PhishZip: A New Compression-based Algorithm for Detecting Phishing Websites

Rizka Purwanto[*‡], Arindam Pal[†‡], Alan Blair[*], Sanjay Jha[*]

[*]School of Computer Science and Engineering, University of New South Wales, Australia
[†]Data61, CSIRO, Sydney, New South Wales, Australia
[‡]Cyber Security Cooperative Research Centre, Australia

Abstract—**Phishing has grown significantly in the past few years and is predicted to further increase in the future. The dynamics of phishing introduce challenges in implementing a robust phishing detection system and selecting features which can represent phishing despite the change of attack. In this study, we propose PhishZip which is a novel phishing detection approach using a compression algorithm to perform website classification and demonstrate a systematic way to construct the word dictionaries for the compression models using word occurrence likelihood analysis. PhishZip outperforms the use of best-performing HTML-based features in past studies, with a true positive rate of 80.04%. We also propose the use of compression ratio as a novel machine learning feature which significantly improves machine learning based phishing detection over previous studies. Using compression ratios as additional features, the true positive rate significantly improves by 30.3% (from 51.47% to 81.77%), while the accuracy increases by 11.84% (from 71.20% to 83.04%).**

Index Terms—**phishing detection, web page, compression, classification**

I. Introduction

The number of phishing attacks has grown significantly in the past few years. The Anti-Phishing Working Group (APWG) recorded a significant increase of unique phishing attacks from 2014 to 2016 [1] causing considerably high financial loss ranging between $60 million and $3 billion per year in the United States [2]. The number of attacks is likely to increase in the future with the availability of phishing toolkits and algorithms which ease the process of phishing [3], [4]. The dynamics in phishing behaviours bring challenges in implementing a robust and accurate phishing detection for the long term [5].

To mitigate the negative impacts of phishing, security software providers, financial institutions, and academic researchers have studied various approaches to build an automated phishing website detection system. These methods include the use of blacklists and detecting phishing websites by investigating the website content, URL, and web-related features [6]–[10].

While most past studies in phishing detection focused on the use of machine learning algorithms, this study introduces the use of compression algorithms to perform phishing website classification. The compression-based classification process aims to predict whether a certain website is phishing or benign based on the textual information. To perform this task, we built two compression models which are optimised for phishing and benign websites respectively. Building these models requires information regarding the word distribution in phishing and benign websites, which is required to optimise the compression for each class. Using these models, we can perform classification by comparing the compression ratios of both models. We expect that a phishing optimised model should compress phishing websites better than a non-phishing optimised compression model, resulting in a higher compression ratio, and vice versa. The main tool that we use in this work is the `zlib` library, which is based on the DEFLATE compression algorithm [11]. DEFLATE is a lossless data compression algorithm used for compressing websites in HTTP [12].

To summarise, this paper makes the following contributions:

- We introduce a systematic process of selecting meaningful words which are associated with phishing and non-phishing websites by analysing the likelihood of word occurrences and calculate the optimal likelihood threshold. These set of words are used as the predefined compression dictionary for our compression models.
- We develop a tool called PhishZip which performs phishing website detection using the DEFLATE compression algorithm. To the best of our knowledge, this study is the first to use compression algorithms to perform phishing website classification. Unlike machine learning based models, performing classification by leveraging compression algorithms does not require training the models nor performing HTML parsing [13]. Thus, classification with compression algorithms is faster and simpler.
- We propose the use of compression ratio as a novel machine learning feature which is robust and easy to extract. Compression ratio measures the distance or cross-entropy between the predicted website and phishing/non-phishing website content distribution. The high compression ratio is associated with low cross-entropy, which indicates that the content distribution is similar to the common word distribution in phishing/non-phishing websites [13].

The paper is organised as follows. In section II, we provide an overview of past studies in phishing detection systems and compression based classification. Section III provides some background regarding phishing and the concept of compression-based classification. Section IV gives the system overview of PhishZip. We provide the word occurrence likelihood analysis for constructing the word dictionary of the compression models in Section V. The experimental setups,

including the evaluation methodology and diversity of our web page corpus, are presented in Section VI. We provide the performance evaluation results in Section VII and provide more analysis regarding these results in Section VIII. Finally, we wrap up with conclusions in Section IX.

II. RELATED WORKS

Several past studies in phishing website detection systems have been conducted, mainly focusing on the use of machine learning algorithms to classify phishing websites based on specific features. Whittaker et al. introduced a machine learning classifier for automatically maintaining Google's phishing blacklist using features extracted from the URL, page hosting information, and the page content [10], [14]. Meanwhile, Xiang et al. proposed CANTINA+ as a comprehensive framework for detecting phishing websites using URL-based, HTML-based, and web-based features [6]. The study by Xiang et al. extends CANTINA, which performs phishing detection based on TF-IDF information retrieval algorithm and seven other content-based heuristics, including age of domain, logo image and domain name inconsistency, and suspicious links in the HTML [8]. Xiang and Hong also studied the use of identity discovery as a hybrid phishing detection approach [7]. Meanwhile, Zhang et al. [9] proposed a content-based phishing website detection by analysing the website textual and visual content and assessing its similarity. In a recent study, Quinkert et al. [15] focused on observing the use of homograph domains to identify scamming and phishing.

