': [true, false] }
grid_search = GridSearchCV(clf, param_grid, scoring=scorers,
refit=refit_score,cv=skf, return_train_score=true, n_jobs=-1)
```

Figure 5.2. Experimental Evaluation of ROC_AUC on balanced and imbalanced datasets

In GridSearchCV, all the combinations we defined are performed with 10-fold cross validation on RF classifier. We have 10 splits (split0-split9) of testing data per each combination (64 in total) and we measure mean ROC_AUC, F1, ACC of all the splits. Thus, our final experimental evaluation is achieved by the best mean test scores of all the combinations.

Table 5.3. Comparison between Previous and Current Evaluations after Hyperparameter Tuning

Dataset	Features	Hyper Parameter Tuning (RF)			Scorers	Refit By	Best Parameter	Prec*	R*	TPR*	FPR*	FNR*	TNR*	ACC*	ROC_AUC*	F1*
		n_estimator	criterion	boot strap												
Balanced	Prev+Entropy	[100,110,120,130,140,150,160,170,180,190,200,300,400,500,600,700]	['gini', 'entropy']	[True, False]	Precision Recall ACC ROC_AUC F1	ROC_AUC	'bootstrap': True, 'criterion': 'entropy', 'n_estimators': 500	0.90	0.90	0.90	0.10	0.10	0.90	89.82	96.20	89.82
	Prev						'bootstrap': True, 'criterion': 'gini', 'n_estimators': 170	0.83	0.80	0.81	0.19	0.19	0.81	80.68	87.51	80.28
Imbalanced	Prev+Entropy						'bootstrap': True, 'criterion': 'gini', 'n_estimators': 600	0.94	0.94	0.74	0.26	0.26	0.74	94.05	89.31	93.20
	Prev						'bootstrap': True, 'criterion': 'gini', 'n_estimators': 160	0.93	0.93	0.66	0.35	0.35	0.66	92.94	84.31	91.36

As we mentioned before, our ROC_AUC score shows better performance with parameter tuning and results in ROC_AUC score of 89.31% with our contributed feature 'entropy' while the performance without entropy results 84.31% on imbalanced datasets. Moreover, it also works well on balanced datasets generating 96.20% of ROC_AUC with entropy winning over 87.51% without it.

*Scorings are measured by mean value i.e. mean_test_precision, mean_test_recall and so on because we performed 10-fold cross validation and we showed our results based on test set.

5.5. Comparison between Feature Sets based on Frequency Probability and Entropy

We also measure probability distribution of frequencies of special characters as separate features without using Entropy. As for the experiment, we the same hyperparameter tuning mentioned in Table 5.3. Then, we compare evaluation result of ROC_AUC with entropy of NAN shown in the Table 5.4.

Table 5.4. Comparison between Feature Set1 (Percentage of NAN) and Feature Set2 (Entropy of NAN)

Dataset	Name	Features	No of Features	ROC_AUC	Time (seconds)	Time difference (seconds)
Imbalanced D2	Feature Set FS1	IP address, exe, sensitive word, '/' redirect, internal link, age of domain, port number, free hosting Percentage of ('#' count, '@' count, '-' count, '.' count, '\$' count, '*' count, '[' count, '(' count, '{' count, ']' count, '}' count, ')' count, '+' count, ';' count, '~' count, ':' count, '"' count, '/' count, '%' count, '?' count, ',' count, '=' count, '&' count, '!' count, '_' count), total nan count	34	90.55	4870	1293
	Feature Set FS2	IP address, exe, sensitive word, '/' redirect, internal link, age of domain, port number, '-' count, '@' count, '.' count Entropy (nan) , free hosting	12	89.31	3577	
Balanced D1	Feature Set FS1	IP address, exe, sensitive word, '/' redirect, internal link, age of domain, port number, free hosting Percentage of ('#' count, '@' count, '-' count, '.' count, '\$' count, '*' count, '[' count, '(' count, '{' count, ']' count, '}' count, ')' count, '+' count, ';' count, '~' count, ':' count, '"' count, '/' count, '%' count, '?' count, ',' count, '=' count, '&' count, '!' count, '_' count), total nan count	34	96.2	381	115
	Feature Set FS2	IP address, exe, sensitive word, '/' redirect, internal link, age of domain, port number, '-' count, '@' count, '.' count Entropy (nan) , free hosting	12	96.2	266	

To compare with our proposed entropy of NAN characters, we first measure percentage of individual 25 NAN characters and total NAN count. We represent them as 26 features. And we combine them with other 8 features. Thus, we get 34 features in total, called as FS1. In FS2, we have a total of 12 features including entropy. Our experiments show that ROC_AUC results are nearly the same because entropy is also based on the frequentist probability distribution between phishing and legitimate websites. We achieved 90.55% of ROC_AUC for FS1 on imbalanced dataset compared to 89.31% of ROC_AUC with FS2.

