2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress

PhishBox: An Approach for Phishing Validation and Detection

Jhen-Hao Li
Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan
r04921041@ntu.edu.tw

Sheng-De Wang
Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan
sdwang@ntu.edu.tw

***Abstract*— In this paper, we propose an approach, called PhishBox, to effectively collect phishing data and generate models for phishing validation and detection. The proposed approach integrates the phishing websites collection, validation and detection into an on-line tool, which can monitor the blacklist of PhishTank and validate and detect phishing websites in real-time. Due to the short life time of phishing websites, the proposed approach uses a two-stage detection model to ensure the performance. First, we design an ensemble model to validate the phishing data and apply active learning for reducing the cost of manual labeling. The result shows that our ensemble validation model can achieve high performance with 95% accuracy and 3.9% false-positive rate. Next, the validated phishing data will be used to train a detection model. Comparing with the original dataset, the false-positive rate of phishing detection is dropped by 43.7% in average. After participating the voting procedure on PhishTank, the result shows that our two-stage model is effective to verify phishing websites. Finally, we monitor the blacklist and found that the blacklist contains lots of invalid data. According to our experiment, we can remove five times more than regularly update after one week.**

Keywords—phishing validation; phishing detection; framework; machine learning; active learning

I. INTRODUCTION

Phishing attacks have become very common in our daily life as the growth of mobile and IoT devices in the Internet[1]. Although there are many tools and techniques for detecting malicious websites or preventing users from leaking their personal information, it is still a difficult thing to keep user totally safe. Phishing websites maker may not only use the social engineering skill to deceive people but also exploit some evasion techniques [2] to bypass the detection.

According to a report in Dec 2016 by the security company Webroot [3], 84% of phishing websites keep alive for less than 24 hours and some websites just appear for less than 15 minutes. Almost all of the phishing websites are hidden within the legitimate domains. It makes security companies or organizations hard to capture the information and analyze the phishing data for coming up with a counterattack strategy.

In order to solve phishing issues, research is focused on two directions: (1) establishment and maintenance of infrastructure for data collection and analysis; (2) the evolution of detection and prevention technology. PhishTank [4] provides a phishing database by using crowdsourcing, which allows users to submit and validate phishing websites for establishing a blacklist. Other public data sources include OpenPhish [5] and Phishload [6]. Based on the public database of phishing websites, further analysis can be done. For example, considering the blacklist of PhishTank, PhishMonger [7] provides an updated database of verified phishing websites. PhishNet proposes predictive blacklisting to detect phishing attacks [8, 9]. PhishStorm detects phishing websites with streaming analytics [10]. Cui et al [11] tracks and analyzes phishing attacks for duplication.

As for phishing detection, there are three main approaches including list-based [8, 12], visual-based [13, 14] and feature-based [15-17]. To achieve high performance, the weighted-ensembles of machine learning models are proposed [18]. The results of machine learning are highly related to the data sources and the selection of features. Among them, the effects of the phishing data source are more significant than the selection of features because the features that can be used are limited and had widely discussed in previous studies [17-20].

The public data sources [4-6] provide basic phishing websites information with specific features and PhishMonger presents an archive of verified phishing websites. However, the updating of PhishTank blacklist is slow such that it contains many invalid data when people crawling the phishing websites according to the blacklist. In order to evaluate the performance, it is often to use a small dataset that is verified manually or use large testing set [18]. These approaches are time-consuming and require labeled datasets. At the same time, the blacklist should be updated regularly and accurately. According to a statistic on July 2016 by PhishTank [21], the median time for verification is 41 hours. It means the voting procedure is not efficient to block the new phishing websites. Also, if the phishing websites are down, there is no clear mechanism to remove invalid phish from the blacklist. It will result in an incorrect database and also take longer for the hacked websites to be removed from the blacklist.

Since phishing websites are known to have the property of short life time, it requires an on-line and real-time validation process to ensure the up-to-date data collection. In this paper, we propose an approach to systematic process the phishing data for phishing validation and detection. Our tool has three features including fast, modular and easy to use. It has been deployed on Google Cloud Platform with the container technology. The proposed approach crawls and extracts the phishing data and then generates the verification and phishing detection models. According to the monitoring module, the system will track the status of websites on the blacklist to detect invalid phishing websites. Moreover, the verifying module will participate the

978-1-5386-1956-8/17 $31.00 © 2017 IEEE
DOI 10.1109/DASC-PICom-DataCom-CyberSciTec.2017.101

voting process on PhishTank with the proposed two-stage model to reduce the verification time.

To summarize, the contributions of this work include:

- The proposed approach can be used to collect phishing data effectively and generate models for phishing validation and detection in real-time.
- We applied active learning to reduce the cost of manual labeling and build an ensemble classifier for cleaning invalid data.
- We proposed a two-stage model for detecting phishing websites with low false-positive rates.

