Diverse Datasets and a Customizable Benchmarking Framework for Phishing

Victor Zeng, Shahryar Baki, Ayman El Aassal, Rakesh Verma, Luis Felipe Teixeira De Moraes and Avisha Das
University of Houston
Houston, Texas
{vzeng,sbaki2,aelaassal,rverma,ltdemoraes,adas5}@uh.edu

ABSTRACT

Phishing is a serious challenge that remains largely unsolved despite the efforts of many researchers. In this paper, we present datasets and tools to help phishing researchers. First, we describe our efforts on creating high quality, diverse and representative email and URL/website datasets for phishing and making them publicly available. Second, we describe PhishBench, a benchmarking framework, which automates the extraction of more than 200 features, implements more than 30 classifiers, and 12 evaluation metrics, for detection of phishing emails, websites and URLs. Using PhishBench, the research community can easily run their models and benchmark their work against the work of others, who have used common dataset sources for emails (Nazario, SpamAssassin, WikiLeaks, etc.) and URLs (PhishTank, APWG, Alexa, etc.).

KEYWORDS

phishing, automatic framework, deception, social engineering, machine learning

ACM Reference Format:
Victor Zeng, Shahryar Baki, Ayman El Aassal, Rakesh Verma, Luis Felipe Teixeira De Moraes and Avisha Das. 2020. Diverse Datasets and a Customizable Benchmarking Framework for Phishing. In *Proceedings of the Sixth International Workshop on Security and Privacy Analytics (IWSPA '20), March 18, 2020, New Orleans, LA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3375708.3380313

1 INTRODUCTION

Phishing is a challenging problem that has been addressed by many researchers in several papers using many different datatsets and techniques [8]. Researchers usually test their proposed methods with limited metrics, datasets, and parameters when presenting new features or approach(es). Hence, the need arises for a benchmarking framework and dataset to evaluate such systems as comprehensively as possible. In this paper, we discuss: (i) our efforts on the creation and dissemination of diverse and representative datasets for phishing email, website and URL detection, and (ii) PhishBench, our framework for benchmarking phishing detection systems. PhishBench allows researchers to evaluate and compare features and classification approaches easily and efficiently on the provided data.

There are many dimensions of dataset quality [6], and no general agreement on what constitutes a high-quality dataset. Nevertheless, the desiderata include: accessibility, completeness, consistency, integrity, validity, interpretability, and timeliness. Creating high-quality datasets for security challenges such as phishing and malware is a tricky issue, since attackers are always evolving new attacks [33]. Hence, no security dataset for challenges such as phishing can be considered complete. Despite this limitation, we have been working on creating diverse, high-quality, public datasets for phishing research [1, 34]. We started with the attack vector of phishing, but we are now also creating datasets for other vectors like malware, etc.

While a variety of machine learning algorithms and evaluation metrics have been used in previous literature, there is a dearth of studies comparing the performance of such methods. Similarly, there also seems to be missing a proper study comparing metrics' suitability based on the nature of the data along with a good benchmarking framework for phishing detection research. We have designed and implemented PhishBench to fill this gap. It offers a variety of algorithms built for detection or classification problems including deep learning algorithms. It also offers metrics and methods suitable for imbalanced datasets, researchers can test and compare their worksin realistic scenarios. Currently, PhishBench provides feature extraction code for over 200 features gleaned from the phishing detection literature published between 2010 and 2018. It also offers a rich variety of classification algorithms including supervised, unsupervised, weighted and imbalanced methods. To summarize, our contributions are:

- We introduce datasets for phishing email, website and URL detection, which have been tested for diversity and quality (Section 2).
- We propose a novel benchmarking framework for machine learning tasks, specifically classification and detection, which provides 12 evaluation metrics and over 30 learning methods (including deep learning, online learning, and classical machine learning) and 15 imbalanced methods (Section 3).[1]
- We collect over 200 features tuned for the phishing detection task from the literature to evaluate the framework.
- We implement and test the complete pipeline on datasets of phishing websites and emails from public sources.

IWSPA '20, March 18, 2020, New Orleans, LA, USA

ACM ISBN 978-1-4503-7115-5/20/03…$15.00
https://doi.org/10.1145/3375708.3380313

[1] https://imbalanced-learn.readthedocs.io/en/stable

Table 1: Statistics of The URL Benchmark Dataset

Source	URLs	Extracted	Domains	TLDs	Logins
Alexa	31,163	29,173	9,554	285	2,056
Alexa Login	4,370	3,992	1,960	117	3,992
PhishTank	26,346	20,803	10,813	406	4,999
APWG	66,929	45,382	7,760	319	2,812
OpenPhish	2,249	1,336	710	94	326

In this paper, due to space limitation, we give just a brief overview of PhishBench; more details will be available in the full versionr [10].

2 PHISHING DATASETS

We now describe: (i) the phishing datasets in the existing literature that are already publicly available to the research community, and (ii) the new datasets that we are creating for public distribution. Information on how to obtain a copy of these datasets can be found at http://www2.cs.uh.edu/~rmverma/datasets.html.

