Phishpedia: A Hybrid Deep Learning Based Approach to Visually Identify Phishing Webpages

Yun Lin and Ruofan Liu, *National University of Singapore;* Dinil Mon Divakaran, *Trustwave;* Jun Yang Ng and Qing Zhou Chan, *National University of Singapore;* Yiwen Lu, Yuxuan Si, and Fan Zhang, *Zhejiang University;* Jin Song Dong, *National University of Singapore*

https://www.usenix.org/conference/usenixsecurity21/presentation/lin

This paper is included in the Proceedings of the 30th USENIX Security Symposium.

August 11–13, 2021

978-1-939133-24-3

Phishpedia: A Hybrid Deep Learning Based Approach to Visually Identify Phishing Webpages

Yun Lin[1], Ruofan Liu[1]*, Dinil Mon Divakaran[2], Jun Yang Ng[1], Qing Zhou Chan[1],
Yiwen Lu[3], Yuxuan Si[3], Fan Zhang[3], Jin Song Dong[1]

School of Computing, National University of Singapore[1]

{dcsliny, dcslirf}@nus.edu.sg, {ng.junyang, chanqingzhou}@u.nus.edu, dcsdjs@nus.edu.sg

Trustwave[2]*; dinil.divakaran@trustwave.com*

College of Computer Science and Technology, Zhejiang University[3]*; {3160102248, 3170105952, fanzhang}@zju.edu.cn*

Abstract

Recent years have seen the development of phishing detection and identification approaches to defend against phishing attacks. Phishing detection solutions often report binary results, i.e., phishing or not, without any explanation. In contrast, phishing identification approaches identify phishing webpages by visually comparing webpages with predefined legitimate references and report phishing along with its target brand, thereby having explainable results. However, there are technical challenges in visual analyses that limit existing solutions from being effective (with high accuracy) and efficient (with low runtime overhead), to be put to practical use.

In this work, we design a hybrid deep learning system, Phishpedia, to address two prominent technical challenges in phishing identification, i.e., (i) accurate recognition of identity logos on webpage screenshots, and (ii) matching logo variants of the same brand. Phishpedia achieves both high accuracy and low runtime overhead. And very importantly, different from common approaches, Phishpedia does not require training on any phishing samples. We carry out extensive experiments using real phishing data; the results demonstrate that Phishpedia significantly outperforms baseline identification approaches (EMD, PhishZoo, and LogoSENSE) in accurately and efficiently identifying phishing pages. We also deployed Phishpedia with CertStream service and discovered 1,704 new real phishing websites within 30 days, significantly more than other solutions; moreover, 1,133 of them are not reported by any engines in VirusTotal.

1 Introduction

Phishing, an important step in an attack chain, has evolved over the past years to such an extent that it is now available and delivered as a service [19, 49, 71]. As per recent reports [26], the price of phishing kits more than doubled from 2018 to 2019, making them the "new bestseller" in the dark market. It is thus not surprising that phishing attacks soared by 4-5 times during the COVID-19 pandemic [3]. Meanwhile researchers have been developing new and different solutions to detect phishing pages. We classify them broadly as *phishing detection* and *phishing identification* approaches.

Phishing detection solutions are often based on dynamic black lists, or supervised machine learning models that are trained on datasets with ground truth. While some phishing detection models use only URLs (for training and predicting) [22, 27, 36, 76], others additionally use HTML contents for feature extraction [31, 39, 44–46, 62, 63, 79, 80, 82]. They suffer from three fundamental limitations: (i) biased phishing datasets used for training leads to biased models, (ii) keeping the model up-to-date requires continuous supply of large labelled phishing datasets, and (iii) there is no explanation for the predicted results. In addition, note that similar looking webpages can be rendered via very different HTML scripts. This leads to technical challenges in inferring the visual semantics of webpages, affecting detection accuracy. Besides, attackers can easily adopt evasion techniques for deceiving such solutions [38, 68].

In contrast, *phishing identification* solutions maintain a reference set of brands (or their webpages) targeted by phishing attacks; based on such a legitimate reference database a model is built. Subsequently, in operation, if the model predicts that a given webpage is similar to that of a specific brand in the reference database, but yet has a domain name that is different from the identified brand, then the webpage is classified as a phishing page [11, 13, 21, 46, 59, 74]. The goal of phishing identification models is to go beyond detecting phishing pages, and also identify the phishing targets.

Some of the early phishing identification proposals compare the screenshot of a given webpage to the screenshots of all the webpages in the reference database. For example, Fu *et al.* [21] propose to compute the similarity of screenshots of two webpages using Earth Mover's Distance (EMD) technique. However, such an approach is limited by the fact that webpages and their contents are dynamic and also updated frequently [20, 64]. This results in lower accuracy; in addition, the computational overhead increases with the increase in

*Ruofan Liu shares equal contribution with the first author for this work.

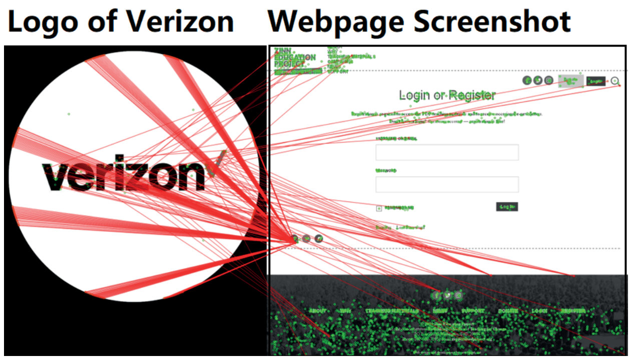
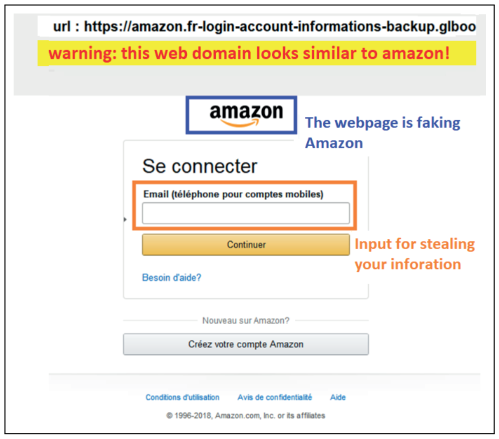


Figure 1: Problem of SIFT-based identification approach. It takes a logo (left) and a screenshot (right), and checks whether the screenshot contains such a logo. SIFT first extracts feature points from the logo and the screenshot, then matches their feature points to recognize whether the given logo appears in the screenshot. The red lines between the logo and the screenshot show the matching relations between the respective feature points. In this figure, the Verizon logo does not appear in the screenshot. However, SIFT matches many irrelevant feature points and reports a wrong match.

the number of referenced screenshots (see Section 5.2.3 for our experimental evaluations). More recent works therefore moved to the use of the very identity of brands — logos — for the purpose of phishing identification [11, 13, 16, 74]. Comparison of logos of a suspicious website to that of the brands in a reference database is tolerant to variations in webpages and their designs. Besides, with the advent of techniques such as Scale-Invariant Feature Transform (SIFT), it is possible to compare images that have differences in scale and orientation. However, SIFT-based approaches [11, 74] are not only computationally expensive (our experiments show that it takes around 19 seconds for processing each screenshot; see Table 2), but are also inaccurate. As illustrated in Figure 1, SIFT often does not extract the relevant feature points to match reference logos. This is also reflected in our experimental evaluations (Section 5.2.3).