The rest of this paper is organized as follows: Section II reviews the related approaches. Section III presents the system architecture. Section IV presents phishing data validation and detection system. Section V shows the experiment results. The final section concludes the paper.

II. Related Work

We review some infrastructures about the phishing data collection and analysis and then give an overview on the evolution of phishing detection.

A. Infrastructure of phishing data collection and analysis

The public data sources such as PhishTank [4] or OpenPhish [5] provide a good collection for phishing websites. However, these databases might not be up-to-date since the phishing websites live for a short time and can disappear or change quickly. It is not sure that the wrong judgment is caused by the invalid data or bad voters [22]. Dobolyi and Abbasi [7] propose a free and open source public archive of phishing websites named PhishMonger to solve this issue. They deployed a crawler system on Amazon Web Service (AWS) and collected over 88754 websites including the JavaScript, CSS, font and image file in each website. Users can keep up to date with the latest phishing data.

Marchal et al. [10] focus on the data analysis with the verified data on PhishTank and propose PhishStorm using real-time analytics architecture based on Storm. They define and use the concept of intra-URL to improve phishing detection. Cui et al. [11] consider the problem of replicas and invalid data on PhishTank. According to their observation, there are over 90% of phishing attacks are the copies or variations of others in the database. Similarity, Lee et al. [9] propose a framework called PhishTrack to verify phishing websites. They leverage the components of PhishNet [8] to update the blacklist. It actively finds phishing URLs as early as possible. Different from the passive collection of phishing data, Han, Kheir and Balzarotti [23] leverage a web honeypot to attract real attackers to hack their websites and generate phishing pages. They draw the first compensative picture about phishing attack flows and give some case studies.

B. The phishing detection and prevention technology

Techniques including list-based, vision-based and feature-based approaches have been developed to detect phishing websites.

List-based phishing detection is the most intuitive idea to prevent users from visiting the websites according to the list maintained by the security team. PhishNet [8] proposes five heuristics to generate new URL variations of original one and match with the blacklist through an approximate matching algorithm. On the other hand, the automated individual white-list (AIWL) [24] keeps records of legitimate login user interface (LUI) of websites. Likewise, an anti-phishing technique using an auto update white-list is proposed by Jain et al. [12]. They analyzed the hyperlinks in the source code of the webpage.

Similar to the blacklist, the aim of visual approaches is to identify the mock target by computing the similarity of the screenshots between official and phishing websites. If they have high similarity but different URLs, it means the target would be a phishing website. Accordingly, various algorithms for calculating image similarity have been proposed, such as the Earth Mover's Distance (EMD) algorithm [25], SURF detector [26] and the Histogram of Oriented Gradients (HOG) descriptor [27] to capture cues of web page layouts with low time-consuming. Moreover, some approaches compare the style of pages. BaitAlarm [28] extracted the styles from CSS files and related elements. Zhang et al. [29] consider the informative spatial layout characteristics. To enhance the processing speed, Ardi and Heidemann propose AuntieTuna [30] by using cryptographic hashing of Document Object Model (DOM) to compare pages.

Approaches based on machine learning consider the features related to the web pages. CANTINA [31] first use the content to detect phishing websites based on the TF-IDF information retrieval algorithm. PhishWHO [15] leverage the approach and propose a three-tier matching system. As for the new kind of features, the X.509 public key certificates of the websites are used by Dong et al. [16]. Their approach increases the difficulty in evading the detection. In response to the current trend of mobile computing, the redirect information is considered by Lin et al. [17]. To understand the importance of some features, Zuhair et al. [19] investigate the prediction susceptibility of 58 hybrid features to find valuable features to enhance the detection performance.

Finally, several methods are combined for achieving high detecting accuracy. A two-stage classification model for malicious websites detecting is proposed by Le et al. [32]. Mensah et al. [13] combine SSL/TLS features and JavaScript-based visual clues. They setup a visual analysis server to detect hacked domains hosting pages. Amrutkar et al. [14] present kAYO, a fast and reliable analysis technique to identify some new features for mobile devices. Their detection model combines with the Google Safe Browsing to enhance the overall performance. To deal with the mobile device, an automated lightweight phishing scheme called MobiFish is presented by Wu et al. [33]. It compares the domain name of URL and the text extracted from screenshots by OCR (Optical Character Recognition). Marchal et al. [18] develop a phishing detection system and achieve high performance with a Gradient Boosting algorithm.

III. Architecture

In this section, we present PhishBox with an integrated architecture to handle the phishing data systematically. The

system is shown in Figure 1. Our goal is to develop a fast, scalable and easy-to-use tool. It contains five main modules including extract-transform-load (ETL), modeling, voting, monitoring and visualization.