However, when we measure processing time of individual feature set on Random Forest model fitting using hyperparameter tuning on imbalanced dataset, FS2 is 1293 seconds (~21 mins) faster than FS1. On balanced dataset, although we achieve same 96.2% of ROC_AUC for FS1 and FS2, FS2 performed 115 seconds (~2 mins) faster than FS1.

5.6. Summary and Discussion

In our research, we observed from perspectives; (i) dataset perspective, and (ii) feature perspective.

From the dataset perspective, we performed the detection on balanced and imbalanced data, in which the majority of data is legitimate sites. Our assumption is that in the real-world, we cannot expect the same or not even close amount of phishing datasets as legitimate ones. Thus, we want to figure out if it works well on imbalanced datasets.

From the feature perspective, we contributed entropy feature for distribution of NAN characters in a URL, which makes our classification have high performance in terms of accuracy.

In addition to these perspectives, we also test on three different classifiers and later perform parameter tuning for a better and robust system. We confirm that with 96.20% of ROC_AUC on balanced and 89.31% on imbalanced datasets, outperforming 87.51% and 84.31% without entropy features respectively.

However, we still have a problem of high FPR. From the feature perspective, we use DMOZ for legitimate dataset, in which, as for the number of subdomains, legitimate websites have fewer or no information at all, while phishing sites do. It makes it difficult to retrieve similar patterns of legitimate URLs, giving high FPR. To address this, we believe it is better to target domain name-based features—instead of entire URL—to extract characteristics of domain name and current page content.

CHAPTER 6

CONCLUSIONS

Cyber phishing is a theft of personal information in which phishers mimic a legitimate website so that users surrender sensitive information. Phishers lure users, as if they are reliable, to access information to break into the victims' system. Phishing detection becomes a crucial research area as the number of phishing attacks grows along with e-commerce and internet transactions. In our research, we focus on URL (Uniform Resource Locator) -based phishing detection techniques testing our proposed feature's performance on balanced and imbalanced datasets. We consider the URL to be a significant criterium to prevent the attacks because of the fact that phishers creating fake websites with less information as possible. To overcome problem of retrieving no or less information of a webpage for detection, URL-based approach is a solution. Because we consider that NAN characters distribution highly impact on phishing URLs, we target our proposed feature – entropy of NAN in our work.

In our work, we performed feature selection of NAN. Our contribution is to propose a new entropy feature for NAN characters and compare with the previously used features. We confirmed that our proposed feature outperforms previous features and we also measured performance based on ROC_AUC score for a more robust phishing detection system and our work achieved ROC_AUC score of 89.31% and 96.20% on imbalanced and balanced datasets respectively.

Since attackers always come up with various techniques, detection has become a primary concern of developers. Only with less information of phishing webpage, we focus on URL-based detection. We have a problem of high FPR over 20% since the number of features we used are insufficient. However, our objective is to improve performance if our proposed feature is applied on any URL-based detection system combined with other features. Thus, we performed comparison to show how evaluation changed. We hope that applying our proposed feature brings a small help in any URL-based detection techniques for a better detection system.

