Logit: Automating Detection of Login Fields

Hassaan Ahmad Waqar
Lahore University of Management Sciences
(LUMS) 22100137@lums.edu.pk

Umme Ammara
Lahore University of Management Sciences
(LUMS) 22100100@lums.edu.pk

ABSTRACT

As the internet continues to expand, novel methods need to be employed to maintain the security and privacy of its users. Emphasis must be paid upon securing instances where users are required to input sensitive information. The login page is one such sensitive area and great care must be taken to keep it secure. One measure involves notifying the user when they are about to input sensitive credentials. This requires detecting these login fields in the first place. Our work takes motivation from this context. We develop a login detector, Logit, based on parsing HTML elements and comparing them with identifier keywords that we collect by inspecting each web page in our train data individually. In addition to that, we collect and label datasets for both benign and phishing web pages that can be used to test the implementation of our login detector. We observed that our detector performed well on benign web pages that do not contain sign-up forms. However, web pages in languages other than English or those that used dynamic HTML elements were able to bypass the detection. Overall, Logit achieves an accuracy of 79.6% in detecting login fields. By increasing the size of labeled and test dataset in the future and catering to corner cases, we can improve the performance of our login detector.

1. INTRODUCTION

The internet has evolved into a vast network of interconnected users over the years with an estimated population of 4.88 billion in 2021 [15]. Such high popularity figures also mean that there is an undeniable need to pay emphasis upon adding additional layers to promote the safety and privacy of these users who go online. Instances where users are required to input credentials need to be protected to prevent any security mishap. The login page is one such instance which should protect the data input by the users. In addition to that, it should notify users that they are about to input sensitive credentials on any site. This would also enable users to be mindful when inputting data to any website. Works have earlier been conducted to scrutinize the security of these login pages [6]. They found widespread vulnerabilities within them. Hence, there is a dire need to enable newer methods to enhance the security of login pages.

Prior to making login pages more secure, we need to detect them. This can serve a number of purposes. Firstly, it would allow us to notify users when they visit a web page that requires login credentials. A similar functionality is used in Google Safe Browsing [14] where phishing websites are detected based on a blacklist record, and users are notified of the threat. Secondly, it can be used to facilitate automated testing procedures of large-scale web pages where the login functionality is required. Once login fields are detected, inputting credentials into it will not be an issue. Finally, a login detector can also be used to study phishing web pages and techniques used by an attacker to capture sensitive inputs from the user.

Our works lie across multiple domains mentioned above. We first collect and label a dataset for active benign and phishing websites in order to study the trends of HTML identifiers used to represent login fields in each of the two. Next, we write a script for a login detector that determines if a web page contains a login field or not. It does so by parsing HTML DOM elements and finding identifier keywords to detect login credentials. Finally, we analyze results and present instances where login detection through HTML parsing works, and others where it does not. Our work can be reproduced and extended by the research community in order to enhance the detector's ability, and carry out large-scale analysis of web pages.

The paper is structured in the following manner. Section 2 presents prior works done in our research domain. Section 3 discusses the methodology for each phase of our work. Section 4 presents our findings and results, along with their explanations. Section 5 sheds light into the limitations we faced. Section 6 provides a direction for future works. Section 7 highlights our contributions in the paper. Finally, Section 8 offers a conclusion.

2. RELATED WORK

Our approach draws its inspiration from the wider literature that is available on each of the domains that our work lies in.

Role of Browsers in Login Fields

The login page is one of the most sensitive platforms for most websites and great care must be taken to keep them secure. While the use of strong passwords is important and can not be underestimated, the security of login pages themselves needs to be paid attention to. Acker et. al conducted an empirical study of Alexa's top 100,000 pages in order to scrutinize their security [6]. They found

widespread vulnerabilities including risks of eavesdropping and sharing passwords with third-party organizations.