The container technology is used to deploy PhishBox to enable the fast environment setup. At the same time, the data can be parallel downloaded with the multi-thread crawlers. Second, our programs are written in Python with some third party tools such as Sci-kit Learn, Selenium and Phantom.js. With the usage of these tools and modular design, it is easy to add new rules for feature extraction or to develop new models. Finally, users can get the insight of phishing data and do interactive operations with our web interface.

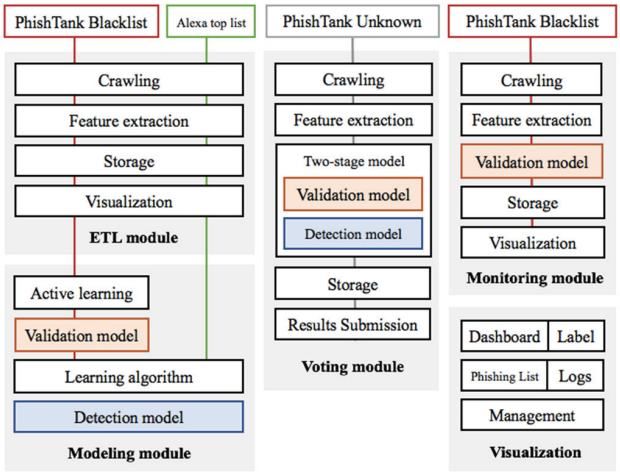

Figure 1. The system architecture of PhishBox.

A. ETL module

In order to analyze the phishing attack, the first step is to collect phishing data. According to the work proposed by Bahrami et al. [34], we maintain a pool to avoid repeating URL and use multiple crawlers to accelerate the download speed. Features and screenshots will be extracted by crawlers and stored in a server with the file and image storage and and an Mongo database. The ETL flow chart is shown in Figure 2.

The information of pages will be extracted with the feature extractor component. Features covers four categories including host information, URLs, contents and screenshots. Table 1 shows the features we collected. In addition to considering features as in the UCI phishing dataset [20] and some works [17-19], we then propose to use other features. For example, we consider the HTTP header as an important indicator. Chandran and Manoharan point out the value of HTTP protocol with compression and cache tuning for increasing the speed of page loading. It is useful to identify the phishing website by the loading speed because the performance optimization might be ignored by phishing website makers. Moreover, for the phishing validation, the layout of website is helpful for identification. We need to compare the difference of screenshots between the one we captured and the other provided by PhishTank. The similarity score will be calculated with phash [13] and histogram between two images as features.

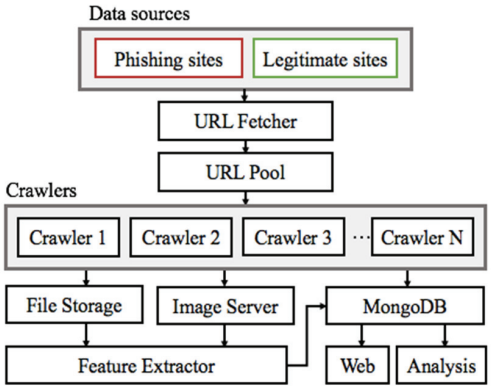

Figure 2. The architecture of ETL module.

TABLE 1. FEATURE SETS

Category	Features
Host information	HTTP response header DNS record WHOIS record IP address
URL	Starting, Landing URL Alexa rank Redirection Usage of HTTPS Length of URL, FQDN, Free URL Usage of specific symbol (&./@)
Content	Count of HTML elements JavaScript function usage Usage of icon Plain text
Screenshot	Raw image The histogram of screenshot The similarity score between screenshots

B. Voting and monitoring module

The suspicious pages will be predicted by a two-stage classifier shown in Figure 1. Our predictions will be submitted to PhishTank by the voter component with Selenium, a framework for website automation testing. Finally, we will verify the voting result of the phishing websites for evaluation.

In order to observe the changing of the blacklist, we develop a monitoring module to track phishing websites. The module will check the online status of websites and detect invalid ones with our validation model. After a while of information collection, the comparison can be done by calculating the amount of our prediction and the actually removed websites. Hence, we can understand the percentage of invalid data on the blacklist and the distribution. The lifecycle of phishing website on the PhishTank and our procedure is shown in Figure 3.

C. Visualization

For deep observation of phishing data, we designed a web-based user interface that was constructed with Node.js Express and Bootstrap framework. It provided a dashboard to display the system status, phishing information, monitoring records and logs. In addition to the messages used in the interface of PhishTank, we provide more detail information for oracles (labelers) to verify phishing websites. Oracles can see not only

the screenshot but also host information and other metrics. The interface is shown in Figure 4.

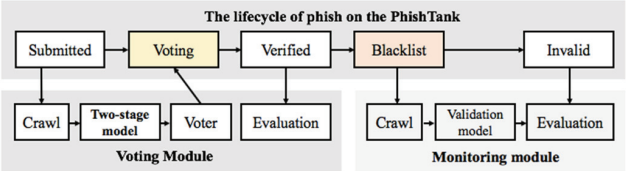

Figure 3. The lifecycle of phishing data on PhishTank blacklist and the procedure of our voting and monitoring module.