In contrast, our work leverages compression algorithm to perform classification which has not been implemented for phishing detection in past studies. A number of studies have previously discussed the use of data compression algorithm to perform text classification in various areas. Marton et al. in [13] evaluated the performance of various compression-based classification methods to classify text based on topic/genre and authorship attribution. Meanwhile, Ziegelmayer and Schrader in [16] discussed the use of Prediction by Partial Matching (PPM) to perform sentiment polarity classification. Compression algorithms have also been used to perform classification of amino acid sequences of DNA as discussed in [17].

III. BACKGROUND

In this section, we provide background on phishing, compression-based classification, and a brief description of the DEFLATE compression algorithm.

A. Phishing

Phishing is a social engineering attack which aims at stealing user account credentials by impersonating legitimate organisations or institutional websites. Several motives underlie these attacks, such as financial gain or stealing identities for hiding illegal actions [18]. To perform phishing, attackers typically broadcast emails to the victims with urgent calls to a particular action, e.g. account subscription status or password reset due to a data breach, with a URL redirecting to these phishing websites. As this type of attack exploits users' vulnerabilities, finding an effective strategy to avoid phishing and reduce its impact is fairly challenging. Despite having an information system which is technically secure against password theft, attackers are still able to obtain user data when unaware end-users type their credentials into a phishing website form.

Three types of anti-phishing techniques exist to mitigate the negative impacts of phishing [19], which are detection, prevention, and revision. The detection technique is arguably the most effective as it minimises human errors, the main vulnerability exploited by phishing attackers [20]. Developing a phishing detection system requires some knowledge regarding how attackers conduct the attacks. In their literature study, Dou et al. mentioned five stages of phishing attacks as reconnaissance, weaponisation, distribution, exploitation, and exfiltration [20]. Based on this life-cycle, phishing detection systems are implemented to avoid phishing during the exploitation step, when victims receive the phishing emails. The detection systems detect phishing websites or phishing emails on the client side through a Web browser or specific anti-phishing software, or on the server side. Whenever a phishing website or email is detected, the detection system will either block user access to the suspected website, notify users through a warning that the visited website or received email is potentially malicious [21], [22].

B. Compression-Based Classification

Text categorisation tasks assign each document to one of the preset categories. Various machine learning algorithms have been used in several studies to perform text categorisation tasks, such as Naive Bayes, SVM, and deep learning methods. Machine learning based text classification typically consists of four steps: (1) word and sentence segmentation on the training files, (2) feature selection based on the word counts, (3) building a model based on a machine learning algorithm, (4) performing feature extraction on the testing files and evaluating the model on this test data [23].

Compared to the standard machine learning approach for classification, compression-based methods are relatively easy to apply and do not require further input document pre-processing or feature extraction [23]. Data compression algorithms build a model of the processed documents based on extensive statistics regarding these documents [16]. Text classification is performed by compressing the documents using compression models optimised for each class of document. After obtaining the compression results, each document is assigned to the preset category which results in the highest compression rate, indicating the best compression. From the perspective of information theory, the compression rate is related to the cross-entropy between the distribution of the class text corpus and the text distribution of the predicted document. By choosing the class model which performs the best for the predicted document, the classification is essentially performed by choosing the category whose text corpus minimises this cross-entropy [13]. In this study, we leverage a compression

algorithm as a novel approach to perform phishing website classification. Further details regarding the DEFLATE algorithm that we use are provided in the following subsection.

C. DEFLATE Algorithm

In this study, we use the DEFLATE lossless compression algorithm which is one of the compression schemes used in the HTTP standard for minimising latency by reducing the byte size transferred across the network [12]. In Web browsers, two of the three encodings defined in the HTTP specification ("Content-Encoding: gzip" and "Content-Encoding: deflate") are based on the DE-FLATE algorithm [24].

The DEFLATE algorithm mainly consists of the LZ77 algorithm and Huffman encoding. LZ77 is a dictionary-based Lempel-Ziv compression algorithm which aims to eliminate duplicate bytes by replacing recurring bytes in the data with a back-reference which points to the first occurrence of the bytes. The next step is using Huffman coding for replacing symbols with new weighted symbols depending on the frequency of occurrence. Thus, symbols which frequently show up are replaced with shorter symbol representations, and vice versa [25].

We have used the zlib data compression library which provides an abstraction of the DEFLATE compression algorithm [26]. The zlib module in Python includes functions which implement data compression and decompression using the DEFLATE algorithm [27] and provides the functionality to set a predefined zlib compression dictionary [28]. This word dictionary can help to improve the compression result and should contain a list of bytes or strings which are expected to occur frequently in the data [27]. In this study, we built two word-dictionaries, which consist of common words in phishing and non-phishing websites respectively. With these two dictionaries, we performed compression on the same document *twice* using each dictionary separately. Ideally, compressing phishing website content using the phishing dictionary should produce a more compressed output than using the non-phishing dictionary and vice versa.

IV. SYSTEM OVERVIEW

This section contains a description regarding the design of PhishZip detection system. There are two main modules in PhishZip, which are the dictionary builder module and compression model.

A. Dictionary Builder

This module aims at creating word dictionaries to build compression models optimised for phishing websites and non-phishing websites respectively. To build these dictionaries, we calculate the likelihood of words occurring in each website class (i.e. phishing and non-phishing) and selecting a minimum likelihood threshold value. The words in the dictionaries are those whose likelihood of occurring in a certain website class are greater than a certain threshold.