2.1 URL Datasets

For phishing URLs/websites, there are three datasets: the URL Benchmark Dataset, PhishTank, and UCI Phishing. The URL Benchmark Dataset will be released by us; PhishTank and UCI Phishing are from existing literature and are publicly available.

The URL Benchmark Dataset: For building our dataset of legitimate websites,[2] we start with the top 40 website domains in each category from Alexa (September 5, 2018), as a seed for our crawler to generate a more realistic dataset. We limit the crawler to crawl up to three levels (it follows links within another website only to the depth of three). To keep the dataset diverse, we only store at most 10 URLs for each domain. We also build a separate dataset (Alexa Login) by downloading only pages with a login form from Alexa. Starting with a set of pages from Alexa as seed, the crawler crawls the web to extract pages that are only loginforms. We use a Python library called 'loginform 1.2.0'[3] to detect such pages.

For the phishing websites, we use three different sources: PhishTank (Sep 5, 2018), Anti-Phishing Working Group or APWG (Oct 30, 2018) and OpenPhish (Sep 5, 2018). We check the availability of URL, network and website information for each instance, and if any of the aforementioned information is not available (if the website is offline, it cannot retrieve the WHOIS information, etc.), the instance is excluded from the dataset.

Table 1 shows the basic statistics about the different datasets that we collected. The third column (Extracted) shows the number of URLs whose features were successfully extracted. The next two columns are the number of unique domains and Top Level Domains (TLDs), and the last column shows the total number of login pages.

PhishTank: Since phishing websites are short-lived, some researchers are creating an archive of the websites from PhishTank data [9]. In the year since we downloaded the URLs in PhishTank, the number of URLs in the archive has increased dramatically. As of September 23, 2019, PhishTank contained approximately 88,754 sites. This could be a rich source of data for conducting many different experiments in addition to building detectors. However, we caution that PhishTank is just one source, albeit many people contribute to it, so this dataset needs to be augmented with other sources for a genuinely diverse dataset.

UCI Phishing: Several researchers have used datasets on phishing websites available in the Attribute-Relation File Format (ARFF), usually used with WEKA machine learning tool. UCI Phishing is a family of datasets: Phishing Websites-old, Phishing Websites, and Website Phishing. Of these datasets, the Phishing Websites dataset (Date donated: 26^{th} March, 2015) comes with a total of 30 extracted attributes and can be accessed through the popular, publicly available University of California - Irvine's (UCI) machine learning repository. The source[4] consists of two slightly imbalanced datasets (the legitimate to phishing ratios are included in the parentheses) - an older version with features from 2456 instances (1094:1362) and a relatively newer dataset with 11,055 instances (6157:4896). The Website Phishing dataset[5] consists of 1353 instances belonging to three classes phishing (702), legitimate (548) and suspicious (103).

2.2 Email Datasets

We discuss three datasets, IWSPA-AP v2.0, The Email Benchmark Dataset, and Bluefin, for phishing email detection. All three datasets have been created by us. IWSPA-AP v2.0 is already publicly available with an agreement, and The Email Benchmark Dataset and Bluefin will be made available in the near future.

IWSPA-AP v2.0: The IWSPA-AP v2.0 dataset contains an improved version of the full-header subsets of the IWSPA-AP v1.0 dataset that was released for the 2018 IWSPA Anti-Phishing Pilot [29]. In this version of the dataset, the headers of the emails have been thoroughly cleaned, eliminating any mention of the source organization missed in the normalization step of the original construction. The detail of this cleaning is further described in [34]. This dataset includes both a full-header subset and a no-header subset.

The Email Benchmark Dataset: The Email Benchmark dataset is a dataset containing 10,500 legitimate and 10,500 phishing emails with unmodified headers from mulitple sources. For legitimate samples, we downloaded 6,779 emails from archives published by Wikileaks (www.wikileaks.org) – Hacking Team: 718, DNC: 3,098, GI files: 1,066, Sony: 1,120, National Socialist Movements: 678, Citizens Commission On Human Rights: 88, Plum emails: 11. We also downloaded 2,046 emails from the publicly available Enron dataset, and 1,675 emails from SpamAssassin.[6] For phishing samples, we downloaded 8,433 emails from the Nazario [7] phishing email dataset, but unlike previous research, we included 1,048 emails from its recently published 2015 to 2017 emails. We also added 1,019 spam emails from SpamAssassin.

Bluefin: One of the authors has collected approximately 300 phishing emails over the 2013-2018 period that *were not caught by the institutional filters*. We also plan to release this set, which, although smaller, is likely to be more challenging than some of the other unfiltered email datasets.