Similarly, a comparative analysis was conducted to access the best performing phishing detection tool. The comparison of 8 tools (browser extensions and plug-ins) showed that AntiPhishing Toolbar outperformed the rest with the highest accuracy [7]. *In our study, we will delve deeper and assess the effectiveness of similar detection tools by creating a login page detector*. Major browsers (Chrome, Safari, Explorer, Opera, etc.) typically use a black-list to pinpoint malicious pages, and through their in-built phishing and malware protection system, detect the attack [8]. A recent study was conducted on measuring the effectiveness of server-side anti-phishing bots in detecting phishing websites. The results concluded that Google Safe Browsing detected all the URLs protected by alert boxes [12]. *However, these browsers do not warn the user about potential threats when entering their credentials in the login fields*. *In our work, we intend to address this limitation by exploring the capability of browsers to detect the login fields and build upon the current research to establish a well-functioning login page detector*.

Phishing Detection

Works have earlier been conducted on studying the detection of phishing websites based on user behavior and login status of web pages when fed with fake credentials. Rap and Pais claimed to be the first to automate this procedure of inputting fake credentials to web pages and seeing the response using Selenium WebDriver [1]. Websites were then filtered and classified as phishing or legitimate. *While this approach is an application of our research domain and based on the various uses of login fields, we focus more on automatically detecting login fields on web pages.*

Vision-based techniques have also been explored for the detection of phishing websites. Afroz and Greedstadt proposed the *PhishZoo* tool that detects phishing based on profiles of sensitive sites' appearance and content [3]. Profiles are matched against already proven phishing content held in a local database. *This technique works well for web pages that have not been blacklisted earlier as well. Our paper involves a similar approach but for the detection of login fields on web pages.*

One popular approach is using machine learning models to classify emails as spam/not spam or websites as phishing/legitimate. A study conducted in 2007 compares different machine learning models, and based on 43 email features, it concluded that Regression Trees outperformed the other classifiers with the highest accuracy [9]. Marchal et. al adopted a machine learning approach to select various features of a webpage that can be used to detect phishing [2]. These included different parts of the URL and constraints faced by phishers. They report a detection success rate of around 90.5-97.3%.

Similarly, deep learning approaches have been used to extract logos (of brands) from web pages and compare them with authentic logos of brands that have been targeted. Lin et al. propose Phishpedia: a tool that uses a hybrid deep learning system that decomposes phishing detection into the identification of identity logs and then recognizing the brands [4].

A large-scale phishing prevention scheme proposed by Florencio and Herley uses password re-use (PRU) to detect an attack. Instead of relying on login fields, URLs, or HTML content to detect phishing or malicious activity, the researchers rely on password re-entering to indicate quick detection [10]. By using large-scale detection, the reliability of detection increases, and the mitigation is made feasible. Another large-scale measurement around phishing threats focuses on a URL-based approach. The researchers used the newly identified features of URL sites to train a logistic regression model that displayed high accuracy. The two primary datasets formed for this study used Google's index infrastructure to obtain features and top-level domains [11]. The analysis of several million URLs validated the threat of phishing. *We will follow a similar approach and conduct large-scale measurements using a diverse dataset.*

Phishing Datasets

Different attempts have been made on collecting data that can be used for detecting phishing. This includes emails, web pages, and image datasets. Common datasets such as Phishtank, Alexa, APWG, and Google Safe Browsing have been organized and used in various papers [5].

[4] collected datasets for identifying solutions to phishing through vision using screenshots, logos, and HTML contents. One relevant and recent research in this regard examines the creation of high-quality, diverse, and representative datasets for URLs and phishing emails. In addition to this, Zeng et al. founded a benchmarking framework called PhishBench, which automates feature extraction from these datasets [13]. By making these datasets and features publicly available, the research community can run their models on this data and further the measurements in this domain.

We did observe that none of the datasets that exist online were classified according to the presence of login fields on a webpage. Therefore, our works provide the first small-scale dataset that attempts to classify web pages on the basis of the presence of login fields, in addition to being benign or phishing.

3. METHODOLOGY

This section presents an overview of our methodology for preparing data, writing the code for detecting login fields and analyzing results.