The word occurrence likelihood in a specific website class is proportional to the word frequency in this class; thus, words which show up more often will be assigned higher likelihood values. Meanwhile, this likelihood value should be comparable for both website classes. Thus, for an arbitrary word, if its likelihood value of occurrence in phishing websites is higher than in non-phishing websites, we should expect the word to appear more often in phishing websites, which indicates higher association to phishing websites than non-phishing websites.

To estimate the likelihood of a word w_k occurring in a certain text category $v_j \in \{phishing, non-phishing\}$ (either *'phishing'* or *'non-phishing'*), we adopt the *m*-estimate with uniform priors and set $m = |V|$ (the size of the word vocabulary V). More formally,

$$P(w_k|v_j) = \frac{n_k + 1}{n + |V|}. \tag{1}$$

Here, n is the total number of word positions in all training examples whose target value is v_j, n_k is the number of times the word w_k is found in all the n word positions, and $|V|$ is the total number of distinct words and other tokens in all examples [29].

B. Compression-Based Classification Model

To perform the classification task, we build two compression models optimised for phishing websites and non-phishing websites respectively. The phishing optimised compression model uses the phishing dictionary, while the non-phishing optimised compression model uses the non-phishing dictionary as the preset compression dictionary. These dictionaries are constructed from the dictionary builder process as discussed in the previous subsection.

To perform website classification, we compress the raw website HTML content using the phishing optimised model and non-phishing optimised model separately. There are two separate compression outputs from this process. We compare these outputs by calculating the compression ratio, which is the ratio of the original input size to the compression output size.

$$Compression\ Ratio_i = \frac{size(HTML\_content)}{size(C_i(HTML\_content))}. \tag{2}$$

Here, C_i indicates the compression model and HTML_content is the raw website content. The decision on whether the website is classified as phishing or non-phishing is based on which compression model produces the higher compression ratio, i.e. if the phishing optimised compression model produces a better compression ratio than the non-phishing optimised model, then the website is classified as phishing, and vice versa. An illustration of the compression-based classification model is shown in Figure 1.

V. WORD OCCURRENCE LIKELIHOOD ANALYSIS

The compression-based classification mainly relies on word distribution differences in each class. Thus, we performed
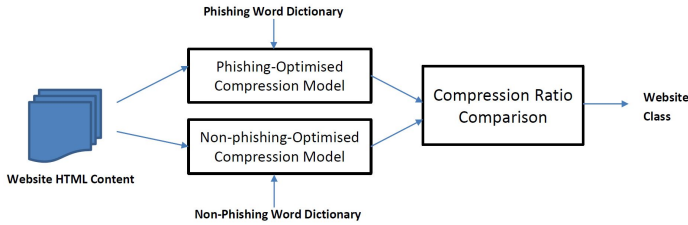
Fig. 1. PhishZip Compression-Based Classification Model

Fig. 2. Word Frequency Histogram

analysis to observe the word distribution in the website textual contents and calculate the likelihood of word occurrences in phishing and non-phishing websites. This information is useful to build the dictionary of common words in phishing and non-phishing websites for the compression models. We also performed an analysis to select the optimal likelihood threshold for constructing the common word dictionary which leads to the best classification performance. Before performing these analyses, we performed some data preprocessing to obtain the website textual contents. These processes will be described further in the following subsections.

To perform this analysis, we use a set of website HTML contents which consist of 5,000 phishing and 5,000 non-phishing websites. These website contents were manually collected by fecthing the content of phishing website URLs from PhishTank [22] and safe website URLs listed by Quantcast [6] using `curl` [30] to securely inspect the website contents without running any potentially malicious code on the browser.

As we focus on the textual information in the website, we performed data preprocessing to extract text from the website HTML content and removed common English stop words (e.g. "the", "a", "an") as specified in Natural Language Toolkit (NLTK) library [31]. After performing these data preprocessing steps, we obtained a collection of phishing and non-phishing website textual contents which were used to perform word occurrence likelihood analysis.

To observe the word distribution in both classes and how distinctive they are, we plot the frequency of words in both phishing and non-phishing text corpora. We normalised the word frequency by dividing the value by the length of phishing or non-phishing text corpus to show the frequency of word occurrence relative to the total number of words in the corpus. The histograms of word distribution of phishing and non-phishing websites are shown in Figure 2. In this histogram, we included the 100 most frequent words to show the word distribution differences more clearly. The histogram shows that the word distribution of phishing websites is typically much steeper than non-phishing websites. This intuitively means that phishing websites typically share common words and use less variety of words in the website content. Meanwhile, the use of words in non-phishing websites are more general as shown by the flatter distribution shape.

After calculating word occurrence likelihood calculation using Equation 1, likelihood threshold analysis is performed
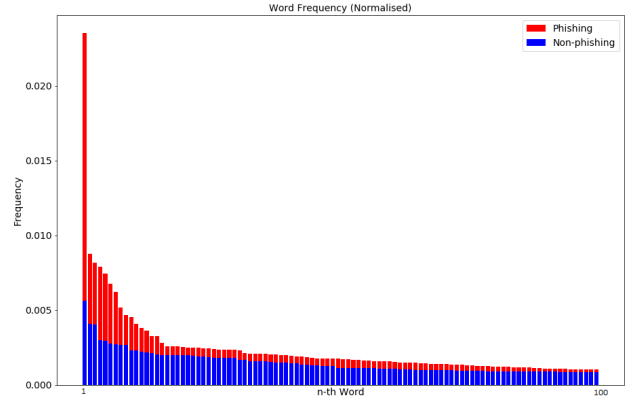
to obtain the likelihood threshold which leads to the best classification accuracy. This optimisation is done by varying the likelihood threshold and creating a hypothetical predefined dictionary using the words whose likelihood of occurrence in phishing and non-phishing websites is greater than this threshold. Two compression models are built using the phishing and non-phishing predefined word dictionaries. Afterwards, classification is performed by comparing which model results in a smaller file output size. The accuracy is calculated for each likelihood threshold.