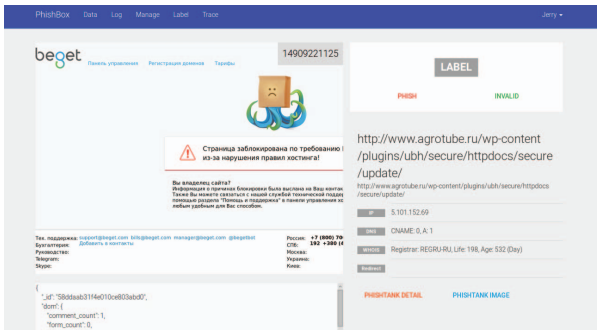

Figure 4. The interface for labeler to verify phishing websites.

IV. Phishing Validation and Detection Model

A. Validation Model

The goal of validation is to remove non-phishing websites from the blacklist. Due to the fast changing of phishing websites, the status of websites on the blacklist will not be the same all the time. Once a page encounters the following situations, we call it invalid. Figure 5 shows some example of invalid pages.

- **Offline**: the website is not reachable.
- **Redirection**: the page is redirected to the legitimate page.
- **Invalid content**: the content of the page is changed and contains invalid keyword such as “this account has been suspended” or “page not found”.

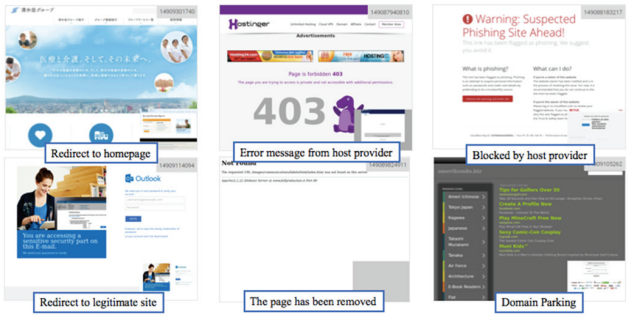

Figure 5 The examples of invalid pages

We can easily check the online status of phishing websites by visiting them and make sure the website is down. But to detect other two situations, we have to construct a validation classifier with different approaches.

As we known, the phenomenon of content changes usually occurs when the phishing websites are removed or blocked by host providers. The page will be redirected or even replaced to specific content to inform the client that this page is not available. To detect the redirection, we can measure the change of meta features including the redirect path, the content structure and the difference of screenshots. As for invalid content, we can use information retrieval algorithms to detect. Because the content is usually made up of some specific keywords. Based on the TF-IDF algorithm with constant smoothing, it removes stop words from the plain text and extracts the term frequency. For instance, some websites show the customized 404 page when the page is not found. We can detect the page easily with the frequency of keywords without setting rules in advance. Therefore, based on our observation, we have to combine three categories of features: information of contents, images and meta features with supervised classification algorithms. Finally, we ensemble the results according to the probabilities. If the mixed score is greater than the threshold, we determine the page as invalid. The rule is summarized in Figure 6.

Algorithm 1

Input: F_C: content feature set
F_M: meta feature set
F_I: image feature set
Output: prediction

$P_C = contentClassifier(F_C)$
$P_M = metaClassifier(F_M)$
$P_I = imageClassifier(F_I)$
$mix_{score} = average(P_C, P_M, P_I)$
if $(mix_{score} \geq threshold)$ **then**
  return *phish*
else
  return *invalid*
end if

Figure 6. The ensemble rule of the validation model.

B. Active learning

Active learning is a machine learning process that can take the initiative to interactively query the experts or other resources for labels. It is expected to use a small amount of training data to achieve the desired output. Hence, the idea is suitable for our validation model because there is no label available for invalid data and it costs a lot to label all of the data manually. Figure 7 shows the flow of an active learning process, where the sampling algorithm selects some sample to be labeled by an oracle.

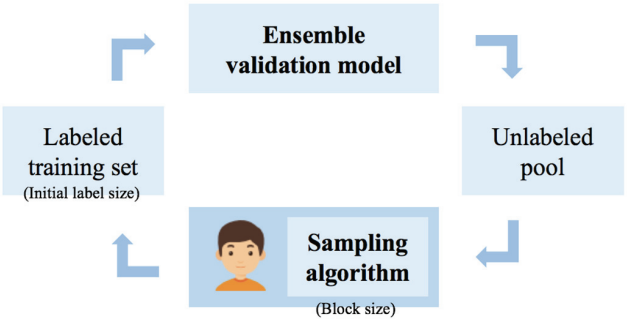

Figure 7 The flow of active learning

The process is made up of a classifier and a sampling strategy. In each iteration, we choose a block of data with an uncertainty sampling algorithm. Similar with [35], using TFIDF and random forest algorithm, we also consider the influence of image and meta features. The entropy will be calculated with the probabilities according to the current ensemble validation model. After sorting, the most uncertain data will be added. These steps will repeat until the end and compare the results with random sampling. Figure 8 shows the proposed sampling algorithm, which is based on the entropy estimation.