Addressing the limitations of the current state-of-the-art research on phishing identification, in this work we propose Phishpedia, a practical and explainable phishing identification system. We design Phishpedia as a hybrid deep learning system which consists of two pipelined deep learning models for identifying phishing pages. More specifically, we decompose the phishing identification problem into i) an identity logo recognition problem, and (ii) a brand recognition problem. We address the former with customized object detection model and the latter with a transfer-learning based Siamese model. The hybrid deep learning system allows Phishpedia to achieve high accuracy in identifying phishing attempts and their targets. And very importantly, Phishpedia achieves this without requiring any phishing dataset for training the models, thus avoiding potential biases in phishing samples (see discussion in Section 6.4). Besides, Phishpedia also provides explain-


url : https://amazon.fr-login-account-informations-backup.glboo

warning: this web domain looks similar to amazon!

amazon

The webpage is faking Amazon

Se connecter

Email (téléphone pour comptes mobiles)

Continuer

Input for stealing your inforation

Besoin d'aide?

Nouveau sur Amazon?

Créez votre compte Amazon

Conditions d'utilisation Avis de confidentialité Aide

© 1996-2018, Amazon.com, Inc. or its affiliates

Figure 2: Screenshot of Phishpedia, highlighting the identity logo annotated with the phishing target brand and the input boxes for providing user credentials. It also generates a warning of how the attacker is disguising the domain name.

able visual annotations on the phishing page screenshot (see Figure 2 for a sample output from our system). Furthermore, since deep-learning based image recognition solutions are prone to evasion attacks [24, 25, 48], we also incorporate a gradient masking technique on to Phishpedia to counter adversarial attacks (Section 3.3). Finally, given a screenshot and its URL, Phishpedia predicts within 0.2 second, which also makes it more practical than existing solutions.

We conduct comprehensive experiments to evaluate Phishpedia. First, we compare Phishpedia with state-of-the-art phishing identification approaches (i.e., EMD, PhishZoo, and LogoSENSE) using six months of phishing URLs obtained from OpenPhish premium subscription. The experiments show that Phishpedia significantly outperforms the baseline approaches in terms of identification accuracy and runtime overhead. Second, we show that our hybrid deep learning system is able to defend some well-known gradient-based adversarial attacks such as DeepFool [48], JSMA [24], StepLL [34], and FGSM [35]. Third, we conduct a phishing discovery experiment where we run Phishpedia with five phishing detectors/identifiers to look for new phishing webpages in the wild. The results show that Phishpedia has a huge performance advantage over the baselines in discovering new phishing pages on the Internet. In comparison to other solutions, Phishpedia reports much more phishing webpages and with much less false positives — Phishpedia discovered 1,704 phishing webpages within 30 days and 1,133 of them are not detected by any engines in VirusTotal [9]. Moreover, 74.6% of them were not reported by VirusTotal even after one week.

We summarize our contributions in this work:

- We propose a phishing identification system Phishpedia, which has high identification accuracy and low runtime overhead, outperforming the relevant state-of-the-art identification approaches.

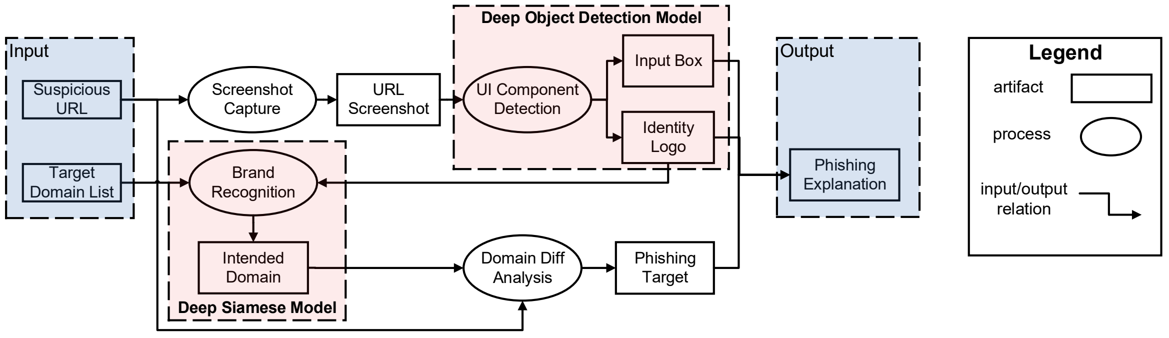


Figure 3: Phishpedia framework: a hybrid deep learning system consisting of pipelined object detection model and Siamese model (highlighted red boxes).

- We prototype our Phishpedia system, which provides explainable annotations on webpage screenshot for explaining the phishing report, facilitating its practical use. For example, with Phishpedia, a security analyst or a user has readily available easy explanations that tell why a page is classified as a phishing attempt.

- We conduct a systematic evaluation of Phishpedia using six months of phishing URLs obtained from OpenPhish (Premium service). The experiments demonstrate the effectiveness and efficiency of our proposed system. Besides, Phishpedia discovers 1,704 real phishing webpages within 30 days.

- To the best of our knowledge, we collected the largest phishing dataset for evaluating *phishing identification solutions* (i.e., including phishing brand information). We publish two datasets [7] for cyber-security and AI community: (i)$\sim$30K phishing webpages with their phishing brands, screenshots and HTML contents, and (ii) the labelled identity logos in over 30K webpage screenshots.

2 Overview of Phishpedia

2.1 Threat model

The threat model considered in this work is the following. An attacker constructs a fake webpage $\overline{W}$ that disguises as a legitimate website W of a particular brand (e.g., Paypal). The constructed webpage $\overline{W}$ has a user interface, more specifically, a form with *input boxes*, that allows a user to input credential information (e.g., username, password, bank account details, etc.). The attacker then sends the URL of the webpage $\overline{W}$ to many users, via e-mail, social networks, etc. A user obtaining such a link becomes a victim when she clicks on the URL of this phishing page and provides sensitive account information corresponding to the legitimate website W. Our goal is to detect such a phishing webpage, identify the target brand, and generate intuitive annotations for explaining the reason(s) for classifying the webpage as a phishing page.