[2] Access dates for the sources are given in parentheses
[3] https://pypi.python.org/pypi/loginform
[4] https://archive.ics.uci.edu/ml/datasets/phishing+websites
[5] https://archive.ics.uci.edu/ml/datasets/Website+Phishing
[6] http://www.csmining.org/index.php/spam-assassin-datasets.html
[7] https://monkey.org/~jose/phishing/

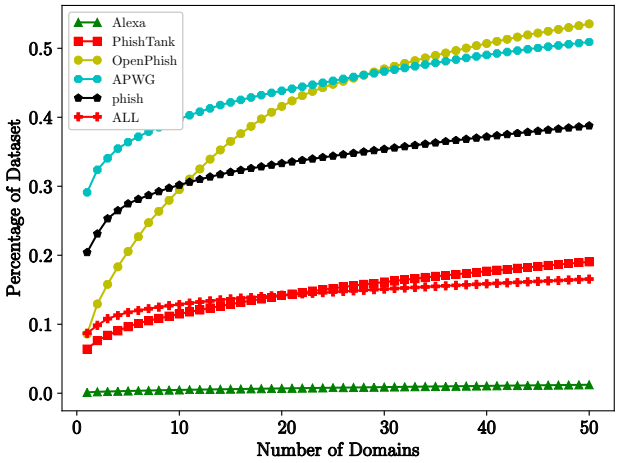


Figure 1: CDF of 50 most common domains in the URL Benchmark dataset by source. *Phish*: CDF of 50 most common domains in phishing subset *ALL*: CDF of 50 most common domains in the entire dataset. Alexa is the only *legit* source.

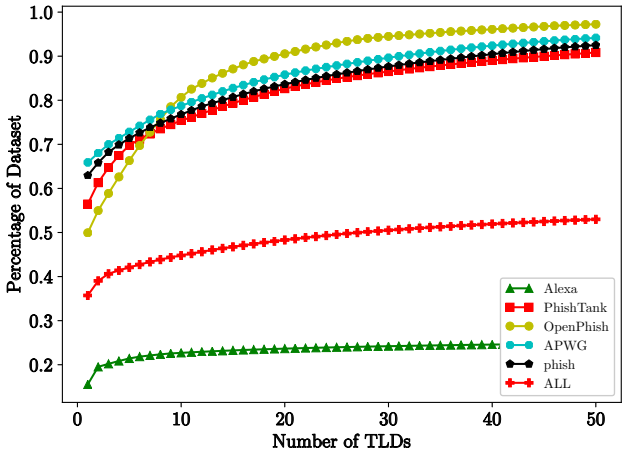


Figure 2: CDF of top 50 TLDs in in the URL Benchmark dataset by source. *Phish*: CDF of 50 most common TLDs in phishing subset *ALL*: CDF of 50 most common TLDs in the entire dataset. Alexa is the only *legit* source.

2.3 Dataset Diversity

Using a diverse dataset is crucial for having a generalizable model. We now discuss our efforts on ensuring dataset diversity.

URLs: We extracted the domains and TLDs from the URLs in the URL Benchmark dataset and analyze their distribution. If many URLs in the dataset are from the same domain, that means we have a bias toward some specific websites. Also, in the real world, we do not see a uniform distribution among different TLDs. Some of the TLDs are used more often, e.g. ".com" and ".org" and some are rarely used,[8] e.g. ".gw" and ".ax," so, we should not expect our dataset to have a uniform TLDs distribution.

In Figure 1 and 2, we rank the domains/TLDs by frequency and plot their respective CDFs. We see that the legitimate URLs are highly diverse. Since we limited the number of URLs per domain to 10, the percentage of URLs from the top 50 domains is almost zero. Among the phishing ones, Openphish and APWG are almost similar but the Phishtank is much more diverse. Figure 2 shows a huge gap for TLDs between the phishing and legitimate dataset which can be a sign of uniformity (lack of diversity) of phishing datasets.

Emails: We compare the content of pairs of emails to see how much similarity exists between them. The text content of each email (including the header) is extracted and all the HTML tags and CSS elements are filtered out. After removing the stop words, we extract Term Frequency Inverse Document Frequency vectors (TFIDF) [27] from the emails. Finally, we use the cosine similarity, which is the cosine of the angle between the two vectors, to measure the similarity between the TFIDF vectors of all email pairs. We present in Table 2 the ranges of similarities in both datasets (with/without header). The table shows that more than 85% of emails pairs have less than 10% similarity.

[8] http://www.seobythesea.com/2006/01/googles-most-popular-and-least-popular-top-level-domains/

Table 2: Distribution of cosine similarities for email pairs in the Email Benchmark dataset. FH: Full Header, NH: No Header

Dataset	Ranges of Similarities					
	[0-10]	(10-20]	(20-30]	(30-40]	(40-50]	>50
FH	85.44%	10.47%	2.60%	0.85%	0.29%	0.33%
NH	84.29%	10.74%	3.92%	0.55%	0.18%	0.29%

2.4 Checking Difficulty of Datasets

URL datasets: We conduct a small sample experiment to test the difficulty of the URL Benchmarking dataset used in our evaluation studies. The dataset consists of 83 attributes which include HTML and script-based features, network level features and URL based features. We use 80% of the dataset for training and the remaining is the 20% held-out test set. A random sample of 50 instances (consisting of both phishing and legitimate URLs) are selected from the training set to train the different classifiers provided by PhishBench.

The performance of the trained models was evaluated on the held-out test set. The experiment was repeated for a total of 30 iterations and we report the average accuracy and F1-score of the classifiers. The highest average F1-score and accuracy observed were 88.08% and 88.68% respectively, with the RandomForest classifier. Here, F1-score has been reported since the dataset is imbalanced.