ACKNOWLEDGEMENTS

First of all, I would like to give sincere thanks to my research supervisor Professor Hayato Yamana of Department of Computer Science and Communications Engineering, Graduate School of Fundamental Science and Engineering, Waseda University not only for his guidance, but also for giving me way of thinking.

I would like to show my gratitude to the lecturers of Department of Computer Science and Communications Engineering, Waseda University. Thanks to them, I could extend my knowledge, which is crucial for research.

Moreover, I would like to genuinely say thanks to my seniors for giving me such precious advices and never-ending guidelines throughout the whole two years. I should not miss my lab-mates and friends, who support me both physically and mentally.

Nevertheless, thank you mom and dad, and also my lovingly sisters. Thanks to your support, love and care, I did not give up and I have come this far.

REFERENCES

- [1] Internet Growth Statistics by Internet World Stats, “Today’s road to e-commerce and global trade internet technology reports”, available at “<https://www.internetworldstats.com/emarketing.htm>,” [Jun. 26, 2019]
- [2] K. L. Chiew, K. Yong, and C. L. Tan, “A survey of phishing attacks: their types, vectors and technical approaches,” in *Journal of Expert Systems with Applications*, vol.106, pp.1-20, 2018.
- [3] Anti-phishing Working Group APWG, “Unifying the global response to cybercrime”, in *Phishing activity trends report, 4th Quarter 2018*, available at “https://docs.apwg.org/reports/apwg_trends_report_q4_2018.pdf,” [Mar. 4, 2019]
- [4] Anti-phishing Working Group APWG, “APWG Report of Criminal Innovation Ramps Up with Phishing Attack in 2018 by Business Wire,” available at “<https://www.businesswire.com/news/home/20181212005652/en/APWGReport-Criminal-Innovation-Ramps-Phishing-Attacks>,” Dec. 12, 2018 [Jan. 8, 2019]
- [5] Phishlabs, “Hacking the human,” in *2018 Phishing trends and intelligence report*, available at “https://info.phishlabs.com/hubfs/2018%20PTI%20Report/PhishLabs%20Trend%20Report_2018-digital.pdf,” [2018]
- [6] B. B. Gupta, N. Arachchilage, and K. Psannis, “Defending against phishing attacks: taxonomy of methods, current issues and future directions,” in *Journal of Telecommunication Systems*, vol.67, no.2, pp.247-267, 2018, available at DOI: “<https://doi.org/10.1007/s11235-017-0334-z>.”
- [7] D. R. Patil, and J. B. Patil, “Malicious URLs detection using decision tree classifiers and majority voting technique,” in *Journal of Cybernetics and Information Technologies*, vol.18, no.1, pp.11-29, 2018.
- [8] H. Yuan, X. Chen, Y. Li, Z. Yang, and W. Liu, “Detecting phishing websites and targets based on URLs and webpage links,” in *the 24th International Conference on Pattern Recognition (ICPR)*, pp. 3669-3674, 2018.
- [9] M. Khonji, Y. Iraqi, and A. Jones, “Phishing detection: a literature survey,” in *IEEE Communications Surveys & Tutorials*, vol.15, no.4, pp.2091-2021, 2013.
- [10] G. Varshney and M. Misra and P. K. Atrey, “A survey and classification of web phishing detection schemes,” in *Security and Communication Networks*, vol.9, 2016.
- [11] D. Sahoo, C. Liu, and S. Hoi, “Malicious url detection using machine learning: A survey,” in *Computing Research Repository (CoRR)*, 2017.
- [12] A. Oest, Y. Safei, A. Doupe, G. J. Ahn, B. Wardman, and G. Warner, “Inside a phisher’s mind: understanding the anti-phishing eco system through phishing kit analysis,” *Proc. 2018 APWG Symposium on Electronic Crime Research (eCrime)*, 2018.