2.2 Overview

Figure 3 provides an overview of our proposed system, Phishpedia. Phishpedia takes as input a URL and a target brand list describing legitimate brand logos and their web domains; it then generates a phishing target (if the URL is considered as phishing) as output. We refer to the logo that identifies with the legitimate brand as the *identity logo* of that brand. Moreover, *input boxes* are the small forms where a user inputs credential information such as username and password.

Given a URL, we first capture its screenshot in a sandbox. Then, we decompose the phishing identification task into two: an object-detection task and an image recognition task. First, we detect important UI components, specifically *identity logos* and *input boxes*, in the screenshot with an object detection algorithm [57,58] (Section 3.1). As the next step, we identify the phishing target by comparing the detected identity logo with the logos in the target brand list via a Siamese model [33] (Section 3.2). Once a logo in the target brand list (e.g., that of Paypal) is matched, we consider its corresponding domain (e.g., paypal.com) as the intended domain for the captured screenshot. Subsequently, we analyze the difference between the intended domain and the domain of the given URL to report the phishing result. Finally, we combine the reported identity logo, input box, and phishing target to synthesize a visual phishing explanation (as shown in Figure 2).

3 Design and development of Phishpedia

3.1 Detection of UI components

We first explain some important concepts. An object detection model takes as input an image and generates a set of *bounding boxes* to annotate the position and size of the objects on the image. In our problem setting, the image is a webpage screenshot and objects of interest are either logos or input boxes. The model is to generate a bounding box for each object (i.e., logo or input box) with a confidence score.

We analyze multiple solutions for detecting the position and shape of a logo and input box [83], and we select Faster-

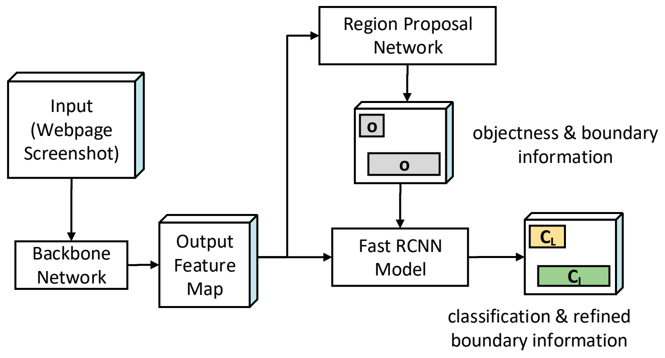

Input (Webpage Screenshot)
Backbone Network
Output Feature Map
Region Proposal Network
objectness & boundary information
Fast RCNN Model
classification & refined boundary information

Figure 4: Faster-RCNN model for logo/input box detection

RCNN model [58] to solve this problem as it is best in meeting our requirement of reporting logos completely. We compare Faster-RCNN model with other candidates in Section 5.5. We briefly describe its network structure for explaining how we apply it to detect logos and input boxes.

Figure 4 presents the network structure of Faster-RCNN; it is a two-stage object detection model, consisting of a region proposal network (RPN) [58] and a Fast RCNN model [23]. Given an input screenshot, we use a backbone network (e.g., Resnet50 [28]) to transform the input screenshot into a feature map of shape $M \times M \times c$, where M denotes the size of the feature map and c denotes the channel size. Taking the feature map as input, Faster-RCNN uses RPN to predict a set of *bounding boxes* on the input screenshot, presenting a set of "objects" for the screenshot. As shown in Figure 4, for each bounding box (grey rectangles), RPN will report an *objectness score* to indicate its probability of containing an object (i.e., UI component in our settings) and its shape. Then, the Fast-RCNN model takes the input of the output feature map and bounding boxes to (i) predict the object class (i.e., logo or input box) and (ii) refine shape and size of each object. Readers can refer to [58] for more details of Faster-RCNN.

As a result, given a screenshot, the Faster-RCNN model reports a set of candidate logos $L = \{l_1, l_2, ..., l_n\}$; each l_i ($i \in [1, n]$) is attached with a confidence score. We rank the logos by their score and take the top one as the identity logo.

3.2 Brand recognition

The target brand list consists of multiple brands considered for phishing identification. For each brand, we maintain a list of logo variants and a list of legitimate domains, for two reasons. First, maintaining multiple brand logo variants allows us to match logo images in a more precise and flexible way. Second, a brand can correspond to multiple legitimate domains. For example, the brand Amazon can have domains such as "amazon.com" and "amazon.jp.co". Capturing such information allows us to reduce false positives.

Given a reported identity logo l, if its similarity with a logo l_t in the target brand list is higher than a predefined threshold θ, then we report the corresponding brand as the phishing target brand. In general, how accurately can we recognize the brand logo we detect, partially depends on the number of brands under protection. Since there are many brands in the world, the general brand recognition may require a very long target brand list. However, we argue that the length of the target brand list is not necessarily that long. First, our empirical study on around 30K phishing webpages based on OpenPhish feed (Section 5.1) shows that the top 100 brands cover 95.8% phishing webpages; see Figure 5. This empirical result is aligned with the intuition that for phishing activities to be profitable, the attackers would have to target well-known large enterprises or financial entities [47]. Besides, a user can add new brands along with their logos and domain names (e.g., local banks) to customize the protected target list.

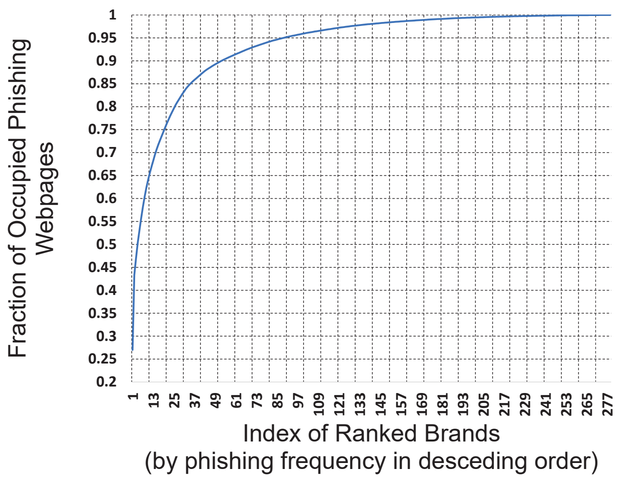


Figure 5: Phishing target brand distribution (CDF), based on ~30K collected phishing webpages. The top 5 phishing brands are Microsoft (7962), Paypal (4811), Chase Personal Banking (1085), Facebook (993), and Amazon (807).