As the word list size is enormous (536,684 unique words in the phishing text corpus and 3,668,395 unique words in the non-phishing text corpus), for practical reasons, we only stored 3,000 words with the highest likelihood in each text corpus and its likelihood values. To obtain a general idea of how many words are included in the dictionary as we vary the likelihood thresholds, we plot the distribution of the likelihood values. This plot could also help in analysing the accuracy value dynamics and fluctuations when we vary the likelihood thresholds in a specific range.

Figure 3 shows the word occurrence likelihood values of the 10th, 20th, up to the 100th percentile. The graph shows that for both tasks, the likelihood value increases exponentially. This distribution shows that the majority of word occurrence likelihood (around 80% of words in the lower-rank) ranges between 0.00001 to 0.0002. Meanwhile, there is a steep increase in likelihood values in the top 20% of words. The optimal likelihood threshold would eliminate as many words as possible, but still include words which are meaningful in providing information regarding a category.

Meanwhile, Figure 4 shows the accuracy trend as the likelihood threshold is varied. This graph shows a relatively regular pattern where the accuracy increases as the likelihood threshold increases and gradually decreases after reaching the optimal likelihood threshold value. The maximum accuracy (75.64%) is reached when the likelihood threshold is set to 0.0005. From the distribution of word occurrence likelihood (Figure 3), 0.0005 is around the 94th percentile in the likelihood data. Thus, by setting the likelihood threshold to
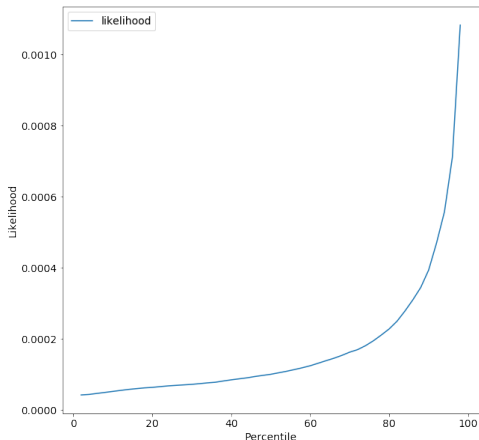
Fig. 3. Distribution of Word Occurrence Likelihood

TABLE I
SAMPLE OF WORDS IN THE DICTIONARIES

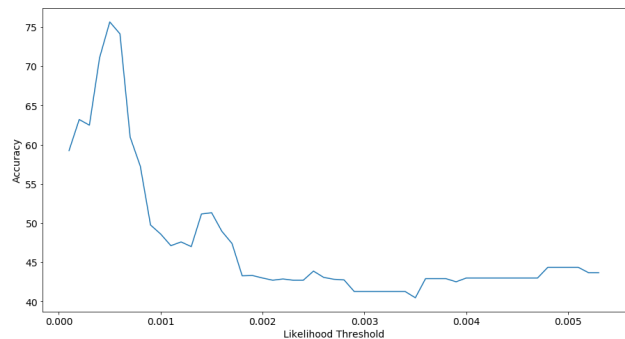| Phishing	Non-phishing
email	us
account	news
sign	get
password	view
please	free
server	best
help	shop
deleting	day

Fig. 4. Accuracy vs Likelihood Threshold

et al. in CANTINA+ [6]. Meanwhile in the second experiment, we evaluated the performance of compression ratios as machine learning features and investigated how this could improve the detection performance of CANTINA+. In these experiments, we focus on the following metrics to evaluate the performances:

- True positive rate (TPR): the ratio between the number of correctly classified phishing websites and the total number of phishing websites,
- False positive rate (FPR): the ratio between the number of misclassified non-phishing websites and the total number of non-phishing websites.
- F1-score: the harmonic mean of recall (true positive rate) and precision.

B. Web Page Corpus

The Web page corpus used to perform these performance evaluations consist of 2,045 phishing and 2,000 non-phishing raw HTML contents. To avoid classification model overfitting, this Web page collection is an entirely different website dataset used for performing word likelihood analysis and building the compression dictionaries.

To contrast, the phishing website dataset for word likelihood analysis comprises of 5,000 phishing websites chosen randomly which were reported by users to PhishTank [32] from 2016 to 2018 (665 websites in 2016, 1921 websites in 2017, and 2414 websites in 2018). On the other hand, the phishing website dataset used for the performance evaluations consists of phishing websites reported to PhishTank between January to May 2019. While we were able to collect 8,492 verified phishing websites listed by PhishTank, 6,447 of these websites have unknown target brand (labeled as 'Others' in PhishTank) and only 2,045 of these are provided with information of the legitimate target brand they attempt to resemble. To strictly observe the dataset diversity, we excluded phishing websites whose target brand name are undefined. The total number of target brands in this dataset is 92 with PayPal, Facebook, and Microsoft as the top brand targets. Further details regarding the phishing dataset diversity are provided in Table II.