Dataset preparation

The objectives for our login detector revolved around identifying the differences observed while detecting different types of web pages. We relied on parsing HTML DOM elements in order to obtain keywords that can be used to distinguish various categories of web pages from one another. In particular, we collected two broad categories of web pages: benign and phishing. We aimed to collect a diverse and large-scale dataset of active web-pages that can first be used to extract identifier information, and then modeled as testing data for our identifier.

In order to form a diverse dataset, we consulted and merged online datasets for both benign and phishing websites. We relied on community collected datasets on *Kaggle* and web pages on *Alexa* for both benign and phishing websites. This ensured that web pages that were used for our purpose did not belong to a single language, domain, or geographical background, and spanned a variety of design templates. Benign websites were further divided into four categories: general web pages, banking websites, web pages with signup forms and web pages with e-commerce newsletter forms. These were used to see if there was any difference observed in how accurately our detector identifies login fields in each of the above categories. On the other hand, identification of phishing websites has remained a challenge. To verify that a website is phishing, we relied on several metrics:

1. *Google Chrome* provides a security warning when a user tries to access a phishing website. We looked out for these warnings. On a side note, these warnings can be ignored by the user in order to proceed to the website for analysis.
2. The address bar turns red when a phishing website is being accessed.
3. The URL of the website preferably does not have HTTPS or the padlock symbol (though this may not always be valid since an increase in phishing websites having SSL certificates has been observed over the years).
4. The contents of a phishing website mimic the original one, but are recognizable due to lower quality, fuzzy images and incorrect linguistics.

For benign web pages, we collected separate datasets for banking, sign-up, and e-commerce newsletters web pages. These were part of the test set only for our model. The main purpose of these datasets was to reduce the chances of false positives produced by our login detector i.e., classifying sign-up pages as login ones because of the similarity of fields. Moreover, banking websites were included because they incorporate additional security to protect their users. This includes that users cannot inspect the source code of the web page as well. We wanted to see the performance of our detector on such pages.

Banking websites were chosen from a diverse geographical domain. We included websites of top banks from the USA, UK, China, Australia, Russia, India, and Pakistan. All banking pages collected in our dataset contained a login field. For web pages with sign-up forms, we focused on the design aspect of the web page and included a variety of design templates. They differentiated in the number of fields offered to the user, their wordings, and the layout. None of the sign-up pages contained login fields Finally, e-commerce websites with newsletter forms were chosen from a diverse geographical domain as well including websites from the countries that were mentioned above. None of these e-commerce web pages contained login fields. All these steps were taken to ensure that our dataset was diverse and representative of web pages that exist only. We try to minimize the bias in web page selection such that our login detector can give sound results.

Our next step involved validating that each of our collected web pages were active. Unfortunately, community collected datasets are not updated regularly. This results in a large proportion of the dataset becoming inactive and hence could not be used in our experiments. To ensure that a web page is active or not, we relied on the HTTP status code returned by the website when it was requested. For this, we used the *Requests* module in *Python*. Websites that returned a status code in the range of 400s were marked as inactive. Others that returned a status code in the range of 200s were marked as active. Inactive websites were filtered out and removed from the dataset. We also observed that most of the phishing websites in our dataset were inactive. This is because once a website is widely reported as phishing, it gets taken down from the internet. Moreover, search engines such as Google have employed active measures to detect and restrict user access to phishing websites. An example of this can be Google Safe Browsing [14] feature that compares each website to a blacklist of phishing websites, and restricts access if a website has been marked as phishing. Therefore, this step was executed in several iterations where our dataset was filtered out for inactive web pages.

Our final step in data preparation was to classify if a login field existed in a web page or not. To the best of our knowledge, we did not find a single dataset online that has classified web pages based on the presence of a login field

in them. For the purpose of our experiments, it was necessary to classify web pages based on login fields so that we can test the performance of our login detector later on. This meant every single web page in our dataset was visited manually and a record was maintained if it contained a login field or not. This acted as our gold labels for the dataset. We considered the input of username/email and password both to mark it as a login field. Sign-up forms and email newsletter fields were not marked as login fields. This step was the most time-consuming step of the entire process and spanned a number of days for completion.