Logo comparison. The key technical challenge here is to estimate the similarity of two logos. One straightforward solution is to consider logo recognition as an image classification task, where the input is a logo image and the output is its brand. However, image classification models have two inherent drawbacks. First, classification models cannot support adding a new brand in the target brand list during runtime [56]. Once we add a new brand, we need to retrain the whole network. Second, and more importantly, classification models need us to pre-define classes (i.e., brands in our settings). Thus, given a logo with an unseen brand in the training dataset, the model will always classify it as one of existing brands, which can cause large false positives in real-world application.

In this work, we choose Siamese neural network model [18, 33, 70] to address the above challenges. In general, a Siamese neural network model transforms an image into a representative vector; thus the similarity of two images can be estimated by the similarity of their representative vectors (e.g., cosine similarity). Typically, a Siamese model is trained by feeding the model with a *pair of images*. A positive sample is a pair of images of the same class and a negative sample is a pair of images of different classes. Then, a loss function (e.g., Triplet loss [65]) that predicts high scores for positive samples and low scores negative samples is employed.

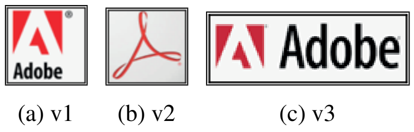


Figure 6: Logo Variants for Adobe Brand

However, our experiments show that training a Siamese model for comparing logos through the above conventional way is ineffective. The conventional training procedure for Siamese model selects three images $\langle I_{c_1}, I'_{c_1}, I_{c_2}\rangle$ as a sample to calculate Triplet loss [65], where I_{c_1} and I'_{c_1} belong to class c_1, and I_{c_2} belongs to class c_2. The goal of training is to make sure the similarity of images in the same class ($sim(I_{c_1}, I'_{c_1})$) should be larger than that in different classes ($sim(I_{c_1}, I_{c_2})$). The challenge in applying such training procedure lies in the fact that the logos under the same brand can be very different (e.g., Figure 6). It is difficult to force the model to learn similar representative vectors for different logo variants of the same brand (e.g., Figure 6a and Figure 6b). Indeed, from the experiments we carried out, we observe that if we force the model to achieve this challenging goal, it incurs the side effect of predicting brands of different classes as similar. Readers can refer to Section 5.5 for the performance of conventional Siamese model training procedure.

In this work, we leverage transfer learning [53, 54] to address the above challenges. As shown in Figure 7, we first design a logo classification task so that the backbone network (e.g., Resnetv2 network [29]) captures the features of logo images. Through the classification task, we allow the model to extract different features from different logo variants of the same brand. We connect the backbone network with a fully connected network with one hidden layer. We use Logo2K+ dataset [75] to train this task for classifying 2341 brands. Then, we take the backbone network as a base and connect it with a global average pooling (GAP) layer to construct a representative vector of 2048 dimensions. The GAP layer aggregates the feature map output from backbone network and represents semantic features of logo images. Thus, different representative vectors are learned for very dissimilar logo variants (e.g., those in Figure 6). Without enforcing the model to learn a unified representative vector for dissimilar images, we avoid the risk of introducing false logo-matching results. We then compute the cosine similarity of the representative vectors of two logo images as their similarity.

Finally, to perform the logo brand classification task, we observe that we can optionally apply a fine-tuning of the model training to make the Siamese model more adaptive to the protected logos in the target brand list. Assume the size

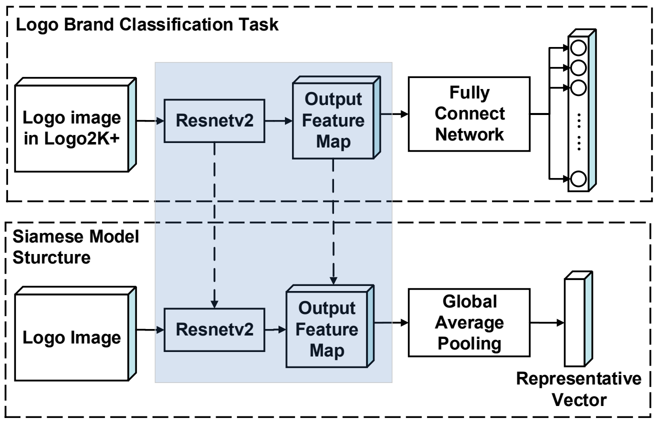


Figure 7: Transfer Learning Task

of target list is n; after training the model on the Logo2K+ dataset, we can replace the fully connected layer with another fully connected layer with n output neurons, corresponding to the number of target brands. Thus, we can train the model specifically for the brands in the target list. Our experiment (see Section 5.5) shows that such an optional training process can improve the logo recognition while still preserving the flexibility of adding unseen new logos in the target brand list.

3.3 Defending against adversarial attacks

Deep learning models are known to be vulnerable to adversarial attacks [15]. State-of-the-art adversarial attacks are designed for both object detection models (e.g., DAG [78]) and classification models (e.g., DeepFool [48] and FGSM [25]). Let a neural network be a function $f(x)$, x being a sample. Generally, most gradient-based approaches carry out attacks based on the partial derivative $\frac{\partial f}{\partial x}$, to find the minimum perturbation δ on x for obtaining $x' = x + \delta$, such that the targeted model can be deceived; i.e., $f(x') \neq f(x)$.

Traditional defense techniques against adversarial attacks usually adopt various adversarial training approaches [37, 51, 66, 72]. However, adversarial training approaches also lowers the original model's performance and they may not work well for some unseen adversarial samples [66, 72]. Instead, we design a new simple adversarial defense technique to transform our Faster-RCNN and Siamese model to counter some of the well-known gradient-based adversarial attacks, while (i) still preserving the model performance, and (ii) not requiring additional (adversarial) training that increases the system complexity.

Specifically, we replace the ReLU function in some layers of both models with a *step ReLU* function. In this approach, we design the step-ReLU function as Equation 1, where the linear function of the traditional ReLU is replaced with a step function; Figure 8 illustrates this. The parameter α determines the gap size in the step function.

$$f(x) = max(0, \alpha \cdot \lceil \frac{x}{\alpha} \rceil) \quad (1)$$

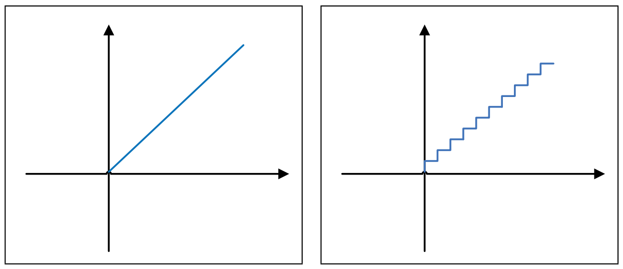

(a) Traditional ReLU (b) Step ReLU

Figure 8: ReLU v/s Step-ReLU Activation Functions

The insight here is that the partial derivative $\frac{\partial f}{\partial x}$ of step-relu is either 0 or infinite, which reduces the effect of gradient-based attacks such as DeepFool [48], JSMA [24], StepLL [34], and FGSM [35]. Moreover, the transformed layers of ReLU activation function can largely preserve the precision of the output values of activation function, which in turn helps in preserving the performance of the original network model.