Algorithm 2

Input: $U = \{u_i\}_{i=1}^{n}$: unlabel data pool
B: block size
Output: sampling targets
1: $Candidates = []$
2: **for** $i = 1$ to n **do**
3: $P_{phish}, P_{invalid} = classifier(u_i)$
4: $Score = entropy(P_{phish}, P_{invalid})$
5: $Candidates[i] = Score$
6: **end for**
7: $Targets$ = sort and select top B of $Candidates$
8: **return** $Targets$

Figure 8 The proposed uncertainty sampling algorithm

In order to ensure fairness, objectivity and consistency of validation, we define the rules for oracles to determine the websites with our interface shown in Figure 9. Basically, oracles should check the screenshot first. If the web pages have errors, blocked or domain parking messages, we consider the page as invalid. While the redirection occurs, oracles should check the URL and host information to make sure it is redirected to the legitimate website or another phishing website. Finally, if it is difficult to distinguish the page, oracles have to verify the URL or the name of the website through the search engine.

Figure 9 The information of website with our interface.

C. Phishing Detection Model

The phishing detection model is trained with some learning algorithms including BayesNet, logistic regression, decision tree and random forest. We want to see the difference between using the original data and the verified phishing data after validation. Our models will be used in the voting module which we mention previously for helping to validate the phishing websites on the blacklist. The two-stage phishing detection model is combined with validation and detection model as shown in Figure 10. The validation classifier will validate the page first. If the page is predicted as legitimate, it will be checked with our phishing detection classifier. If the prediction of detection classifier is a phishing website, we will consider the page as a phish. Otherwise, we consider as a non-phish.

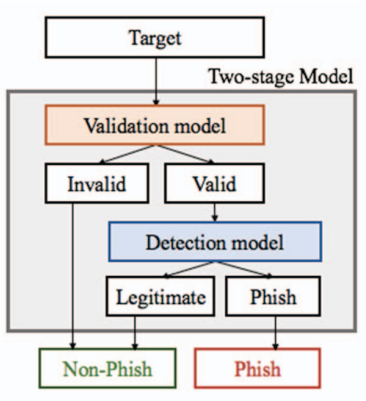

Figure 10 Two-stage phishing detection model.

V. Evaluation

A. Environment and Dataset

We obtained the data from two sources with our tool. The "valid" phishing websites were collected through Phishtank blacklist in April 2017. In the same time, the legitimate data was collected from the Alexa top list. By visiting each legitimate website, we also collected the login page. The crawlers searched all the URLs that contain keyword "login" or "sign in" in the HTML source code for downloading the page. In our collected data, there are 28607 phishing websites and 18671 legitimate websites. It turns out that there are 4213 login pages in the legitimate set.

In order to have a balanced dataset, we randomly sample data and generate subsets for different purposes. Table 2 shows the detail of each dataset. For validation, we manually label 10% of data through our interface to evaluate the performance of supervised learning algorithms. Then we adopt active learning to measure the cost saving.

Table 2. Data Sets

Purpose	Dataset	Sources	Distribution (phish: legitimate)
Validation	PhishSub	PhishTank	1200:1200
Detection	PhishLogin	PhishTank, Alexa	4000:4000
	PhishAlexa	PhishTank, Alexa	15000:15000

B. Evaluation Metrics

In other to measure the performance of algorithms, we used several evaluation metrics as follows. However, the definition of metrics is slightly different between the phish validation and the detection.

- **Accuracy**: for validation, the ratio of the number of phish and invalid websites that are correctly classified over the total number of websites in the dataset; for detection, the number of phish and legitimate websites that are correctly classified over the total number of websites in the dataset.

- **False-Positive Rate (FPR)**: the percentage of phish websites for validation or legitimate websites for detection are incorrectly classified as false-positive.
- **True-Positive Rate (TPR)**: the percentage of invalid websites are correctly classified as true-positive for validation.

C. Phishing Validation Result

Our validation model is capable of filtering out the invalid websites with high accuracy and low false-positive rates. Table 3 shows the experiment results with 10-fold cross-validation. The accuracy of the naïve Bayes algorithm is not doing well because the feature dependency is not considered. While the random forest algorithm has a better performance as it takes the ensemble result from multiple decision trees with the different feature set. Accordingly, our ensemble classifier combines three random forest classifier to improve the performance. Finally, the accuracy is 95% and the false-positive rate is 3.9%.