- [13] J. Kang and D. Lee, “Advanced white list approach for preventing access to phishing sites,” in *Proc. International Conference on Convergence Information Technology (ICCIT 2007)*, pp.491-496, 2007.
- [14] Y. Cao, W. Han, and Y. Le, “Anti-phishing based on automated individual white-list,” *Proc. the 4th ACM Workshop on Digital Identity Management*, pp.51–60, 2008.
- [15] Safe Browsing API – Google Developer, available at “<https://developers.google.com/safe-browsing/>”.
- [16] M. Sharifi, and S. H. Siadati, “A phishing sites blacklist generator,” in the

- IEEE/ACS 2008 International Conference on Computer Systems and Applications, pp. 840-843, 2008.
- [17] P. Prakash, M. Kumar, R. R. Kompella and M. Gupta, "PhishNet: Predictive Blacklisting to Detect Phishing Attacks", in Proc. IEEE 2010 INFOCOM, pp.1-5, 2010.
 - [18] N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh, and J. C. Mitchell, "Client-side defense against web-based identity theft," in report of the Network and Distributed System Security Symposium (NDSS), 2004.
 - [19] Y. Joshi, S. Saklikar and D. Das and S. Saha, "PhishGuard: A browser plugin for protection from phishing," in the IEEE 2008 2nd International Conference of Internet Multimedia Services Architecture and Applications (IMSAA), 2008.
 - [20] Y. Zhang, J. I. Hong, and L. F. Cranor, "Cantina: a content-based approach to detecting phishing web sites," Proc. the 16th International Conference on World Wide Web, pp.639-648, 2007.
 - [21] G. Xiang, J. Hong, C. P. Rose, and L. Cranor, "Cantina+: a feature-rich machine learning framework or detecting phishing web sites," in the ACM Transactions on Information and System Security (TISSEC), vol.14, no.2, pp.21:1-21:28, 2011.
 - [22] M. Moghimi, and A. Y. Varjani, "New rule-based phishing detection method," in the Journal of *Expert Systems Application*, vol.53, pp.231–242, 2016.
 - [23] G. Xiang, and J. I. Hong, "A hybrid phish detection approach by identity discovery and keywords retrieval," Proc. the 18th International Conference on World Wide Web, pp.571-580, 2009.
 - [24] V. Ramanathan, and H. Wechsler, "Phishing website detection using latent dirichlet allocation and adaboost," in 2012 IEEE International Conference on Intelligence and Security Informatics (ISI), pp.102-107, 2012.
 - [25] M. Aburrous, and A. Khelifi, "Phishing detection plug-in tool-bar using intelligent fuzzy-classification mining techniques," in the International Journal of Soft Computing and Software Engineering [JSCSE], vol.3, no.3, pp.54-61, 2013.
 - [26] L. Wenyin, G. Huang, L. Xiao Yue, Z. Min, and X. Deng, "Detection of phishing webpages based on visual similarity," in Special interest tracks and posters of the 14th International Conference on World Wide Web, pp. 1060-1061, 2005.
 - [27] Y. Fu, L. Wenyin, and X. Deng, "Detecting phishing web pages with visual similarity assessment based on earth mover's distance (EMD)," in IEEE Transactions on Dependable and Secure Computing, vol. 3, no. 4, pp. 301-311, 2006
 - [28] T. Chen, T. Stepan, S. Dick, and J. Miller, "An anti-phishing system employing diffused information," in Journal of ACM Transactions on Information and System Security (TISSEC), vol.16, no.4, pp.1-31, 2014.
 - [29] Y. Fu, W. Liu, and X. Deng, "EMD based visual similarity for detection of phishing webpages," Proc. of International Workshop on Web Document Analysis, 2005.