The table below records the total number of web pages included in our dataset by category:

Table 1: Final counts of web pages in each dataset

		With Login	Without Login	Total
Benign	**Train data**	43	450	493
	Test data	168	661	829
	Total	211	1111	1322

		With Login	Without Login	Total
Phishing	**Train data**	38	405	443
	Test data	19	72	91
	Total	57	477	534

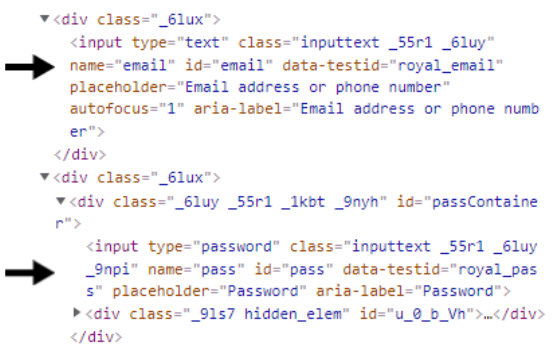
Used for test data only	
Category	**Count**
Banking web pages	110
Web pages with sign-up form	76
E-commerce web pages with newsletter form	59

Implementing the login detector

As mentioned earlier, HTML DOM parsing formed the basis of our login detector. Each webpage has an underlying HTML structure that forms a tree. Elements are embedded inside the tree under specialized tags. Our implementation uses these tags to identify each element and compare it against identifier lists maintained for benign and phishing web pages. We use Selenium WebDriver to automate the process of web page identification and manipulation.

We split our dataset into train and test data. The train data consisted of benign and phishing web pages each labeled as with or without a login field. Banking, sign-up and newsletter web pages were only part of the test data since they were used to see if false positives were being produced or not. For our train data, we visited each web page individually to see if it contained a login field. If it did, we inspected the source code and listed the *name* and *id* element identifiers of both username/email and password (separately for both benign and phishing websites). This way, we collected identifiers used in web pages to recognize the username/email and password fields. These lists of identifiers were later used to compare them with the fields in test data in order to see if similar keywords were present or not. Figure 1 displays the portions of the HTML code that we inspect for each web page in order to identify the keywords used for username/email and password:

Figure 1: HTML code showing name and id fields



If a web page in the test data contained fields with the same identifiers (as collected from the train data), we would label it as returning a *true* (login field present). Else, we would rely on our second strategy.

In case our first strategy did not find a login field, we would look for other keywords that are associated with login fields generally. Our manual examination of these web pages hinted that most login fields were accompanied by *Remember me* or *Forgot your password* keywords. Therefore, we compiled a list of these phrases that are linked to login fields using training data. The list included keywords such as ‘Remember me’, ‘Remember password’, ‘Keep me signed in’, ‘Forgot password’, and ‘Forgot username and password’ etc. Our observations during this stage show that these keywords are mainly associated with login pages only and not with other pages such as sign-ups. If a web page in the test data contained these keywords, we

would label it as returning a *true* (login field present). Else, we would rely on our third strategy.

Our final strategy involved searching for the *Login* button on the web page. We assume that the URL provided to the login detector could be of a home page of a website with a *Login* button available. The login form appears when this button is clicked. In order to increase coverage of our detector, and cater to login forms that appear as pop-ups upon clicking the *Login* button (without the URL changing), we locate the *Login* button and attempt to click it. If this operation is successful, we re-try the previous two strategies to search for a login field. If one is found, we would label the URL as returning a *true* (login field present). If strategy three is unable to find a Login button, we simply return a *false* (login field not found).

Finally, in order to distinguish login pages from sign-up ones (since fields may be similar in the two), we search for keywords associated with sign-up forms. An identifier list for sign-up forms was created from the training data as well. The list included keywords such as 'by signing up', 'agree to our terms', and 'Terms of Service and Privacy Policy' etc. This helped in making our login detector more robust, and reducing the number of false positives.