Email datasets: To test the difficulty of the email datasets, we devised a small sample experiment in which we partitioned the dataset into a training set and test set. We then selected a small random sample from the training set consisting of 50 legit emails and 10 phish emails. We use this sample to train several classifiers in PhishBench and measure the performance of the trained models against the held-out test set. For the for datasets without headers, we used 48 features from the body, including the TFIDF vector of the body, and for the full-header datasets, we used the same 48 features, plus another 69 features from the headers.

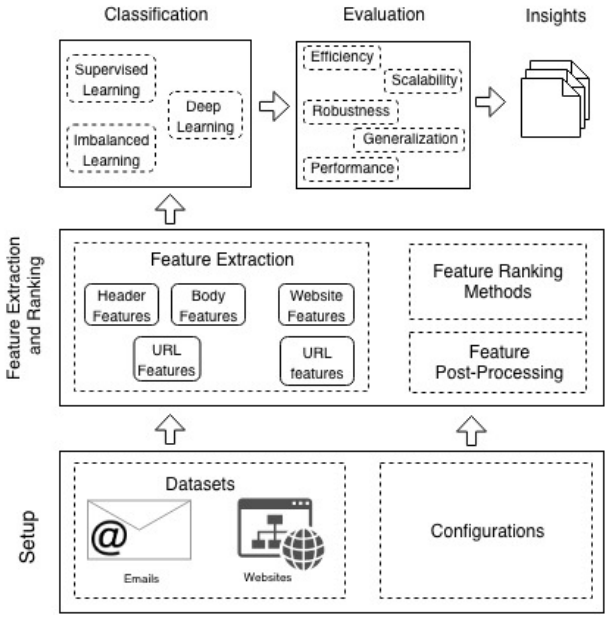


Figure 3: Proposed benchmarking framework and evaluation for phishing

Of special note is the method by which we selected the small sample. In the small sample experiment, we stratified our random sample by class. Our reasons for this are twofold. Firstly, it ensures that the classifiers are always exposed to instances from every class. Secondly, it eliminates variations in class distribution as a source of variation in classifier performance. To further reduce the variation in the results, we repeated this experiment 20 times with new samples from the training set.

On the IWSPA-AP v2.0 dataset, we found that the no-header subset produced an average accuracy of 88.9%, F1-score of 26.8%, and MCC (Matthews correlation coefficient) of 0.264 The full-header subset produced an average accuracy of 96.4%, F1-score of 83.6%, and MCC of 0.833 across all classifier implemented in PhishBenchs. These results demonstrate that while the full-header subset might still have some difficulty issues, these flaws are not present in the no-header subset. Note that since both subsets are imbalanced, accuracy is not a good measure. The F1-score and MCC gives a more realistic picture.

On the Email Benchmarking dataset, we obtained an accuracy of 79.2%, F1-score of 68.6%, and MCC of 63.0. These results indicate that the Email Benchmarking Dataset is not as affected by data difficulty issues as the full-header subset of the IWSPA-AP datasets.

3 PHISHBENCH ARCHITECTURE AND MODULES

PhishBench is a general framework for benchmarking machine-learning based phishing detection systems. It has five modules that represent the different stages of a general machine learning-based detection system. The modules have been set up to function as independent units based on the user's needs and the input given. The framework (shown in Figure 3) also tracks meta-information about its modules including feature extraction time, classification time, etc.

The **Input module** handles data loading and preprocessing functions. To focus on the issue of phishing detection, the input can be a list of URLs for phishing websites/URLs detection or a folder of email files for email detection. Based on the users' specifications for URL detection, this module parses the website HTML content and downloads all the network meta-data available. For email datasets, the module extracts header information and body content from each email. The parsed and decoded content from websites and emails is then used to extract features in the Feature Extraction module.

The **Feature Extraction** module contains the necessary functions to extract required information from the data passed by the *Input* module. This module uses Python's reflection functionality to enable rapid and easy implementation of features and allows the user to select which features to extract at run-time. The framework will be published with an attached ReadMe file that documents the process for implementing new features. The flexibility provided by this module helps researchers compare the results of their new features with already existing ones in the literature.

The **Feature Processing & Ranking** module handles feature ranking (Information Gain (IG), Gini Index (Gini), Chi-Square Metric (Chi-2), and Recursive Feature Elimination (RFE)) and normalization (Max-Absolute Scaler, Min-Max Scaler, Mean scaling, and L2 normalization) to be used on the raw input before classification. This module outputs a file containing a sorted list of features based on the results of the ranking algorithm used, and a sparse matrix of the best features returned by the algorithm. The ranking algorithm, scaling method and the number of best features are user-specified.

The **Classifiers** module implements popular machine learning-based classifiers used for classification tasks including both supervised and unsupervised, weighted, online learning, and imbalanced methods [20] for use as a baseline. The user has the option to choose which classifiers to run, weighted or not, and with or without imbalanced methods. For the supervised classification module, we implement the following learning algorithms: Support Vector Machines (with linear kernel), Random Forest, Decision Tree, Gaussian & Multinomial Naive Bayes, Logistic Regression, K Nearest Neighbors, Boosting (base classifier: decision tree), Bagging (base classifier: decision tree), and Deep Neural Networks. We also implement methods to handle imbalanced datasets including Repeated Edited Nearest Neighbor, ADASYN, and SMOTE [20].[9] Like the feature extraction module, this module also utilizes Python's reflection capabilities for rapid implementation of user-defined classifiers.