Meanwhile, the non-phishing website dataset for this performance analysis consists of 2,000 non-phishing Web pages

0.0005, we are selecting the top 6% of words from our list which are associated with phishing and non-phishing classes. Setting this likelihood threshold results in a phishing word dictionary which consists of 178 words and a non-phishing word dictionary that contains 246 words. Sample of words with high likelihood values in both dictionaries are provided in Table I. We also attempted to select the words using other methods, i.e., based on the occurrence likelihood difference of phishing and non-phishing websites. However, we were unable to achieve good performances with the dictionaries constructed using this word selection method, as the method includes words with high likelihood differences and low occurrence likelihoods into the dictionaries; thus, adding words which do not frequently appear in non-phishing or phishing websites.

VI. EXPERIMENTAL SETUP

In this section, we briefly provide the evaluation methodology that we use in this study, including the performance metrics to assess the models' performances, followed by further details regarding the Web page corpus size and distribution.

A. Evaluation Methodology

We conducted two experiments in this study. In the first experiment, we examined the performance of PhishZip in detecting phishing websites and compared this to a machine learning model using HTML-based features proposed by Xiang

Brand Name	Number of Websites
Paypal	846
Facebook	262
Microsoft	125
ABSA Bank	117
RuneScape	107
ING Direct	79
eBay, Inc.	56
Delta Air Lines	40
Allegro	30
Blockchain	25

TABLE III
PHISHZIP PERFORMANCE COMPARISON TO CANTINA+

Performance Metrics	PhishZip	CANTINA+ (HTML-based features)
TPR	80.04%	51.47%
FPR	18.25%	8.92%
Accuracy	80.89%	71.20%
F1-score	80.89%	64.21%

selected at random from the list of safe websites by Quantcast [33]. Similar to our approach when building the phishing dataset, we also make sure that the non-phishing websites in this dataset are not one of the websites included in the non-phishing dataset for the word likelihood analysis.

VII. RESULTS

In this section, we provide the results of two experiments conducted in this study: the first experiment examines the performance of PhishZip as a compression based classification method for detecting phishing websites, while the second experiment investigates the performance of website compression ratio as a viable machine learning feature for phishing detection systems.

A. PhishZip Performance Evaluation

To assess the performance of PhishZip, we compress each website in the phishing and non-phishing evaluation dataset with compression models optimised for phishing and non-phishing websites respectively, then compare the compression ratio of each model. Phishing websites are expected to have higher compression ratio using phishing optimised compression model than non-phishing compression model, and vice versa.

By performing classification using compression algorithms alone, we are able to detect phishing websites with a true positive rate of 80.04%, false positive rate of 18.25%, and accuracy of 80.89%.

As a comparison, we also perform phishing website classification with features proposed by Xiang et al. in CANTINA+ [6]. We choose the features proposed in this study as they

provide comprehensive results regarding the performance of the system and each individual feature. In their study of CANTINA+, Xiang et al. investigated the performance of various URL-based features, HTML-based features, and Web-based features. The top-performing features are page in top search results, bad forms, bad action fields, and non-matching URLs. As our work focuses on website content based features, we compare the performance of PhishZip with top-performing HTML-based features only, which are bad forms, bad action fields, and non-matching URLs.

A summary of the performance evaluation results of PhishZip and the selected features in CANTINA+ is shown in Table III. We found that the machine learning model achieved an accuracy of 71.20%, low false positive rate around 8.92%, and relatively low true positive rate of roughly 51.47%. These results show that while the use of these HTML-based features still produce in a relatively descent accuracy and low false positive rate, they did not perform well in detecting phishing websites as reflected by the true positive rate. Meanwhile, as shown in Table III, PhishZip outperforms the use of HTML-based features in CANTINA+ with a true positive rate of 80.04%, false positive rate of 18.25%, and accuracy of 80.89%. Further results of each HTML-based feature are discussed as follows.

1) Bad forms: One of the best performing HTML-based features in CANTINA+ [6] is the bad form indicator, which is a binary feature set to 1 if any malicious form exist in the Web page HTML content. A malicious form exists if a Web page has all of the following:

- an HTML form,
- an <input> tag in the form,
- sensitive keywords (e.g. "password", or "credit card number") or image only (with no text) in the scope of the form,
- a non-https URL in the action field of the form or in the Web page URL (if the action form is empty).

We applied the approach above to identify bad forms in the 2,045 phishing and 2,000 non-phishing Web page corpora. Using this heuristic, we categorise the website as phishing if any bad forms exist, and vice versa. This approach gives us a true positive rate of 8.90% and false positive rate of 19.45%.

While the true negative rate is relatively descent (80.55%), plenty of the phishing websites (91.10% from the phishing Web page corpora) were misclassified as benign. This is due to the strict condition that the bad forms should have a non-https scheme URL in the action field or the Web page URL. We found that 1,209 out of 2,045 phishing websites (59.12%) in our phishing Web page corpora use HTTPS in its URL. Further, Anti-Phishing Working Group (APWG) [1] reported that there is a significant increase in the use of secure connection in phishing websites to increase its credibility.

2) Bad action fields: Another best performing features in CANTINA+ [6] is the bad action field indicator, which is a binary feature that detects if an action field is empty or a simple file name, or if it points to a different domain than the Web page

domain. Assessing the binary feature performance on our Web page corpora, this approach was able to achieve a true positive rate of 19.95% and false positive rate of 38.25%. Further, we found that 232 phishing websites in our corpora have empty or simple filename in the action field and 182 phishing websites have a domain in the action field different to the Web page domain.