TABLE 3. PERFORMANCE OF VALIDATION MODEL

Approach	Method	Acc.	TPR	FPR
Content	TFIDF + Naïve Bayes	83.8	56.7	1.1
	TFIDF + Logistic Regression	89.4	90.7	11.3
	TFIDF + Random Forest	**88.0**	**80.6**	**7.9**
Meta	Naïve Bayes	75.2	49.3	10.2
	Logistic Regression	69.3	17.5	1.9
	Random Forest	**92.4**	**85.7**	**5.0**
Image	kNN	84.9	66.5	4.9
	Logistic Regression	73.2	41.9	8.9
	Random Forest	**89.3**	**80.1**	**5.6**
Ensemble	Random Forest	**95.0**	**92.9**	**3.9**

The goal of active learning is to reduce the labeling task of data. Hence, we compare our results with random sampling. The experiment is done with the 10-fold cross validation. According to Figure 11, we can see the uncertainty approach is faster to reach the maximum accuracy which is about 95%. It saves 65% of the labeling task than random sampling.

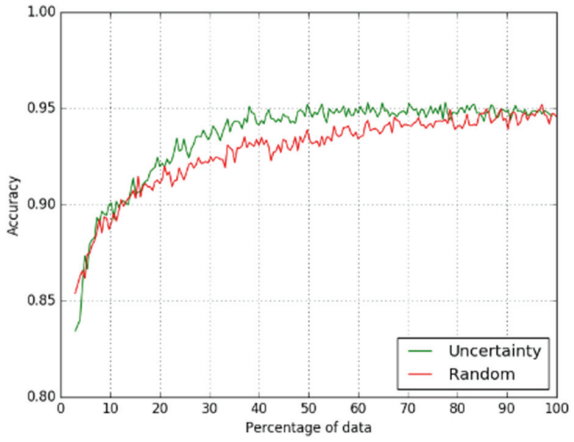

Figure 11. The accuracy with different sampling algorithm.

D. Phishing Detection Result

The result of phish detection is related to the dataset and the learning algorithm. Therefore, we conduct experiments with different settings. The result is shown in Table 4. First, we observe that the overall performance of the PhishAlexa dataset is better than PhisLogin. Pages directly fetched from the Alexa top list are usually front pages. Also, since the phishing websites are usually designed as the login page to cheat users, it will be easy to distinguish the difference. Next, we compare the effect of phishing validation applied. After removing the invalid data with the validation model, we detect phishing websites again. The result shows that the accuracy is improved in all of the algorithms and the false-positive rate drops by 43.7% in average.

TABLE 4. PERFORMANCE OF DETECTION MODEL

Dataset	Algorithm	Accuracy		FPR	
		Org.	Valid	Org.	Valid
PhishLogin	BayesNet	87.68	91.71	9.7	7.2
	LR*	91.33	93.28	9.4	5.4
	Decision Tree	92.11	93.38	7.9	4.9
	Random Forest	94.91	95.54	4.2	2.0
PhishAlexa	BayesNet	93.36	96.73	6.6	2.6
	LR*	95.46	95.56	5.5	2.7
	Decision Tree	97.70	98.16	2.6	1.6
	Random Forest	98.44	98.64	2.2	1.3

* Logistic Regression

E. Voting Result

The amount of verification is only about a quarter of submission in each day on the PhishTank. It means the crowd voting procedure is time-consuming. To accelerate the verification, we take part in the verification using the proposed tool. Hence, we submit about 18000 validation result within 2 days (May 3-4, 2017) to see the difference. According to the daily phishes verified status, we increased the amount of verification about 8 times than normal. After 5 days of tracking, 6181 websites (34% of our submission) have been verified with others. However, the verified result is not reliable because everyone sees the phishing websites at the different time with the different environment. The page could be invalid while the phish maker blocks client's IP or the page only shown by redirecting from the specific domain. Hence, in order to measure the performance of phishing detection, we should remove the invalid data and calculate the performance again. As shown in Table 5, the accuracy is 95% and the false-positive rate is 8.5%.

TABLE 5. PERFORMANCE OF ENSEMBLE MODEL

Method	Acc.	TPR	FPR
Two-stage model	74.9	69.0	1.8
Two-stage model w/o invalid data	95.0	95.3	8.5

F. Monitoring Result

It is important to remove the invalid phish properly on the blacklist. Therefore, we conduct experiments that monitor the status of phishing websites on PhishTank. According to the description, the blacklist will update in every hour. However, the amount of change is small. To have an efficient inspection, we set the monitoring interval as 12 hours and continue to visit these websites. With our observation, only about 60% of "verified"

websites are still phishing websites when we visit them. It means that the blacklist contains lots of invalid data.

The cumulative amount of phishing websites is shown in Figure 12. From the beginning, the total phishing websites are 30297 and among them 12550 invalid websites have been detected. In each time interval, we calculate the cumulative amount that was removed from the blacklist by PhishTank. After one week of monitoring, we can remove five times more than regularly updates. Also, 59% of websites removed by PhishTank because that the hosts are not reachable (offline). We think they don't consider the invalid situation of the phishing websites.