The **Evaluation** module evaluates the performance of phishing detection method. It reports the training/running times for each classifier, along with multiple common evaluation metrics, including accuracy, precision and recall for both legitimate and phishing classes, F1 score, weighted F1 score, geometric mean, matthews correlation coefficient, balanced accuracy score, and the ROC AUC.

Thus, PhishBench offers a uniform and highly customizable setup that will help researchers effectively compare their methods

[9] https://imbalanced-learn.org/en/stable/index.html

with works in the literature based on several evaluation metrics and tracking features.

4 AUTOML FRAMEWORKS

Automated Machine Learning (AutoML) frameworks provide methods and tools for non-expert users [5]. Given a dataset of extracted features, systems like AutoSKLearn [12] and TPOT [23] automate the entire pipeline of selecting and evaluating a wide variety of machine learning algorithms and finally outputting the decision. While auto-sklearn uses meta-learning along with Bayesian optimization to search the best algorithms from Python's ScikitLearn library; TPOT uses genetic programming to select the best ScikitLearn pipeline [13]. While these systems can automate the process pipeline selection, the frameworks are limited by the algorithms provided by the ScikitLearn library. Moreover, these systems evaluate their models on a dataset of preprocessed and extracted features, contrary to PhishBench. Such frameworks also fit several machine learning models on datasets while using multiple preprocessing steps (scaling, feature selection, etc) and hyperparameter tuning for all the models, thus increasing the time to converge.

5 USING PHISHBENCH

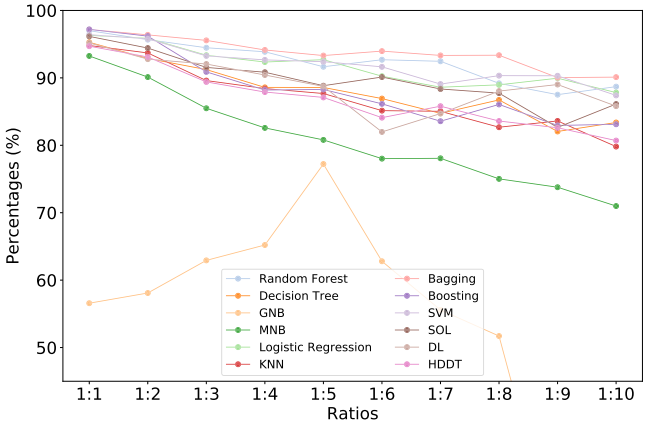
We now show the results obtained using our framework on the phishing emails and URL datasets. We performed hyperparameter tuning using Randomized Parameter Optimization to optimize the classifiers.

5.1 URLs

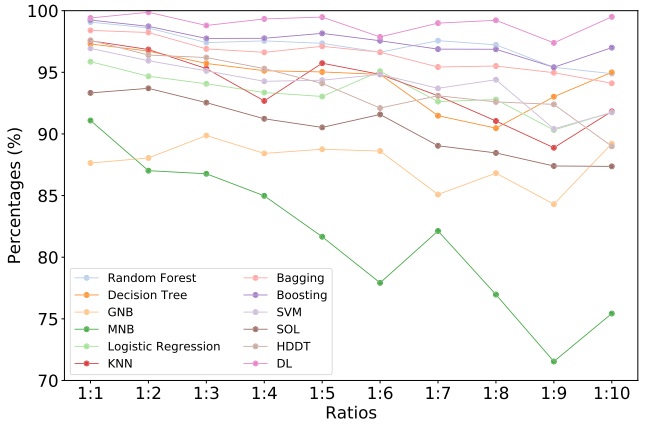
In the real world, the chance of encountering a phishing URL is much lower than its legitimate counterpart, which in turn affects classifier performance. To have a more realistic evaluation of the models' performance, we evaluate our models with various imbalance ratios. Keeping the total number of legitimate and phishing URLs fixed to 36,457 (the maximum allowable size to obtain the 1:10 ratio), we change the ratio of phishing to legitimate samples from 1:1 to 1:10.

Figure 4a displays classifiers' performance using all the features implemented in PhishBench on the different URL dataset ratios. We report the F1-score instead of accuracy since it is a metric more appropriate for evaluating classification of imbalanced datasets. A common observation among all the classifiers is the performance downtrend concerning class imbalance ratio. Gaussian Naive Bayes (GNB) has the most decline in F1-score while bagging and Logistic Regression have the least decline, 7.5% and 8.48% respectively. Boosting which had a similar performance to Bagging in 1:1 ratio has a reduction twice that of Bagging (14.4%) which makes it a bad choice for the real-world scenario. MCC values for URL dataset (Figure 5) also showed the same pattern.