3) Non-matching URLs: The other best performing HTML-based feature in CANTINA+ [6] is the non-matching URLs indicator, which is set to 1 if the percentage of highly similar URLs or the percentage of empty or ill-formed URL in the website is greater than a certain threshold. To obtain this threshold, we observed the distribution of these percentages and chose the threshold which gives the best performance in terms of the total number of correctly classified samples (true positive + true negative). Using this setting, we are able to detect phishing websites using this heuristic with a true positive rate of 47.92% and a relatively low false positive rate of 8.0%.

B. Compression Ratios as Machine Learning Features

Compression ratio is a novel machine learning feature. To the best of our knowledge, this has not been applied to detect phishing websites in past studies. Compression ratio is also relatively robust against the dynamic behaviour of phishing, as we could easily update the predefined word dictionaries and optimise the compression models correspondingly. Calculating compression ratio is relatively easy, as website browsers perform website compression frequently to improve user experience when opening websites [24]. As defined in HTTP specification, compression is one of the options in the standard for minimising latency during data transfer over the internet [12] and is a built-in feature in current Web browsers [24]. This could be incorporated through protocol option support, where no re-compression may be needed. These characteristics make compression ratio as a viable option as a machine learning feature for performing phishing detection.

1) Dataset Size: We allocated roughly 80% of our Web page corpus (1,672 phishing websites and 1,630 non-phishing websites) for training the machine learning model and around 20% (373 phishing websites and 370 non-phishing websites) to evaluate the performance of the machine learning model. We apply temporal-split between training and testing dataset, following past studies that recommend this approach over a randomised cross-validation evaluation which introduces performance overestimate due to the risk of training future data and testing on the past [34], [35]. Our phishing training dataset consists of phishing websites reported to PhishTank in January-April 2019 targeting 86 unique brands, while the testing dataset are phishing websites reported to PhishTank in May 2019. Meanwhile, we collected the non-phishing Web page corpus in a single time point rather than collecting at each time point following the methodology of CANTINA+ [6], which is based on the study by Fetterly et al. discovering that Web page content is relatively stable over time [36].

TABLE IV
MACHINE LEARNING MODEL PERFORMANCE USING COMPRESSION
RATIOS AS FEATURES

Classifier	Performance			
	TPR	*FPR*	*Accuracy*	*F1-score*
Logistic regression	75.34%	17.03%	79.14%	78.38%
SVM	78.82%	16.76%	81.02%	80.66%
K-nearest neighbours	77.48%	12.43%	82.50%	81.64%
Decision tree	75.34%	11.89%	81.70%	80.52%
Random forest	**78.28%**	**12.97%**	**82.64%**	**81.91%**
Neural network	79.89%	30.27%	74.83%	76.11%
Naive Bayes	42.09%	8.38%	66.76%	55.97%

2) Model Training: We used several machine learning algorithms to perform classification, including logistic regression, support vector machine, k-nearest neighbours, decision tree, random forest, neural network, and Naive Bayes. We used the `scikit-learn` library in Python which provides the implementation of these machine learning algorithms. Model parameter tuning is performed by using grid search to find the model parameters which produce the best performance. We used 3-fold cross-validation method to validate the model performance during the model training process which is the default cross-validation method of `GridSearchCV` in the `scikit-learn` library.

3) Performance Evaluation: Using solely compression ratios as features of our machine learning model, the best performance is achieved by the model trained using random forest algorithm with an F1-score of 81.91%. The random forest algorithm works well in this classification problem as it combines different individual models which lead to less bias and less variance. The neural network is able to achieve the best true positive rate of 79.89%; however, this model suffers from a high false positive rate. The performances of the machine learning models are provided in Table IV.

We also assessed the performance of the use of compression ratios and HTML-based features in CANTINA+ to perform phishing website detection. Using both these features, the best performing model achieved a true positive rate of 81.77%, false positive rate of 15.68%, and accuracy of 83.04%. This shows that the combination of compression ratios and HTML-based features could enhance the performance of the phishing website detection with improvements in the true positive rate of 58.87% over the model using HTML-based features in CANTINA+. While there is a compromise in an increase of false positive rate, the use of this combination of feature has also helped in improving the accuracy and F1-score as well. A comparison of the best performances using various combination of features is provided in Table V.

We also attempted to assess the performance using an imbalanced testing dataset with a class ratio of non-phishing to phishing websites of 100:1, as suggested in past studies in phishing detection systems [10], [20]. During this evaluation, we down-sampled the phishing dataset by selecting a few

Performance metrics	Features		
	Compression ratios	HTML-based features	Compression ratios & HTML-based features
TPR	78.28%	51.47%	81.77%
FPR	12.97%	8.92%	15.68%
Accuracy	82.64%	71.20%	83.04%
F1-score	81.91%	64.21%	82.88%

Performance metrics	Features		
	Compression ratios	HTML-based features	Compression ratios & HTML-based features
TPR	77.25%	51.00%	82.50%
FPR	12.97%	8.92%	15.68%
Accuracy	86.92%	90.65%	84.30%
F1-score	11.17%	10.34%	10.08%

phishing samples in random and iterate the testing process *n* number of times using different random phishing samples. In this experiment, we chose $n = 100$ iterations and calculated the average true positive rate, false positive rate, accuracy, and F1-score. A summary of the machine learning models performances in this experiment is provided in Table VI.