 - [30] L. Wenyin, G. Huang, L. Xiaoyue, X. Deng, and Z. Min, "Phishing web page detection," in the 8th International Conference on Document Analysis and Recognition (ICDAR'05), pp.560-564, 2005.
 - [31] M. Hara, A. Yamada, and Y. Miyake, "Visual similarity-based phishing detection without victim site information," in IEEE Symposium on Computational Intelligence in Cyber Security (CICS'09), pp.30-36, 2009.

- [32] G. Liu, B. Qiu, and L. Wenyin, "Automatic detection of phishing target from phishing webpage," in IEEE 20th International Conference on Pattern Recognition (ICPR), pp.4153-4156, 2010.
- [33] S. Afroz, and R. Greenstadt, "Phishzoo: an automated web phishing detection approach based on profiling and fuzzy matching," Proc. of the 5th IEEE International Conference on Semantic Computing (ICSC), 2009.
- [34] C. Kuan-Ta, C. Jau-Yuan, H. Chun-Rong, and C. Chu-Song, "Fighting phishing with discriminative keypoint features," in IEEE Internet Computing, vol.13, no.3, pp.56-63, 2009 .
- [35] M. Eric, K. Engin, and K. Christopher, "Visual-Similarity-Based Phishing Detection," in Proc. of the 4th international conference on Security and privacy in communication networks, SecureComm 2008, article no 2, pp. 1-11, 2008.
- [36] C. Teh-Chung, D. Scott, and M. James, "Detecting visually similar web pages: application to phishing detection," in ACM Transactions on Internet Technology (TOIT), vol.10, no.2, pp.1-5, 2010.
- [37] N. Sanglerdsinlapachai and A. Rungsawang, "Using domain top-page similarity feature in machine learning-based web phishing detection," in the 3rd International Conference on Knowledge Discovery and Data Mining, WKDD'10, pp.187-190, 2010.
- [38] H. Zhang, G. Liu, T. Chow, and W. Liu, "Textual and visual content-based anti-phishing: a bayesian approach," in IEEE Transactions on Neural Networks, vol.22, no.10, pp.1532-1546, 2011.
- [39] M. Maurer and D. Herzner, "Using visual website similarity for phishing detection and reporting," in ACM Conference on Extended Abstracts on Human Factors in Computing Systems, pp.1625-1630, 2012.
- [40] M. Aburrous, M. A. Hossain, K. Dahal, and F. Thabtah, "Intelligent phishing detection system for e-banking using fuzzy data mining," in Journal of Expert System with Applications, vol.37, no.12, pp.7913-7921, 2010.
- [41] M. Babagoli, M. P. Aghababa, and V. Solouk, "Heuristic nonlinear regression strategy for detecting phishing websites," in Soft Computing, vol.22, no.236, pp.1-13, 2018.
- [42] M. Dadkhah, S. Shamshirband, and A. Wahab, "A hybrid approach for phishing web site detection," in the Electronic Library, vol.34, no.6, pp.927-944, 2016.
- [43] A. Hodzic, and J. Kevric, "Comparison of machine learning techniques in phishing website classification," Proc. International Conference on Economic and Social Studies (ICESoS'16), vol.3, pp.249-256, 2016.
- [44] M. Moghimi, and A. Y. Varjani, "New rule-based phishing detection method," in Expert System with Applications, vol.53, pp.231-242, 2016.
- [45] S. Marchal, J. Francois, R. State, and T. Engel, "PhishStorm: detecting phishing with streaming analytics," in IEEE Transactions on Network and Service Management, vol.11, no.4, pp.458-471, 2014.
- [46] Z. Duo, I. Khalil, A. Khreishah, A. Al-Fuqaha, and M. Guizani, "Systematization of knowledge (SoK): a systematic review of software based web phishing detection," in IEEE Communications Surveys & Tutorials, vol.19, no.4, pp.2797-2819, 2017.
- [47] E. Kirda, and C. Kruegel, "Protecting users against phishing attacks with AntiPhish," in Proc. of the International Computer Software and Applications

- Conference (COMPSAC), vol.2, pp.517-524, 2005.