To further investigate the diversity of different data sources, we used the best classifier (Bagging) and performed a cross-dataset experiment in which we train on one dataset and test on a different dataset (the legitimate subset stays the same). First, we used URLs from APWG as the phishing training data source and Alexa as the legitimate source. After training the classifier, we used URLs from PhishTank as a test set and the model achieved 57% accuracy. In



(a)



(b)

Figure 4: F1-score with varying ratios between phishing and legitimate instances (*a.* for URL Benchmark Dataset and *b.* for email Dataset B). k-NN with $k = 5$ for URLs and $k = 3$ for emails. Bagging and Boosting use Decision Tree as their base classifier SOL: Scalable Online Learning [35], DL: Deep Learning [19], HDDT: Hellinger Distance Decision Tree [21]

the next experiment, we switched APWG and PhishTank (PhishTank used for training and APWG for testing). With PhishTank as training set, the model achieved 97.5% accuracy, which is significantly higher than 57%. These results are in line with our earlier discussion about diversity of PhishTank and APWG (Section 2.3); we did compute the overlap between the datasets and there were only 80 identical URLs in PhishTank and APWG.

5.2 Emails

We also run the email classification experiment using all the features on different phishing to legitimate ratios to simulate real-world scenarios. We fix the size of the dataset to 11550 emails (the maximum allowable size to obtain the 1:10 ratio) and we test on a range of different ratios from 1:1 to 1:10. Same as the URL experiment, we see a decreasing trend in F1-score as we decrease the ratio of

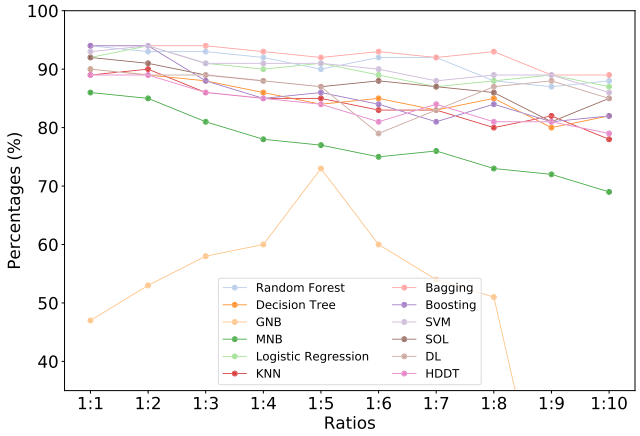


Figure 5: MCC with varying ratios between phishing and legitimate instances for URL Benchmark Dataset

phishing to legitimate emails (Figure 4b). The classifier with the biggest decline between 1:1 and 1:10 ratios is Multinomial Naive Bayes (MNB) with a total loss of 15.66% in F1-score, which happened gradually as the ratio increases. The model with the highest F1-score at 1:10 ratio is Deep Learning (99%). We do not show the plot for MCC values since it revealed the same pattern as F1-score (same as the URL experiment).

For the cross-dataset experiment, first we compared Nazario and Bluefin using Boosting as the best classifier. Since none of these sources has legitimate emails, we included legitimate emails from *Email Benchmark Dataset* for the training. Both models performed poorly when tested another source. The model trained on Nazario got the accuracy of 21% on Bluefin and the model trained on Bluefin got the accuracy of 25%. We also compared SpamAssassin and Nazario, even though the former has spam emails and not phishing. The model trained on SpamAssassin and tested on Nazario achieved 78% accuracy which is much higher than the model trained on Bluefin. It shows that the Bluefin dataset (which is also more recent), contains different, and probably more advanced, type of attack emails.

6 RELATED WORKS

In this work, we introduced a phishing detection framework as well as diverse datasets for phishing websites and emails. So, we divide the related works into two parts 1) Benchmarking Frameworks and 2) Phishing Dataset.

6.1 Benchmarking Frameworks

To our knowledge, this is the first benchmarking framework specific for phishing detection task. There are several libraries available for researchers to implement machine learning systems from scratch, e.g. Scikit-learn (Python), Weka (Java) and Caret (R). Besides libraries, tools like RapidMiner and SPSS makes it easier for users to implement a model by removing the programming burden. The main issue with all the above methods is the need for implementing the code for feature extraction. In PhishBench, we implemented more than 200 existing features from the literature to make it easier for researchers to compare their new methods with existing systems with identical parameters.

Despite all the research in phishing website and email detection, there is no existing framework that combine all the features and algorithm in one place. Most researchers do not make their system' implementation available, which makes the comparison with previous work hard. Besides, there is a lack of common dataset that can be used by researchers to compare their works, which we discuss it in the next section.

6.2 Phishing Datasets

6.2.1 Website. PhishTank [24] and OpenPhish [25] are the main sources for collecting phishing websites. For legitimate URLs, researchers have been using Alexa [4], DMOZ (deprecated), and Yahoo Directory (deprecated) as the sources. Although they do not provide a dataset for phishing detection, researchers combine the URLs from these sources to create their own dataset. Knowing the fact that these dataset are dynamic and change over time, it makes the comparison between existing works almost impossible. There has been some efforts in creating fixed datasets to make the comparison possible [2, 26, 28] but none of the researchers analyzed diversity of their dataset. They only provided a set of predefined feature sets without providing the actual URL that these features are collected from. So, we could not do the diversity analysis of these datasets ourselves. Sizes of these datasets are also generally small, authors in [2] provided 1,353 URLs with only 10 features, and the ataset created in [26] has 2,456 URLs with 30 features. The third one is a little larger with 10,000 URLs (5,000 each) and 48 features.