4 Implementation

We build Phishpedia on top of the components described in the previous section. We select 181 brands in our target list as these are the most popular phishing targets covering 99.1% of phishing attacks according to our empirical study (see Figure 5). It is worth recalling that, Phishpedia requires no phishing dataset for training.

Object detection model. We train our Faster-RCNN model based on Detectron2 framework [77]. Different from the original Faster-RCNN model [58] which trains region proposal network and Fast-RCNN model interchangeably, our adopted Detectron2 framework uses four feature pyramid layers and trains both models jointly for better training efficiency. The dataset used for training our model is described in Section 5.1.

Siamese model. We train our Siamese model via PyTorch framework. We choose Resnetv2 [29] as the backbone network. We use Logo2k+ dataset [75] for training the brand classification task as base model for transfer learning.

Both neural networks are trained on an Ubuntu16.04 server with Xeon Silver 4108 (1.8GHz), 128G DDR4 RAM, and NVIDIA Tesla V100 GPU. All experiments for evaluations (Section 5) are conducted on the same server.

5 Performance evaluation

Next, we carry out comprehensive experiments to answer the following research questions:

- **RQ1:** How accurate is Phishpedia in identifying phishing pages, in comparison to state-of-the-art baselines?
- **RQ2:** What is the accuracy of the core components of Phishpedia, namely, the object detection model and the Siamese model?
- **RQ3:** How does Phishpedia perform if the target brand list is added with new logos during runtime (in other words, when Phishpedia is presented new logos not seen by the trained Siamese model)?
- **RQ4:** What are the alternative technical options for Phishpedia and how do they perform?
- **RQ5:** How well does Phishpedia defend against state-of-the-art adversarial attacks?
- **RQ6:** Does Phishpedia facilitate discovering of phishing pages in the wild (i.e., the Internet)?

To answer RQ1, we conduct experiments comparing the performance of Phishpedia with other baseline approaches on ∼30K phishing webpages (obtained by subscribing to Openphish Premium Service) and another ∼30K benign webpages (from Alexa's top-ranked websites). To answer RQ2, we evaluate the performance of our object detection model and Siamese model separately. For RQ3 and RQ4, we conduct a controlled experiment to evaluate the performance of Phishpedia when unseen logos are added to the target brand list, and also when we adopt alternative technical options for Phishpedia. To answer RQ5, we evaluate the model accuracy and the success rate of adversarial attacks before and after applying the gradient masking technique on the our models. For RQ6, we conduct a phishing discovery experiment to compare the performance of Phishpedia and five other solutions in reporting real-world phishing webpages in the wild (see Section 6). The experiment details are available at [7].

5.1 Datasets

To answer the above research questions, we collect relevant datasets. The details are as follows:

Phishing Webpage Dataset. To collect live phishing webpages and their target brands as ground truth, we subscribed to OpenPhish Premium Service [4] for a period of six months; this gave us 350K phishing URLs. We ran a daily crawler that, based on the OpenPhish daily feeds, not only gathered the web contents (HTML code) but also took screenshots of the webpages corresponding to the phishing URLs. This allowed us to obtain all relevant information before the URLs became obsolete. Moreover, we manually cleaned the dead webpages (i.e., those not available when we visited them) and non-phishing webpages (e.g., the webpage is not used for phishing any more and has been cleaned up, or it is a pure blank page when we accessed). In addition, we use VPN to change our IP addresses while visiting a phishing page multiple times to minimize the effect of cloaking techniques [30, 81]. We also manually verified (and sometimes corrected) the target brands for the samples. As a result, we finally collected 29,496 phishing webpages for our experimental evaluations. Note that, conventional datasets crawled from PhishTank and the free version of OpenPhish do not have phishing target brand information. Though existing works such as [36] and [80] use larger phishing datasets for phishing *detection* experiments (i.e., without identifying target brands), to the best of our knowledge, we collected the largest dataset for phishing identification experiments.

Benign Webpage Dataset. We collected 29,951 benign webpages from the top-ranked Alexa list [1] for this experiment. Similar to phishing webpage dataset, we also keep the screenshot of each URL.

Labelled Webpage Screenshot Dataset. For evaluating the object detection model independently, we use the ∼30K Alexa benign webpages collected (for the benign dataset) along with their screenshots. We outsourced the task of labelling the identity logos and user inputs on the screenshots.

We publish all the above three datasets at [7] for the research community.

5.2 Comparing Phishpedia with state-of-the-art baselines (RQ1)

5.2.1 Logo frequency in phishing webpages

We randomly sampled 5,000 webpages from the phishing webpage dataset, and manually validated that 70 of them have no logos. That is, the ratio of phishing webpages with logos is about 98.6%. Figure 9 shows the screenshot of a webpage reported by OpenPhish as a phishing webpage for Adobe. However, without a logo, we argue that the phishing attack is unlikely to be successful, as a user may not even know which credential to provide at such a page. In other words, to be effective, logo is an important feature for a phishing webpage.

5.2.2 Baselines for evaluations

We select EMD [21], PhishZoo [11], and LogoSENSE [13] as the baseline phishing identification approaches. Table 1 shows the details of baseline approaches. They are representatives for different visual similarity based identification approaches, i.e., screenshot similarity (EMD), SIFT-based similarity (PhishZoo), and HOG vector based similarity (LogoSENSE). The target brand list is the same for PhishZoo and Phishpedia, which consists of 181 brands.

Since EMD is basically a measurement technique for estimating the similarity of two screenshots, it can perform differently (both in terms of identification and runtime overhead) based on the number of referenced screenshots. The larger the number of referenced screenshots, the higher the recall that can be achieved, but at a larger runtime cost. Therefore, we define two versions of EMD for evaluations:

- $\text{EMD}_{\text{normal}}$: In this version, we equip EMD with 181 representative screenshots (collected online) as its reference, and evaluate its performance against the entire phishing and benign webpage datasets (see Section 5.1).

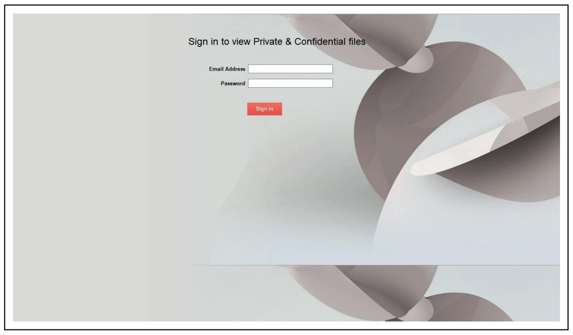


Figure 9: An Adobe phishing webpage without logo, reported by OpenPhish.