The flowchart below summarize all the strategies for implementing the login detector:

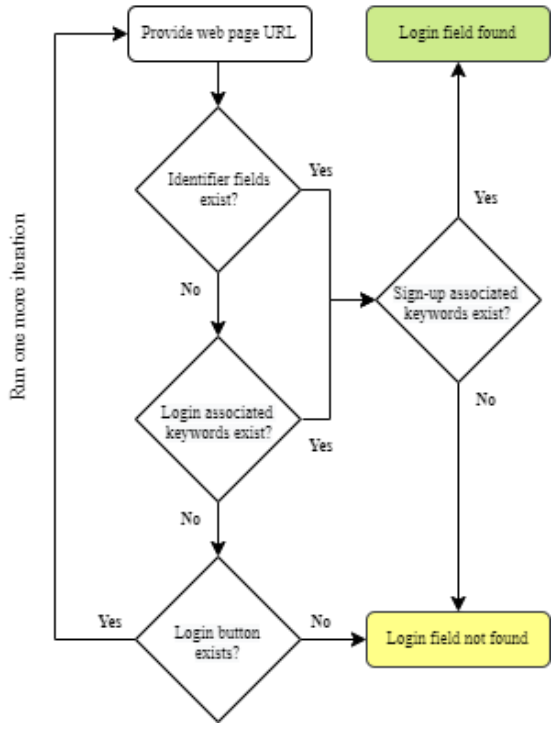


Figure 2: Algorithmic flowchart of the login detector

Analyzing results

We ran the login detector with each categorical (benign, phishing, banking…) list of URLs sequentially and collected the results. Each iteration returned a list of labels as predicted by the detector. We compared the predictions with the true labels and calculated relevant evaluation metrics. For each misprediction, we visited the URL manually again and tried to infer why the misprediction occurred. Our findings highlight instances where the detector worked and others where it failed to deliver the correct output.

4. RESULTS

We evaluated the performance of our login detector on both train and test data. We discuss the results for each phase separately below.

Training Phase

We first ran our detection algorithm on training data. It consists of web pages that contained login fields and those that did not. A key point to consider is that we collected the identifier keywords for *name* and *id* fields of username/email and password fields from the same training data. Separate lists were maintained for benign and phishing web pages. The results for benign web pages are shown in Table 2 below.

The following notation will be used as the key while representing headers of evaluation tables:

LFE: Login field exists
LFDE: Login field does not exist
LFD: Login field detected
LFND: Login field not detected

Table 2: Evaluation results for benign train set

Benign web pages				
	Gold labels			
System Labels		LFE	LFDE	Total
	LFD	41	4	45
	LFND	2	446	448
	Total	43	450	493

Accuracy: 98.7%
F-Score: 0.93
Precision: 0.91
Recall: 0.95

Similarly, we perform the same evaluation for phishing web pages as well. Table 3 below shows the results:

Table 3: Evaluation results for phishing train set

Phishing web pages				
	Gold labels			
System Labels		LFE	LFDE	Total
	LFD	29	3	32
	LFND	9	402	411
	Total	38	405	443

Accuracy: 97.3%
F-Score: 0.83
Precision: 0.91
Recall: 0.76

As we can see, there is only a subtle difference between the results obtained for benign and phishing web pages, with benign performed slightly better. The priority strategy for our algorithm is to detect web pages through identifier fields in HTML source code. We observed that the identifiers used by both benign and phishing web pages were very similar. Since the overall structure of both websites is similar as well, strategy two of Logit which locates login-related keywords produces similar results as well.

We also inspected each of the misclassifications manually to see why they had been misclassified. The most repetitive case was when the detector identified contact or query forms as login fields. This is because contact forms also contain similar fields such as those in the login ones. Our work currently does not include ways to combat misidentification of contact forms but this can be included in future works when the detector is enhanced.

Testing Phase

Similar to the train set, we next ran our detector on our test set. A key thing to note is that we used the same identifier keywords list that we collected from our train set. This was to observe how generalizable are our identifiers for web pages that are not a part of the train set. We also merged the banking web pages in our general test set of benign web pages. This was to reduce class imbalance as banking all web pages contained login fields, and thus contributed to our minority class. We also merged our sign-up and newsletter datasets since they had a similar purpose of being included in the test data. We wanted to see if these fields lead to false positives or not. Since the data for each is relatively small as well, combining both will give a better idea of the evaluation results. Moreover, these two categories of web pages did not have login fields in them.