6.2.2 Email. The Nazario Phishing Corpus [22] and SpamAssassin[10] are the most commonly used sources for phishing and legitimate emails in research respectively. Researchers have used a combination of these two datasets in varying ratios in [11, 14, 18, 30]. However, these publicly available datasets maybe sanitized like the emails from the Enron[11] corpus (headers are sanitized) and SpamAssasin ("lightly sanitized" headers according to the source) which have obfuscated addresses and domains. Therefore, researchers also choose to use their own email datasets [31] or collect email data from company archives and logs [17, 36]. Spear phishing datasets are also quite difficult to come by in literature. In [16], researchers used an imbalanced dataset of 1467 spear phishing emails collected from Symantec's enterprise email scanning services along with 14,043 benign emails collected between 2011 and 2013. Some of the papers that evaluated classifier performance on different ratios of legitimate to phishing datasets were [3, 7, 14, 15, 32].

7 CONCLUSIONS

In this paper, we have presented several robust datasets and a flexible benchmarking framework, PhishBench, for the pernicious phishing problem. Some of these datasets are already publicly available and the rest will be made publicly available for researchers working on phishing detection to compare their work and also to quickly prototype their ideas and features. PhishBench is also slated

[10] http://www.csmining.org/index.php/spam-assassin-datasets.html
[11] https://www.cs.cmu.edu/~enron/

for public release. We hope that these products will spur phishing research of high quality.

ACKNOWLEDGMENTS

We thank the IWSPA reviewers for their suggestions. Thanks to NSF for partial support under grants CNS 1319212, DGE 1433817, and DUE 1356705. This material is also based upon work supported in part by the U. S. Army Research Laboratory and the U.S. Army Research Office grant number W911NF-16-1-0422.

REFERENCES

[1] Ayman El Aassal, Luis Moraes, Shahryar Baki, Avisha Das, and Rakesh Verma. 2018. Anti-Phishing Pilot at ACM IWSPA 2018: Evaluating Performance with New Metrics for Unbalanced Datasets. In *Proc. of IWSPA-AP: Anti-Phishing Shared Task Pilot at the 4th ACM IWSPA*. 2–10. http://ceur-ws.org/Vol-2124/#anti-phishing-pilot

[2] Neda Abdelhamid, Aladdin Ayesh, and Fadi Thabtah. 2014. Phishing detection based associative classification data mining. *Expert Systems with Applications* 41, 13 (2014), 5948–5959.

[3] Andronicus A. Akinyelu and Aderemi O. Adewumi. 2014. Classification of Phishing Email Using Random Forest Machine Learning Technique. *Journal of Applied Mathematics* 2014 (2014), 1–6. https://doi.org/10.1155/2014/425731

[4] Alexa. 2019. Alexa Top Sites. https://aws.amazon.com/alexa-top-sites/

[5] Adithya Balaji and Alexander Allen. 2018. Benchmarking Automatic Machine Learning Frameworks. (2018).

[6] Carlo Batini, Cinzia Cappiello, Chiara Francalanci, and Andrea Maurino. 2009. Methodologies for Data Quality Assessment and Improvement. *ACM Comput. Surv.* 41, 3 (July 2009), 16:1–16:52.

[7] André Bergholz, Jan De Beer, Sebastian Glahn, Marie-Francine Moens, Gerhard Paass, and Siehyun Strobel. 2010. New filtering approaches for phishing email. *Journal of Computer Security* 18 (2010), 7–35.

[8] A. Das, S. Baki, A. El Aassal, R. Verma, and A. Dunbar. 2019. SoK: A Comprehensive Reexamination of Phishing Research from the Security Perspective. *IEEE Communications Surveys Tutorials* (2019), 1–1. https://doi.org/10.1109/COMST.2019.2957750

[9] D. G. Dobolyi and A. Abbasi. 2016. PhishMonger: A free and open source public archive of real-world phishing websites. In *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*. IEEE, Tucson, AZ, USA, 31–36. https://doi.org/10.1109/ISI.2016.7745439

[10] Ayman El Aassal, Shahryar Baki, Avisha Das, and Rakesh Verma. 2020. An In-Depth Benchmarking and Evaluation of Phishing Detection Research for Security Needs. *IEEE Access* (2020).

[11] Ian Fette, Norman Sadeh, and Anthony Tomasic. 2007. Learning to Detect Phishing Emails. In *Proceedings of the 16th International Conference on World Wide Web (WWW '07)*. Association for Computing Machinery, New York, NY, USA, 649–656. https://doi.org/10.1145/1242572.1242660

[12] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, and Frank Hutter. 2015. Efficient and Robust Automated Machine Learning. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'15)*. MIT Press, Cambridge, MA, USA, 2755–2763. http://dl.acm.org/citation.cfm?id=2969442.2969547

[13] P. J. A. Gijsbers, Erin LeDell, Janek Thomas, Sébastien Poirier, Bernd Bischl, and Joaquin Vanschoren. 2019. An Open Source AutoML Benchmark. (2019).