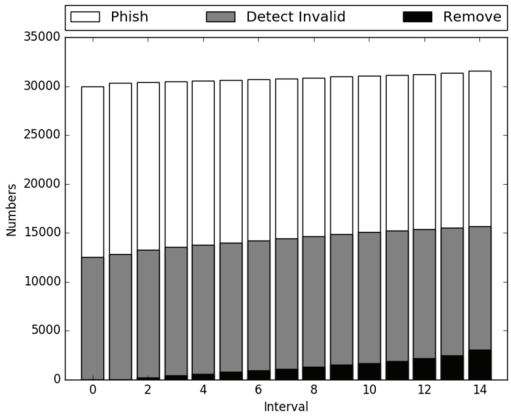

Figure 12. The cumulative amount of phishing websites.

G. Comparasion and Discussions

Many frameworks have been proposed against phishing attacks. The main difference of our approach is that we integrate data collection, validation, modeling and detection, monitoring and visualization. The modular design enables the separation of data analysis and the infrastructure. As compared with PhishMonger, our crawlers not only download archive files but also automatically extract features. The URL pool is flexible for collecting data. The data source may not limit to PhishTank. However, third party tools are not stable sometimes. For instance, the phantom.js framework which we used to take screenshots will crash due to unknown reasons. In the meantime, since some phishing websites update the webpage with JavaScript dynamically, we cannot guarantee that the page we have downloaded is complete. Fortunately, these issues don't cause much trouble after we validate the data. The invalid data will be removed and the missing part can be filled with appropriate information.

PhishTrack does the validation by applying SVM classifiers with a natural language processing (NLP) approach. Since we use a large dataset for training and consider more features on the websites, the proposed ensemble model is effective to validate phishing websites. The proposed ensemble model can apply in the voting procedure and the online monitoring. Cui et al. [11] monitor the blacklist over ten months to track phishing attacks. They focused on detecting the replicas by calculating the DOM similarity. We are doing the same thing for data cleaning, while we focus on removing invalid websites. The proposed approach can have a systematic fight against phishing websites.

VI. Conclusions

In this paper, we present a phishing validation and detection approach. It can effectively collect phishing data, build models, monitor the blacklist of PhishTank and detect phishing websites in real time. For the validation, we applied active learning to reduce the cost of manual labeling and built an ensemble model to remove invalid phishing websites. In the detection phishing websites, we proposed a two-stage classifier that combines the validation model and the detection model. The result shows that our classifier is effective to detect phishing websites. Finally, we monitor the blacklist and found lots of invalid data should be removed.

References

[1] B. B. Gupta, A. Tewari, A. K. Jain, and D. P. Agrawal, "Fighting against phishing attacks: state of the art and future challenges," *Neural Computing and Applications,* pp. 1-26, 2016.

[2] B. Liang, M. Su, W. You, W. Shi, and G. Yang, "Cracking Classifiers for Evasion: A Case Study on the Google's Phishing Pages Filter," presented at the Proceedings of the 25th International Conference on World Wide Web, Montreal, Quebec, Canada, 2016.

[3] Webroot. *Webroot Quarterly Threat Update: 84% of Phishing Sites Exist for Less Than 24 hours*. Available: https://www.webroot.com/us/en/about/press-room/releases/quarterly-threat-update-about-phishing

[4] *PhishTank*. Available: https://www.phishtank.com/

[5] *OpenPhish*. Available: https://openphish.com/

[6] *Phishload*. Available: http://www.medien.ifi.lmu.de/team/max.maurer/files/phishload/

[7] D. G. Dobolyi and A. Abbasi, "PhishMonger: A free and open source public archive of real-world phishing websites," in *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*, 2016, pp. 31-36.

[8] P. Prakash, M. Kumar, R. R. Kompella, and M. Gupta, "PhishNet: Predictive Blacklisting to Detect Phishing Attacks," in *2010 Proceedings IEEE INFOCOM*, 2010, pp. 1-5.

[9] L.-H. Lee, K.-C. Lee, H.-H. Chen, and Y.-H. Tseng, "POSTER: Proactive Blacklist Update for Anti-Phishing," presented at the Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, Arizona, USA, 2014.

[10] S. Marchal, J. François, R. State, and T. Engel, "PhishStorm: Detecting Phishing With Streaming Analytics," *IEEE Transactions on Network and Service Management,* vol. 11, no. 4, pp. 458-471, 2014.

[11] Q. Cui, G.-V. Jourdan, G. V. Bochmann, R. Couturier, and I.-V. Onut, "Tracking Phishing Attacks Over Time," presented at the Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 2017.

[12] A. K. Jain and B. B. Gupta, "A novel approach to protect against phishing attacks at client side using auto-updated white-list," *EURASIP Journal on Information Security,* journal article vol. 2016, no. 1, p. 9, 2016.