- $\text{EMD}_{\text{more\_ref}}$: We performed digest matching across the phishing webpage dataset and found that the screenshots in the first temporal half can match 48% of the screenshots in the second temporal half. This indicates that EMD with more references can potentially achieve higher recall. Therefore, in this version, we split the phishing webpage dataset of six months, temporally, into two equal halves. For improving the runtime efficiency of $\text{EMD}_{\text{more\_ref}}$, we apply the digest matching on the ∼15k screenshots in the first temporal half; this reduces the number of referenced screenshots to ∼3k.

For LogoSENSE, we let it detect phishing webpages targeting five specific brands — Paypal, Microsoft, Chase Personal Banking, DHL Airway, and Bank of America. We selected these brands for their popularity in our empirical study (see Figure 5). The list is limited to five brands because LogoSENSE requires us to train a classifier for each brand, which means that we need to label enough phishing screenshots for each of those 181 brands and train 181 classifiers for this experiment. Given this high cost of experimentation, we instead manually labelled the phishing and benign screenshots of top-5 brands to train LogoSENSE. Yet, to have a fair comparison, we run LogoSENSE to detect and identify phishing webpages only targeting for these five brands; note that the corresponding number of phishing pages is a high count of 15,658; while we still maintain 29,951 benign webpages to evaluate its false positive rates. Since the code for these three approaches are not open sourced, we implemented them for our evaluations (refer Section 8 for further details).

Table 1: Baselines for phishing identification

Baseline	Matching Criteria	Details
EMD	screenshot similarity	Use EMD measurement to compare screenshot similarity.
Phishzoo	logo similarity	Detect and match logo in a screenshot using SIFT approach.
LogoSENSE	logo similarity	Detect and match logo in a screenshot by training a HOG vector based classifier from every target brand.

In this experiment, we let the similarity threshold of $\text{EMD}_{\text{normal}}$ to be 0.92, that of EMD $_{\text{more\_ref}}$ to be 0.96, that of

Table 2: Best performance of Phishpedia and baselines.

Tool	Identification Rate	Detection Rate		Model Prediction Time (s)
		Precision	Recall	
$\text{EMD}_{\text{normal}}$	27.7%	52.0%	76.2%	0.19
$\text{EMD}_{\text{more\_ref}}$	96.7%	89.0%	74.4%	15.6
Phishzoo	28.5%	68.9%	81.8%	18.2
LogoSENSE	37.8%	20.5%	26.9%	27.2
Phishpedia	99.2%	98.2%	87.1%	0.19

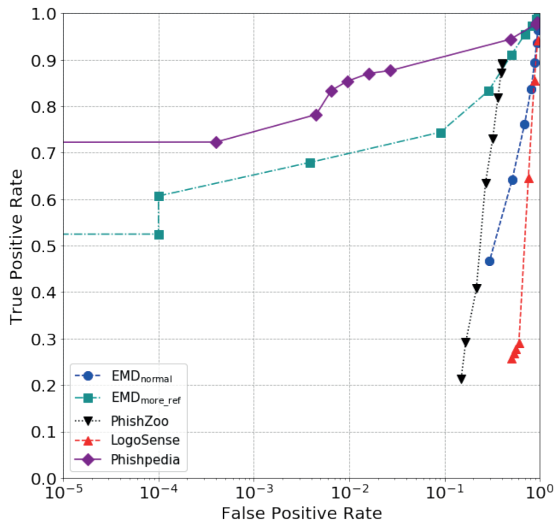


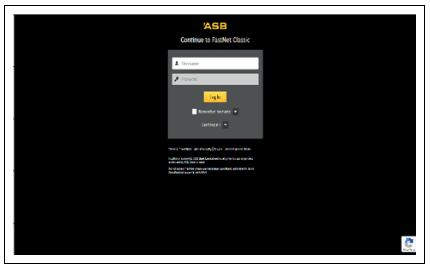
Figure 10: ROC curves (with FPR in log scale) for the four phishing identification solutions.

PhishZoo to be 0.4, and that of Phishpedia to be 0.83. These values are the optimal thresholds after we experimented multiple thresholds for each model. Readers may refer to [11], [21] for the details on their respective thresholds.

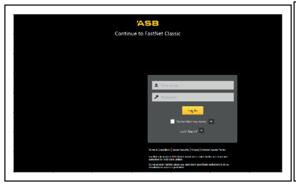
5.2.3 Results (RQ1): Phishing identification accuracy

In Table 2, we compare Phishpedia and the baseline approaches on their phishing identification rate (Identification Rate), the support for phishing detection (Detection Rate), and the runtime overhead. We calculate each column as follows. Let the number of total phishing webpages be Num^p, the number of reported phishing webpages be Rep^p, the number of reported true phishing webpages be Rep^p_{TP}, and the number of reported true phishing webpages with brand reported correctly be Id^p. The column 'Identification rate' is calculated as $\frac{\text{Id}^p}{\text{Rep}^p_{\text{TP}}}$. Precision is computed as $\frac{\text{Rep}^p_{\text{TP}}}{\text{Rep}^p}$, and Recall as $\frac{\text{Rep}^p_{\text{TP}}}{\text{Num}^p}$.

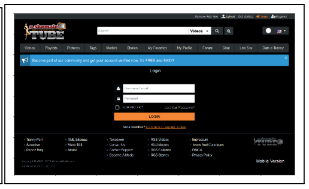
Table 2 presents the best results of the approaches (balancing between identification rate, precision, and recall). Note, all the approaches take as input a URL, thus they all share the same process and cost of transforming a URL to its screenshot, which takes approximately 1.88s on average. Also, the techniques to optimize network communications and capture screenshots are out of the scope of this work. Observe that Phishpedia outperforms the baseline approaches in identification rate, detection rate, and runtime overhead. $\text{EMD}_{\text{normal}}$ has a similar runtime efficiency as Phishpedia, but it has worse identification and detection accuracy. In contrast, $\text{EMD}_{\text{more\_ref}}$ achieves a much better performance in terms of precision and recall, but at a much higher and impractical runtime — on average, it takes 15.6 seconds to process a given webpage. PhishZoo also takes high computational time to decide on a webpage, while LogoSENSE has low detection and identification rates. Furthermore, we plot the ROC (Receiver Operating Characteristic) curves for all the identification approaches in Figure 10. As the FPR decreases, we observe a widening gap between Phishpedia and the baseline approaches except for $\text{EMD}_{\text{more\_ref}}$. Besides Phishpedia, $\text{EMD}_{\text{more\_ref}}$ is the only other approach to achieve meaningful recall (TPR) at lower FPRs (albeit this comes with a high computational cost). Yet, if we consider low FPR values of $10^{-2}, 10^{-3}$ and 10^{-4}, which are required for operational deployment, we observe that Phishpedia still achieves higher recall than $\text{EMD}_{\text{more\_ref}}$.