[14] I. R. A. Hamid and J. Abawajy. 2011. Phishing Email Feature Selection Approach. In *2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*. IEEE, Changsha, China, 916–921. https://doi.org/10.1109/TrustCom.2011.126

[15] I. R. A. Hamid and J. H. Abawajy. 2013. Profiling Phishing Email Based on Clustering Approach. In *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*. IEEE, Melbourne, VIC, Australia, 628–635. https://doi.org/10.1109/TrustCom.2013.76

[16] YuFei Han and Yun Shen. 2016. Accurate Spear Phishing Campaign Attribution and Early Detection. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing (SAC '16)*. Association for Computing Machinery, New York, NY, USA, 2079–2086. https://doi.org/10.1145/2851613.2851801

[17] Cheng Huang, Shuang Hao, Luca Invernizzi, Jiayong Liu, Yong Fang, Christopher Kruegel, and Giovanni Vigna. 2017. Gossip: Automatically Identifying Malicious Domains from Mailing List Discussions. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security (ASIA CCS '17)*. Association for Computing Machinery, New York, NY, USA, 494–505. https://doi.org/10.1145/3052973.3053017

[18] M. Khonji, Y. Iraqi, and A. Jones. 2011. Lexical URL analysis for discriminating phishing and legitimate e-mail messages. In *2011 International Conference for Internet Technology and Secured Transactions*. IEEE, Abu Dhabi, United Arab Emirates, 422–427.

[19] Hung Le, Quang Pham, Doyen Sahoo, and Steven CH Hoi. 2018. URLNet: Learning a URL Representation with Deep Learning for Malicious URL Detection. (2018).

[20] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. 2017. Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research* 18, 17 (2017), 1–5. http://jmlr.org/papers/v18/16-365

[21] Wei Liu, Sanjay Chawla, David A Cieslak, and Nitesh V Chawla. 2010. A robust decision tree algorithm for imbalanced data sets. , 766–777 pages.

[22] Jose Nazario. 2004. The online phishing corpus. https://monkey.org/~jose/phishing/

[23] Randal S. Olson, Nathan Bartley, Ryan J. Urbanowicz, and Jason H. Moore. 2016. Evaluation of a Tree-based Pipeline Optimization Tool for Automating Data Science. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016 (GECCO '16)*. ACM, New York, NY, USA, 485–492. https://doi.org/10.1145/2908812.2908918

[24] OpenDNS-PhishTank. 2012. The PhishTank Database. http://www.phishtank.com/developer_info.php.

[25] OpenPhish. 2019. OpenPhish. https://openphish.com/index.html

[26] Mustafa A Mohammad Rami, McCluskey Lee, and Thabtah Fadi. 2015. UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/datasets/phishing+websites

[27] Gerard Salton and Michael J. McGill. 1986. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.

[28] Choon Lin Tan. 2018. Phishing Dataset for Machine Learning: Feature Evaluation. http://dx.doi.org/10.17632/h3cgnj8hft.1#file-286768bb-83f2-4e59-9210-6fed84e3c7fd

[29] Rakesh Verma and Avisha Das (Eds.). 2018. *Proceedings of the 1st Anti-phishing Shared Pilot at 4th ACM IWSPA (IWSPA-AP)*. CEUR. http://ceur-ws.org/Vol-2124/.

[30] R. Verma and N. Rai. 2015. Phish-IDetector: Message-ID based automatic phishing detection. In *2015 12th International Joint Conference on e-Business and Telecommunications (ICETE)*, Vol. 04. IEEE, Colmar, France, 427–434.

[31] Rakesh Verma, Narasimha Shashidhar, and Nabil Hossain. 2012. Detecting Phishing Emails the Natural Language Way. In *Computer Security – ESORICS 2012*, Sara Foresti, Moti Yung, and Fabio Martinelli (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 824–841.

[32] Rakesh M. Verma and Nabil Hossain. 2014. Semantic Feature Selection for Text with Application to Phishing Email Detection. In *Information Security and Cryptology – ICISC 2013*. Springer International Publishing, Seoul, Korea, 455–468.

[33] Rakesh M. Verma and David Marchette. 2019. *Cybersecurity Analytics*. Chapman and Hall/CRC, Boca Raton/London.

[34] Rakesh M. Verma, Victor Zeng, and Houtan Faridi. 2019. Data Quality for Security Challenges: Case Studies of Phishing, Malware and Intrusion Detection Datasets. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS '19)*. Association for Computing Machinery, New York, NY, USA, 2605–2607. https://doi.org/10.1145/3319535.3363267

[35] Yue Wu, Steven C.H. Hoi, Chenghao Liu, Jing Lu, Doyen Sahoo, and Nenghai Yu. 2017. SOL: A library for scalable online learning algorithms. *Neurocomputing* 260 (2017), 9–12. https://doi.org/10.1016/j.neucom.2017.03.077

[36] J. Yearwood, M. Mammadov, and A. Banerjee. 2010. Profiling Phishing Emails Based on Hyperlink Information. In *2010 International Conference on Advances in Social Networks Analysis and Mining*. IEEE, Odense, Denmark, 120–127. https://doi.org/10.1109/ASONAM.2010.56