- [48] H. Le, Q. Pham, D. Sahoo, and S. C. H. Hoi, "URLNet: learning a url representation with deep learning for malicious url detection," in *Journal of Computing Research Repository (CoRR)*, 2018.
 - [49] A. K. Jain, and B. B. Gupta, "Towards detection of phishing websites on client-side using machine learning based approach," in *Journal of Telecommunication System*, vol.64, no.4, pp.687-700, 2017.
 - [50] A. C. Bahnsen, E. C. Bohorquez, S. Villegas, J. Vargas, and F. A. González, "Classifying phishing URLs using recurrent neural networks," *Proc. 2017 APWG Symposium on Electronic Crime Research (eCrime)*, pp.1-8, 2017.
 - [51] M. Zouina, and B. Outtaj, "A novel lightweight URL phishing detection system using SVM and similarity index," in *Journal of Human-centric Computing and Information Sciences*, vol.7, no.17, pp.1-13, 2017.
 - [52] S. Parekh, D. Parikh, S. Kotak, and P. S. Sankhe, "A New Method for Detection of Phishing Websites: URL Detection," in *2018 2nd International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pp.949-952, 2018
 - [53] C. Liu, L. Wang, B. Lang, and Y. Zhou, "Finding Effective Classifier for Malicious URL Detection," *Proc. the 2nd International Conference on Management Engineering, Software Engineering and Service Sciences (ICMSS)*, pp.240-244, 2018.
 - [54] A. Ankesh, G. Kshitij, M. Joel, P. Noseong, C. Tanmoy, and C. Bei-Tseng, "Phishing URL detection with oversampling based on text generative adversarial networks," *Proc of 2018 IEEE International Conference on Big Data (Big Data)*, pp.1167-1176, 2018.
 - [55] S. Shivangi, P. Debnath, K. Saieevan, and D. Annapurna, "Chrome extension for malicious URLs detection in social media applications using artificial neural networks and long short term memory networks," in *the International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 1993-1997, 2018.
 - [56] A. Vazhayil, R. Vinayakumar, and K. Soman, "Comparative study of the detection of malicious URLs using shallow and deep networks," in *the 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 2018, pp.1-6, 2018.
 - [57] PhishTank, "Out of the Net into the Tank," available at "<https://www.phishtank.com/index.php>"
 - [58] DMOZ, the Directory of Web, available at "<http://dmoz-odp.org>".
 - [59] GitHub Repository, "Phishing URL detection using machine learning," available at "<https://github.com/swagster420/Phishing-Url-Detection-Using-Machine-Learning>".
 - [60] Free Hosting Sites For Phishers, Internet : <https://blackbackhacker.blogspot.com/2012/09/free-hosting-sites-for-phishers.html>, Sept. 2012.

PUBLICATION

1. Eint Sandi Aung, Hayato Yamana, Chaw Thet Zan, “A survey of URL-based phishing detection,” in Proc. 11th Forum on Data Engineering and Information Management (DEIM), Article G2-3, pp.1-8, 2019.

APPENDIX

• Dataset Nature

We gather legitimate data from DMOZ repository and extract maximum two resource URLs from each Topic. Then, we validate and apply active URLs only on our system.

```
<Topic r:id="Top/Arts/Animation">
  <catid>423945</catid>
  <link1 r:resource="http://www.awn.com/"></link1>
  <link r:resource="http://animation.about.com/"></link>
  <link r:resource="http://www.toonhound.com/"></link>
  <link r:resource="http://www.digitalmediafx.com/Features/animationhistory.html"></link>
  <link r:resource="http://www.animated-divots.net/"></link>
</Topic>
```

Figure A.1. Xml file sample of legitimate dataset

The following figure is the sample dataset that we collected from PhishTank. We gather currently reported phishing URLs to make sure its activity.