(a) Home page



(b) Missed phishing page (similarity of 0.921). Due to change of layout, EMD does not report this as phishing.



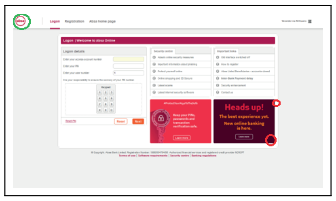
(c) False phishing (similarity of 0.947). It is caused by over abstracting the pixel colors (see Section 3 in [21]).

Figure 11: Qualitative analysis of EMD: EMD matches webpage screenshot based on most frequent color pixels and their positions, causing false positives and false negatives.

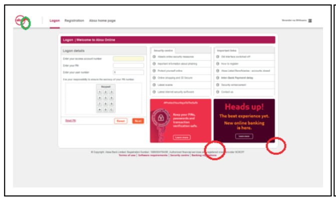
Qualitative analysis of baselines. EMD suffers from extracting coarse features (e.g., pixel colors) from webpage screenshots. Figure 11 shows a phishing page EMD missed to report (false negative) and one that it mistakenly reported (false positive).

PhishZoo is disadvantaged due to the technical limitations of SIFT. SIFT matches logo by extracting, say, k feature points from the logo and k' feature points from a screenshot. As long as k' out of k feature points are matched, such that $k' \leq k$ and $\frac{k'}{k}$ is larger than a threshold, SIFT reports that the logo appears on the screenshot. We observe that its limitations largely lies in extracting incomplete feature points and the mismatches

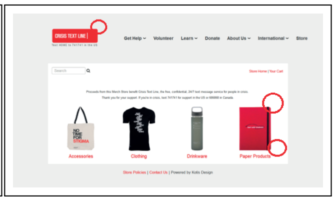
(a) Logo



(b) Recognized phishing



(c) Missed phishing page (similarity score 0.27)



(d) False phishing (similarity score 0.48)

Figure 12: Qualitative analysis of PhishZoo (threshold 0.3). Compared to the correctly matched logo regions (see green circle), SIFT matches the ABSA logo to many irrelevant regions (see red circles).

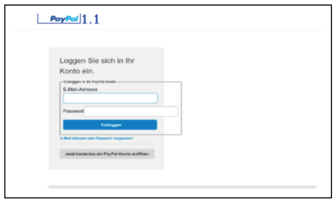
the extracted feature points have, as shown in Figure 12.

LogoSENSE incurs both high false positives and false negatives. LogoSENSE uses a sliding window of logo size through the screenshot. The content of the sliding window will be transformed into a HOG vector, to be fed to a set of trained SVM models, each of which represents a brand logo. The output is the brand logo that has highest similarity with this HOG vector. In our experiments, we use the sliding window through the screenshot with three different scales as in [13]. We observe that the fixed sliding window usually covers a partial logo (see Figure 13c), which challenges the corresponding SVM model to predict well. Besides, LogoSENSE is hard to be generalized to more complicated (or unseen) screenshots, and therefore often reports a button as logo (as showed in Figure 13d). We also observe that a large number of sliding windows on a screenshot incurs much runtime overhead.

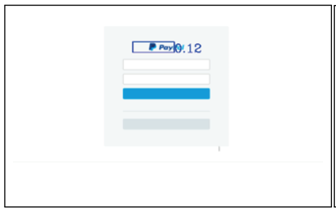
Qualitative analysis of Phishpedia. Phishpedia, with precise identity logo recognition and logo image comparison, can overcome the challenges faced in phishing identification. Yet, in this section, we investigate specific important cases of false predictions made by Phishpedia.

False positive. Phishpedia makes false positive predictions when a benign webpage has a logo looking like a well-known legitimate brand logo. As shown in Figure 14, the logo of the benign website looks similar to a logo variant the brand Navy Federal (see Figure 14b). Such a pair of similar logo confuses the Siamese model in Phishpedia. A remedy can be that we force stronger restriction on image similarity through aspect ratio and more detailed layout. We plan to explore this problem in our future work.

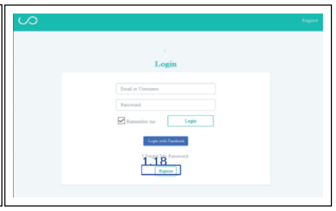
(a) Logo



(b) Recognized phishing

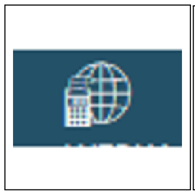


(c) Missed phishing

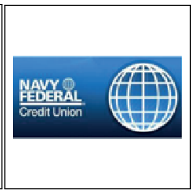


(d) False phishing

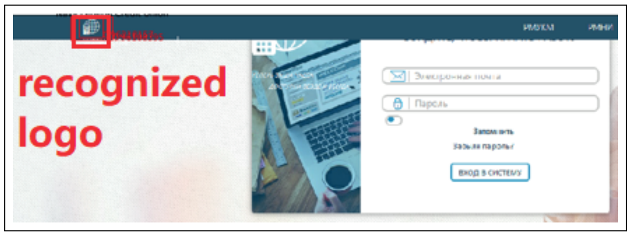
Figure 13: Qualitative analysis of LogoSENSE.



(a) Detected logo



(b) Matched logo



(c) Screenshot of benign website `https://webkassa.kz`

Figure 14: False phishing page reported by Phishpedia

False negatives. Unsurprisingly, Phishpedia misses the phishing webpages targeting a brand beyond the protected target brand list. This is a common problem for all phishing identification approaches. In practice, we can mitigate this issue by enhancing the target list. Section 5.4 shows the performance of Phishpedia when the logos of new brands are added to the target brand list at runtime.

5.3 Analyses of individual components (RQ2)

In this section, we conduct step-by-step experiments to evaluate the core components of Phishpedia independently.