We ran our detector on all the categories of test sets separately. The results are as follows:

Table 4: Evaluation results for benign test set

Benign web pages				
	Gold labels			
System Labels		LFE	LFDE	Total
	LFD	135	12	147
	LFND	33	649	682
	Total	168	661	829

Table 5: Evaluation results for phishing test set

Phishing web pages				
	Gold labels			
System Labels		LFE	LFDE	Total
	LFD	12	2	14
	LFND	7	70	77
	Total	19	72	91

Table 6: Evaluation results for sign-up and newsletters

Sign-up and newsletters web pages				
	Gold labels			
System Labels		LFE	LFDE	Total
	LFD	0	48 + 7	55
	LFND	0	28 + 52	80
	Total	0	76 + 59	135

* The first value in the table above comes from sign-up forms while the second refers to newsletters.

Table 7: Evaluation metrics for all test sets

Categories of test set			
	Benign	Phishing	Sign-ups and newsletters
Accuracy	94.5%	90.1%	59.2%
Precision	0.92	0.86	-
Recall	0.80	0.63	-
F-Score	0.86	0.73	-

Observing trends of misclassification

Similar to training results, we inspected each misclassified web page of the test dataset and recorded any observed trends.

Unique Identifiers

As mentioned earlier, the testing was conducted using the set of identifiers collected in the training phase. Hence, all the webpages in the test set that used a unique identifier were able to bypass the detection. An example of this case is as follows:

Figure 3: Unique identifiers in HTML

Security Measures in Banking Websites

Although the login detector achieved a good accuracy on banking websites, some escaped the detection. This is expected as banking websites are implemented with added security and privacy measures. The common trend observed was that the webpages did not allow inspection of DOM elements to take place and given that Logit is based on parsing HTML elements, it was unable to detect the login field. An example of this is shown below:

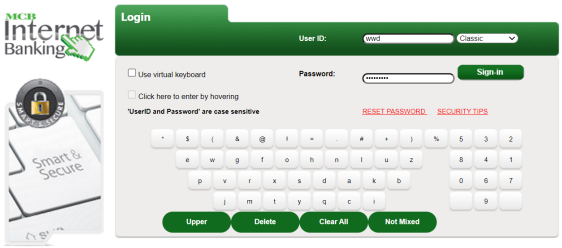


Figure 4: Security measures in banking web pages

Non-English Web Pages

The websites which were not in English often had the IDs and names in the login form fields in non-English as well, which bypassed our detector. This is because the list of identifiers that we are using is in English.

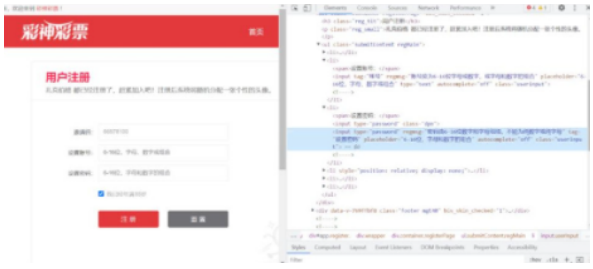


Figure 5: A web page in Chinese language

Non-login web pages that were not in English were often misclassified as login. These false positives were surprising since these pages did not have login fields. Instead, the input query boxes were incorrectly being detected as login fields.

Newsletters

Some websites containing newsletter and subscription fields were incorrectly identified as login fields. Particularly, this issue was prevalent on those web pages that took username and message as input. For newsletters that required only an email address, there were no false positives. An example of a case of misclassification is given below:

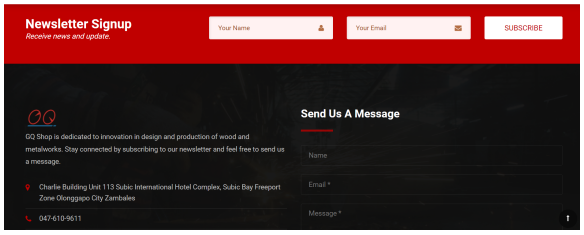


Figure 6: Newsletter form

User-Input based Dynamic Forms

As of yet, Logit does not cater to dynamic forms that are based on user input. Hence, websites containing dynamic HTML elements were not detected.