phish_id	url	phish_detail_url	submission_time	verified	verification_time	online	target
5709890	http://www.xweb.io/free/standardbanking/standard-bank-online-banking	http://www.phishtank.com/phish_detail_	2018-06-29T09:27:46+00:00	yes	2018-06-29T09:51:44+00:00	yes	Other
5709867	http://webm24.moonfruit.com/	http://www.phishtank.com/phish_detail_	2018-06-29T09:11:54+00:00	yes	2018-06-29T09:50:59+00:00	yes	Other
5709794	http://bcptonasegura.vlabcp.m/promocion/	http://www.phishtank.com/phish_detail_	2018-06-29T06:41:55+00:00	yes	2018-06-29T06:44:29+00:00	yes	Other
5709793	http://servinok.pe/	http://www.phishtank.com/phish_detail_	2018-06-29T06:36:54+00:00	yes	2018-06-29T06:39:41+00:00	yes	Other
5709666	http://cduloures.org/aaa/ch/share/verificationAttempt.php?sf58gfd1s689sxd2sdf8ang264s9df23sd2f1n	http://www.phishtank.com/phish_detail_	2018-06-29T05:00:36+00:00	yes	2018-06-29T09:03:47+00:00	yes	Other
5709665	http://cduloures.org/aaa/ch/share/verification.php?sf58gfd1s689sxd2sdf8ang264s9df23sd2f1n495K3L	http://www.phishtank.com/phish_detail_	2018-06-29T05:00:32+00:00	yes	2018-06-29T07:58:29+00:00	yes	Other
5709664	http://estanciadapicanha.com.br/gggggg/drob/read_file/pmb.php	http://www.phishtank.com/phish_detail_	2018-06-29T05:00:24+00:00	yes	2018-06-29T08:09:39+00:00	yes	Other
5709660	http://hansaticartgallyeohraim.com/error.php	http://www.phishtank.com/phish_detail_	2018-06-29T04:59:54+00:00	yes	2018-06-29T08:12:15+00:00	yes	Other
5709659	http://hansaticartgallyeohraim.com/pass.php	http://www.phishtank.com/phish_detail_	2018-06-29T04:59:51+00:00	yes	2018-06-29T07:50:50+00:00	yes	Other
5709656	http://www.systemrazors.com/report/box/2996f645bf78c5f4a5e42636bb4f157/	http://www.phishtank.com/phish_detail_	2018-06-29T04:59:24+00:00	yes	2018-06-29T08:38:08+00:00	yes	Other
5709655	http://systemrazors.com/report/box/index.php	http://www.phishtank.com/phish_detail_	2018-06-29T04:59:23+00:00	yes	2018-06-29T07:12:04+00:00	yes	Other
5709654	http://www.systemrazors.com/report/box/6b4e8fb72380f21397d2d4e2d9cd0116f/	http://www.phishtank.com/phish_detail_	2018-06-29T04:59:22+00:00	yes	2018-06-29T09:28:36+00:00	yes	Other

Figure A.2. Csv file sample of phishing dataset

• Entropy

Let's consider `non_alphanumeric_characters` = {'.', '@', '\$', '-' }.

If the *distribution of all elements is similar*, its entropy is ~ 1.0 (can be greater than 1 depending on the number of elements in the datasets)(*high entropy*=*low purity*).

Example 1: <http://setting444545page-16mb.com>

(Ignore red part 'http://' or 'www.')

No of NAN characters : 2 in total ('-' = 1 and '.' = 1)

$$Entropy = -(0.5\log 0.5 + 0.5\log 0.5) = 1$$

Example 2: <http://setting444545page.main1.16mb.com>

(Ignore red part 'http://' or 'www.')

No of NAN characters : 3 in total ('.' = 3)

(probability = 3/3=1.0)

$$Entropy = -1\log 1 = 0$$

If entropy = 0.0, it is *low entropy i.e. high purity*.