Using HTML-based features in CANTINA+, the best model was able to achieve an accuracy of 90.65%. However, accuracy would not be the best metric to measure the model performance in an imbalanced dataset. As shown in Table VI, the model only achieved a true positive rate of 51%, which means around half of the phishing websites in the dataset are not detected. As the class ratio of non-phishing to phishing is 100:1, misclassifying one phishing website would still lead to 99% accuracy. Thus, in this scenario, we measure the model performance based on the true positive rate. Combining the HTML-based features with compression ratios, the best model was able to achieve a significant increase of true positive rate, from 51% to 82.50%.

VIII. DISCUSSION

In this section, we discuss the model evaluation results in Section VII, including some limitations of the approach and possible future works.

A. Performance Evaluation

In this study, we propose the use of compression algorithm to perform phishing website classification. The compression ratio measures the cross entropy between the distribution of the website content we are trying to classify and the word distribution of phishing or non-phishing websites. As shown in Table III, PhishZip outperforms the performance of top-performing HTML-based features proposed in CANTINA+ [6],

achieving a true positive rate of 80.04%. The use of compression ratios as an additional feature to add with CANTINA+ features is able to significantly improve the machine learning model performance as shown in Table V, achieving a true positive rate of 81.77% and accuracy of 83.04%. As shown in Table V, using solely compression ratios, we are also able to detect phishing websites with a comparably good true positive rate of 78.28% and an accuracy of 82.64%. These results show that compression ratio is a viable option as a machine learning feature for detecting phishing websites and as an additional feature to improve the performance of existing phishing website detection systems.

B. Limitations and Future Works

There are some limitations in the current PhishZip implementation. First, it may not perform well when classifying websites which are purely made up of images without any text to analyse. Therefore, the system can possibly be bypassed when attackers use images only on the phishing website pages to imitate the design of the targeted website. On the other hand, legitimate websites rarely contain solely images and no text [6]. This characteristic would be useful to distinguish legitimate websites from phishing websites. One possible solution is by training classification models to detect potential phishing Web pages with no sufficient text and mostly filled up with images.

PhishZip also suffers when dealing with websites with content obfuscation, e.g. using an external file to load the website content, as well as cross site scripting (XSS) attacks and the use of `iframe`. However, we believe that existing methods, e.g. cookie protection and DOM or HTML sanitizer, would complement PhishZip and provide solutions to mitigate these vulnerabilities.

There are also possibilities during which attackers intentionally avoid PhishZip detection by selecting words which are not included in the compression dictionaries or using words with low likelihood of showing up in phishing websites. One possible approach to avoid this scenario is by keeping the likelihoods and dictionaries confidential. This is similar to the case of machine learning models, where the model architecture, weights, or parameters should be kept secure to avoid adversarial attacks.

For future works, there is also an opportunity to improve the performance of PhishZip by making use of website HTML structure information. As discussed by Cui et al. [37], phishing websites have similar HTML DOMs and are often variations of other phishing websites. This information might be useful to optimise the compression models better or select the best compression algorithms which will be well-suited to compress websites with slight variation of the HTML DOM trees.

IX. CONCLUSION

In this study, we propose PhishZip as a novel approach to perform phishing website classification using a dictionary-based compression algorithm. This method leverages word dictionaries constructed by analysing the word occurrence

likelihood, which is also demonstrated in this work. PhishZip outperforms the use of best-performing HTML-based features proposed in past studies with a true positive rate of 80.04%. We also introduced the use of compression ratio as a novel machine learning feature which has shown to significantly improve past studies in machine learning based phishing detection systems. Using compression ratios as additional features, the true positive rate has significantly improved by around 30.3%, from 51.47% to 81.77%, while the accuracy increased roughly by 11.84%, from 71.20% to 83.04%.

References

[1] APWG, *APWG Phishing Trends Reports*. Anti Phishing Working Group, 2016.

[2] J. Hong, "The state of phishing attacks," *Commun. ACM*, vol. 55, no. 1, pp. 74–81, Jan. 2012. [Online]. Available: http://doi.acm.org/10.1145/2063176.2063197

[3] A. K. Sood and S. Zeadally, "A taxonomy of domain-generation algorithms," *IEEE Security & Privacy*, vol. 14, no. 4, pp. 46–53, 2016.

[4] T. Holz, C. Gorecki, K. Rieck, and F. C. Freiling, "Measuring and detecting fast-flux service networks." in *NDSS*, 2008.

[5] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Identifying suspicious urls: An application of large-scale online learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: ACM, 2009, pp. 681–688. [Online]. Available: http://doi.acm.org/10.1145/1553374.1553462

[6] G. Xiang, J. Hong, C. P. Rose, and L. Cranor, "CANTINA+: A Feature-Rich Machine Learning Framework for Detecting Phishing Web Sites," *ACM Transactions on Information and System Security*, vol. 14, no. 2, pp. 1–28, sep 2011. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2019599.2019606

[7] G. Xiang and J. I. Hong, "A hybrid phish detection approach by identity discovery and keywords retrieval," in *Proceedings of the 18th international conference on World wide web*. ACM, 2009, pp. 571–580.

[8] Y. Zhang, J. I. Hong, and L. F. Cranor, "Cantina: A content-based approach to detecting phishing web sites," in *Proceedings of the 16th International Conference on World Wide Web*, ser. WWW '07. New York, NY, USA: ACM, 2007, pp. 639–648. [Online]. Available: http://doi.acm.org/10.1145/1242572.1242659

[9] H. Zhang, G. Liu, T. W. Chow, and W. Liu, "Textual and visual content-based anti-phishing: a bayesian approach," *IEEE Transactions on Neural Networks*, vol. 22, no. 10, pp. 1532–1546, 2011.