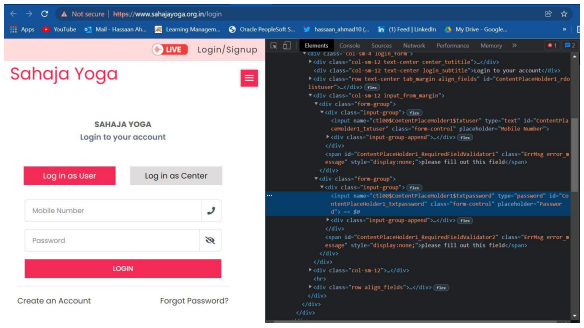


Figure 7: Dynamic Form

Logit, essentially a login detection tool, was also able to successfully identify signup web pages through a set of signup identifiers. To cater to false positives (cases when a signup page was incorrectly identified as a login page), we added a condition in the code that checked for signup fields. Logit achieved an accuracy of 63.2% in detecting signup pages. In the case of newsletter pages, Logit achieved an accuracy of 88.1% in ensuring that newsletter subscription fields were not incorrectly identified as login fields. This work can be extended by utilizing the base code of Logit to implement signup or a newsletter detector.

5. LIMITATIONS

While we aimed our best to incorporate sound measurement methodology, transparency, and reproducibility of our works, there were a few factors that limited the scope and efficiency of our research. For one, expansion of our work was limited by the lack of classified datasets for login fields that existed online. In fact as stated earlier, we did not come across a single dataset that suited our purpose. Therefore, we undertook the task of manually labeling each web page of the dataset. This was a very time consuming task that took days to complete. Secondly, our work was limited by the spontaneity and inactivity of phishing web pages. Most of the phishing web pages in our dataset were inactive, and thus had to be filtered out. From the ones that remained, their HTTP status needed to be checked regularly since some of them were taken down during the course of our research. This meant that preparing a dataset for active phishing web pages had to be done in several iterations to maintain active status. Finally, we relied on community collected datasets for phishing web pages. While we try to minimize the possibility that a web page in our phishing dataset is benign by verifying through the metrics stated in *Section 3*, there is a slight chance that a few of those web pages might not be phishing. Detecting phishing websites is itself a field of its own, and we can delve deeper into this in future works.

6. FUTURE ASPECTS

We acknowledge that there is great potential in this topic as it can be used to create bots for logging into websites for large-scale web analysis, or to study the behaviors adopted by phishers to avoid detection of login fields. To systemize the way forward, we seek to collect a larger labeled dataset of phishing and benign websites so that we can judge the performance of the detector at a larger-scale. It will also help us in expanding the list of our identifier keywords which would promote greater coverage. We then wish to carry out a large-scale empirical study of phishing websites in order to observe trends and techniques used by phishers to bypass security checks, and hide the presence of login fields. We can also use login field detection to display security messages to the users once they are on a page that requires personal credentials to be input. This will make users more mindful while submitting their credentials, and act as an added security layer while dealing with sensitive information online.

7. CONTRIBUTIONS

We strongly advocate for delivering reproducible works that can benefit the larger research community. We publicly make available the source code of Logit for the research community along with guidelines of how to run the code. We also publicly make available all labeled datasets and identifier keywords that we collected and used as part of our research. We hope that our efforts benefit the research community at large and help in advancing the topic.

Source code and data release: The source code of Logit is publicly available and can be accessed here. Datasets can be accessed here.