[13] P. Mensah, G. Blanc, K. Okada, D. Miyamoto, and Y. Kadobayashi, "AJNA: Anti-phishing JS-based Visual Analysis, to Mitigate Users' Excessive Trust in SSL/TLS," in *2015 4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)*, 2015, pp. 74-84.

[14] C. Amrutkar, Y. S. Kim, and P. Traynor, "Detecting Mobile Malicious Webpages in Real Time," *IEEE Transactions on Mobile Computing,* 2016.

[15] C. L. Tan, K. L. Chiew, K. Wong, and S. N. Sze, "PhishWHO: Phishing webpage detection via identity keywords extraction and target domain name finder," *Decision Support Systems,* vol. 88, pp. 18-27, 8// 2016.

[16] Z. Dong, A. Kapadia, J. Blythe, and L. J. Camp, "Beyond the lock icon: real-time detection of phishing websites using public key certificates," in *2015 APWG Symposium on Electronic Crime Research (eCrime)*, 2015, pp. 1-12.

[17] I.-C. Lin, Y.-L. Chi, H.-C. Chuang, and M.-S. Hwang, "The Novel Features for Phishing Based on User Device Detection," *JCP,* vol. 11, no. 2, pp. 109-115, 2016.

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

[19] H. Zuhair, A. Selamat, and M. Salleh, "New Hybrid Features for Phish Website Prediction," *International Journal of Advances in Soft Computing & Its Applications,* vol. 8, no. 1, 2016.

[20] M. Lichman, "UCI Machine Learning Repository," ed, 2013.

[21] *PhishTank Statistic*. Available: https://www.phishtank.com/stats/2016/07/

[22] T. Moore and R. Clayton, "Evaluating the Wisdom of Crowds in Assessing Phishing Websites," in *Financial Cryptography and Data Security: 12th International Conference, FC 2008, Cozumel, Mexico, January 28-31, 2008. Revised Selected Papers*, G. Tsudik, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 16-30.

[23] X. Han, N. Kheir, and D. Balzarotti, "PhishEye: Live Monitoring of Sandboxed Phishing Kits," presented at the Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 2016.

[24] Y. Cao, W. Han, and Y. Le, "Anti-phishing based on automated individual white-list," presented at the Proceedings of the 4th ACM workshop on Digital identity management, Alexandria, Virginia, USA, 2008.

[25] A. Y. Fu, L. Wenyin, and X. Deng, "Detecting Phishing Web Pages with Visual Similarity Assessment Based on Earth Mover's Distance (EMD)," *IEEE Transactions on Dependable and Secure Computing,* vol. 3, no. 4, pp. 301-311, 2006.

[26] R. S. Rao and S. T. Ali, "A Computer Vision Technique to Detect Phishing Attacks," in *2015 Fifth International Conference on Communication Systems and Network Technologies*, 2015, pp. 596-601.

[27] A. S. Bozkir and E. A. Sezer, "Use of HOG descriptors in phishing detection," in *2016 4th International Symposium on Digital Forensic and Security (ISDFS)*, 2016, pp. 148-153.

[28] J. Mao, P. Li, K. Li, T. Wei, and Z. Liang, "BaitAlarm: Detecting Phishing Sites Using Similarity in Fundamental Visual Features," in *2013 5th International Conference on Intelligent Networking and Collaborative Systems*, 2013, pp. 790-795.

[29] W. Zhang, H. Lu, B. Xu, and H. Yang, "Web phishing detection based on page spatial layout similarity," *Informatica,* vol. 37, no. 3, p. 231, 2013.

[30] C. Ardi and J. Heidemann, "Auntietuna: Personalized content-based phishing detection," 2016.

[31] Y. Zhang, J. I. Hong, and L. F. Cranor, "Cantina: a content-based approach to detecting phishing web sites," presented at the Proceedings of the 16th international conference on World Wide Web, Banff, Alberta, Canada, 2007.

[32] V. L. Le, I. Welch, X. Gao, and P. Komisarczuk, "Two-Stage Classification Model to Detect Malicious Web Pages," in *2011 IEEE International Conference on Advanced Information Networking and Applications*, 2011, pp. 113-120.

[33] L. Wu, X. Du, and J. Wu, "Effective Defense Schemes for Phishing Attacks on Mobile Computing Platforms," *IEEE Transactions on Vehicular Technology,* vol. 65, no. 8, pp. 6678-6691, 2016.

[34] M. Bahrami, M. Singhal, and Z. Zhuang, "A cloud-based web crawler architecture," in *2015 18th International Conference on Intelligence in Next Generation Networks*, 2015, pp. 216-223.

[35] D. DeBarr and H. Wechsler, "Spam detection using clustering, random forests, and active learning," in *Sixth Conference on Email and Anti-Spam. Mountain View, California*, 2009.