[10] C. Whittaker, B. Ryner, and M. Nazif, "Large-scale automatic classification of phishing pages," in *NDSS '10*, 2010. [Online]. Available: http://www.isoc.org/isoc/conferences/ndss/10/pdf/08.pdf

[11] P. Deutsch, "Deflate compressed data format specification version 1.3," United States, 1996.

[12] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, "Hypertext transfer protocol – http/1.1," United States, 1999.

[13] Y. Marton, N. Wu, and L. Hellerstein, "On compression-based text classification," in *European Conference on Information Retrieval*. Springer, 2005, pp. 300–314.

[14] C. N. Gutierrez, T. Kim, R. D. Corte, J. Avery, D. Goldwasser, M. Cinque, and S. Bagchi, "Learning from the ones that got away: Detecting new forms of phishing attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 6, pp. 988–1001, Nov.-Dec. 2018. [Online]. Available: doi.ieeecomputersociety.org/10.1109/TDSC.2018.2864993

[15] F. Quinkert, T. Lauinger, W. Robertson, E. Kirda, and T. Holz, "It's not what it looks like: Measuring attacks and defensive registrations of homograph domains," in *2019 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2019, pp. 259–267.

[16] D. Ziegelmayer and R. Schrader, "Sentiment polarity classification using statistical data compression models," in *2012 IEEE 12th international conference on data mining workshops*. IEEE, 2012, pp. 731–738.

[17] S. Chiba, K. Sugawara, and T. Watanabe, "Classification and function estimation of protein by using data compression and genetic algorithms," in *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546)*, vol. 2. IEEE, 2001, pp. 839–844.

[18] M. Khonji, Y. Iraqi, and A. Jones, "Phishing detection: A literature survey," *IEEE Communications Surveys Tutorials*, vol. 15, no. 4, pp. 2091–2121, Fourth 2013.

[19] S. Abu-Nimeh, D. Nappa, X. Wang, and S. Nair, "A comparison of machine learning techniques for phishing detection," in *Proceedings of the Anti-phishing Working Groups 2Nd Annual eCrime Researchers Summit*, ser. eCrime '07. New York, NY, USA: ACM, 2007, pp. 60–69. [Online]. Available: http://doi.acm.org/10.1145/1299015.1299021

[20] Z. Dou, I. Khalil, A. Khreishah, A. Al-Fuqaha, and M. Guizani, "Systematization of Knowledge (SoK): A Systematic Review of Software-Based Web Phishing Detection," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2797–2819, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/8036198/

[21] G. Varshney, M. Misra, and P. K. Atrey, "A survey and classification of web phishing detection schemes," *Security and Communication Networks*, vol. 9, no. 18, pp. 6266–6284, 2016.

[22] I. Fette, N. Sadeh, and A. Tomasic, "Learning to detect phishing emails," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 649–656.

[23] W. J. Teahan and D. J. Harper, "Using compression-based language models for text categorization," in *Language modeling for information retrieval*. Springer, 2003, pp. 141–165.

[24] "Compressing the Web," https://blogs.msdn.microsoft.com/ieinternals/2014/10/21/compressing-the-web/, note = Accessed: 2019-06-024.

[25] A. Feldspar, "An Explanation of the Deflate Algorithm," https://zlib.net/feldspar.html, accessed: 2019-06-24.

[26] "zlib," https://www.nltk.org/, accessed: 2019-06-24.

[27] "Compression compatible with gzip," https://docs.python.org/3.6/library/zlib.html, accessed: 2019-06-24.

[28] P. Deutsch and J.-L. Gailly, "Zlib compressed data format specification version 3.3," United States, 1996.

[29] T. M. Mitchell, *Machine Learning*, 1st ed. New York, NY, USA: McGraw-Hill, Inc., 1997.

[30] "curl," https://curl.haxx.se/, accessed: 2018-09-30.

[31] "Natural Language Toolkit," http://www.zlib.net, accessed: 2019-02-21.

[32] "PhishTank: An Anti-Phishing Site," https://www.phishtank.com/, accessed: 2018-10-1.

[33] "Quantcast: Top International Websites & Ranking," https://www.quantcast.com/top-sites/, accessed: 2018-10-1.

[34] G. Ho, A. Cidon, L. Gavish, M. Schweighauser, V. Paxson, S. Savage, G. M. Voelker, and D. Wagner, "Detecting and characterizing lateral phishing at scale," in *28th {USENIX} Security Symposium ({USENIX} Security 19)*, 2019, pp. 1273–1290.

[35] F. Pendlebury, F. Pierazzi, R. Jordaney, J. Kinder, and L. Cavallaro, "{TESSERACT}: Eliminating experimental bias in malware classification across space and time," in *28th {USENIX} Security Symposium ({USENIX} Security 19)*, 2019, pp. 729–746.

[36] D. Fetterly, M. Manasse, and M. Najork, "On the evolution of clusters of near-duplicate web pages," in *Proceedings of the First Conference on Latin American Web Congress*, ser. LA-WEB '03. Washington, DC, USA: IEEE Computer Society, 2003, pp. 37–. [Online]. Available: http://dl.acm.org/citation.cfm?id=951953.952397

[37] Q. Cui, G.-V. Jourdan, G. V. Bochmann, R. Couturier, and I.-V. Onut, "Tracking phishing attacks over time," in *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2017, pp. 667–676.