8. CONCLUSION

This paper presents a login detection tool, which leverages the parsing of HTML ID and name fields to detect login pages. To implement the tool, we collected and labeled a set of diverse dataset for both benign and phishing websites. The detector maintains a maximum accuracy of 79.6% when correctly identifying login pages and a maximum accuracy of 99.2% in detecting non-login web pages. The evaluation looks into identifying the trends of misclassification. It is observed that banking websites bypass detection and newsletter subscription fields lead to false positives. In addition to detection of login pages, Logit is able to detect signup pages with an accuracy of 63%.

9. REFERENCES

[1] Routhu Srinivasa Rao and Alwyn R. Pais. 2017. Detecting Phishing Websites using Automation of Human Behavior. In *Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security* (*CPSS '17*). Association for Computing Machinery, New York, NY, USA, 33–42. DOI:https://doi.org/10.1145/3055186.3055188

[2] S. Marchal, K. Saari, N. Singh and N. Asokan, "Know Your Phish: Novel Techniques for Detecting Phishing Sites and Their Targets," *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*, 2016, pp. 323-333, doi: 10.1109/ICDCS.2016.10.

[3] S. Afroz and R. Greenstadt, "PhishZoo: Detecting Phishing Websites by Looking at Them," *2011 IEEE Fifth International Conference on Semantic Computing*, 2011, pp. 368-375, doi: 10.1109/ICSC.2011.52.

[4] Lin, Yun & Liu, Ruofan & Divakaran, Dinil Mon & Ng, Jun & Chan, Qing & Lu, Yiwen & Si, Yuxuan & Zhang, Fan & Dong, Jin. (2021). Phishpedia: A Hybrid Deep Learning Based Approach to Visually Identify Phishing Webpages (USENIX Security 2021).

[5] Vrbancic G, Fister J I, Podgorelec V. Dataset for phishing websites detection. Data Brief 2020; 33: 106438

[6] Steven Van Acker, Daniel Hausknecht, and Andrei Sabelfeld. 2017. Measuring login web page security. In *Proceedings of the Symposium on Applied Computing* (*SAC '17*). Association for Computing Machinery, New York, NY, USA, 1753–1760. DOI:https://doi.org/10.1145/3019612.3019798

[7] H. Sharma, E. Meenakshi and S. K. Bhatia, "A comparative analysis and awareness survey of phishing detection tools," 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2017, pp. 1437-1442, doi: 10.1109/RTEICT.2017.8256835

[8] Biersdorfer, J, "How does my Browser Recognize Malware", *The New York Times,* Dec 2, 2016, https://www.nytimes.com/2016/12/22/technology/personaltech/how-does-my-browser-recognize-malware.html

[9] Saeed Abu-Nimeh, Dario Nappa, Xinlei Wang, and Suku Nair. 2007. A comparison of machine learning techniques for phishing detection. In *Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit* (*eCrime '07*). Association for Computing Machinery, New York, NY, USA, 60–69. DOI:https://doi.org/10.1145/1299015.1299021

[10] Dinei Florencio and Cormac Herley. 2007. Evaluating a trial deployment of password re-use for phishing prevention. In *Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit* (*eCrime '07*). Association for Computing Machinery, New York, NY, USA, 26–36. DOI:https://doi.org/10.1145/1299015.1299018

[11] Sujata Garera, Niels Provos, Monica Chew, and Aviel D. Rubin. 2007. A framework for detection and measurement of phishing attacks. In *Proceedings of the 2007 ACM workshop on Recurring malcode* (*WORM '07*). Association for Computing Machinery, New York, NY, USA, 1–8. DOI:https://doi.org/10.1145/1314389.1314391

[12] Sourena Maroofi, Maciej Korczyński, and Andrzej Duda. 2020. Are You Human? Resilience of Phishing Detection to Evasion Techniques Based on Human Verification. In *Proceedings of the ACM Internet Measurement Conference* (*IMC '20*). Association for Computing Machinery, New York, NY, USA, 78–86. DOI:https://doi.org/10.1145/3419394.3423632

[13] 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). Association for Computing Machinery, New York, NY, USA, 35–41. DOI:https://doi.org/10.1145/3375708.3380313

[14] Google, Google, https://safebrowsing.google.com/.

[15] "Digital around the World." *DataReportal*, https://datareportal.com/global-digital-overview.