5.3.1 Evaluating logo detection

We use ∼29K samples in labelled screenshot dataset for training the model and around 1,600 for testing. We compute Average Precision (AP) for each class (i.e., logo and input

Table 3: Object Detection Accuracy (Average Precision)

Object Class	Logo	Input Boxes	Overall (mAP)
Training AP	52.7	73.5	63.1
Testing AP	49.3	70.0	59.7

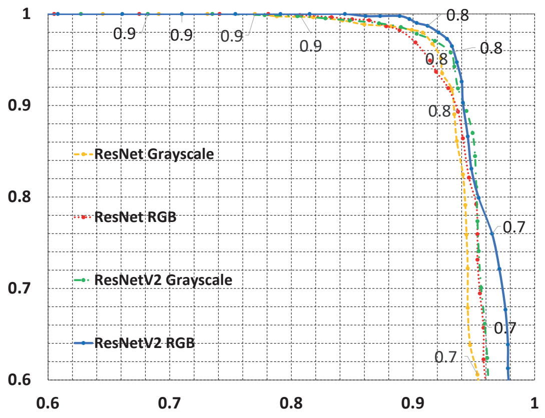


Figure 15: Accuracy of Siamese model (Precision-Recall Curve). The x-axis is recall and the y-axis is precision.

box) for IoU[1] threshold ranging from 0.5 to 0.95 with intervals of 0.05. Table 3 presents the results. In comparison to the Faster RCNN proposal [58], which achieved mAP of 67.9 on PASCAL VOC dataset [73], the mAP we achieve (i.e., 63.1 for training and 59.7 for testing) in this experiment for predicting logo and input box indicates acceptable performance.

5.3.2 Evaluating logo recognition

For evaluating the Siamese model independently, we *manually* labelled the identity logo for 1,000 phishing webpage screenshots over 181 brands and sampled 1,000 benign webpage screenshots with labelled identity logos. Then, we give 2,000 identity logos (each from a screenshot) as input to our trained Siamese model, to evaluate how well our Siamese model can compare logos. Note that, these logos from the screenshots are samples not in the training dataset.

We also experiment one alternative backbone network and one alternative input (in terms of logo color). We experiment Resnet50 [28] and RestnetV2-50 [29] as backbone networks, and consider two forms of logo input — one in RGB and another in grey-scale. By changing the similarity threshold of Siamese model from 0.5 to 0.9 with 0.05 as interval, we plot the precision-recall curve for each configuration in Figure 15. In general, the performances of four configurations are comparable and acceptable. Moreover, the Resnetv2 with RGB logo (blue plot) achieves the best performance. With the above results, we conclude that both Faster-RCNN and Siamese model achieve good performance to recognize and compare logos.

[1]IoU stands for intersection of union, and is used to evaluate the overlap of the reported bounding box with the ground truth box. The concept of IoU, average precision (AP), and mean average precision (mAP) are established terminologies in object detection algorithms (see [83] for more details).

5.4 Phishpedia generalization (RQ3)

In this experiment, we evaluate whether our Siamese model is generalizable when new logos (not used in training) are added to target brand list. To this end, we train the model on the second stage of transfer learning (see our discussion in Section 3.2) with only 130 brand logos in the target brand list and check whether it can effectively match the remaining 51 brand logos (on which the model is not trained, but forms the new target list). We randomly sample logos of 51 brands, which cover 7,411 phishing webpages in our labelled dataset.

Among the 7,411 webpages covered by 51 "new" brands, Phishpedia recognized 87.46% phishing webpages with high identification rate of 99.91%. It indicates that the Siamese model well captures generalizable features extracted from logo samples. Thus, our approach is *generalizable* for adding new logos in the target brand list during runtime.

5.5 Alternative options (RQ4)

Next, we evaluate other technical options to implement Phishpedia. We investigate the following technical options:

- **Op1:** How does other well-known object detection algorithm (one-stage model, e.g., Yolov3 [57]) perform logo recognition?
- **Op2:** How does a Siamese model trained with one-stage transfer learning and two-stage transfer learning perform logo comparison (see our discussion in Section 3.2)?
- **Op3:** How does a Siamese model trained with conventional procedure (e.g., Triplet loss function) perform logo comparison?
- **Op4:** Can we replace the Siamese model with a simpler approach such as perceptual hashing (PH) [14]?

5.5.1 Setup

For Op1, we select Yolov3, a popular one-stage object detection model. We adopt a Yolov3 model implemented with Tensorflow 1.4 framework. We train the model on the same cluster where our Faster-RCNN model is trained (see Section 4). For Op2, we compare the Siamese model trained with one-stage training with the model trained with two-stage training. For Op3, we train Siamese model in a conventional way using Triplet loss. In this experiment, we use Triplet loss function [65] to train the model. For Op4, we replace the Siamese model with a standard perceptual hashing algorithm implemented in [8].

5.5.2 Results

Table 4 shows our experimental results on the different technical options. Overall, we observe that Yolov3 has a good

Table 4: Evaluation of alternative technical options

Option	Technical Option		Identi-fication Rate	Detection Rate		Model Prediction Time (s)
	Identity Logo Recognition	Brand Recognition		Precision	Recall	
Base	Faster -RCNN	two-stage training	99.68%	99.13%	88.67%	0.19
Op1	Yolov3	two-stage training	96.59%	96.33%	63.92%	0.20
Op2	Faster -RCNN	one-stage training	99.63%	99.29%	81.07%	0.19
Op3	Faster -RCNN	non-transfer learning	90.89%	88.61%	78.57%	0.19
Op4	Faster -RCNN	perceptual hashing	24.58%	59.03%	79.37%	0.10

identification accuracy although it misses a lot of phishing webpages. Moreover, we see that two-stage training for the Siamese model improves the recall in comparison to one-stage training, and training Siamese model in a conventional way has adverse effect on the overall performance. Finally, we observe that perceptual hashing algorithm is not as competent as the Siamese model since it is less flexible to minor changes in logos. Thus, we conclude that Phishpedia employs a sound solution in terms of logo recognition and logo comparison.

5.6 Adversarial defense (RQ5)

5.6.1 Experiment on Gradient-based Technique

In this set of experiments, we apply state-of-the-art adversarial attacks on both the object detection model and the Siamese model, with two specific goals: (i) to analyze the efficacy of Phishpedia in defending against adversarial attacks, and (ii) to evaluate the effect of adversarial defense technique on the performance (in terms of accuracy) of Phishpedia.

We use DAG adversarial attack [78] to evaluate the robustness of our object detection model. We apply DAG on a test set of around 1,600 screenshots (as in Section 5.3.1). We select four adversarial attacks to evaluate the robustness of our Siamese model: DeepFool [48], i-FGSM [25], i-StepLL [34], and JSMA [24]. We apply these adversarial attacks on logos labelled in 1,000 screenshots as in Section 5.3.2, to see whether the Siamese model can still accurately match them against the logos in a target brand list. For each adversarial attack, we set the attack iteration limit as 100. Moreover, we take the attack learning rate ε of 0.5 for DAG attack, and 0.05 for i-FGSM, and i-StepLL attack (note, DeepFool and JSMA use no learning rate).

Table 5 reports the effect of the adversarial attacks on the object detection model; the prediction accuracy of both the original model and the transformed model (i.e., after the application of the defense technique described in Section 3.3) are shown. Similarly, Table 6 reports results of adversarial attacks on the Siamese model. As for the logo match accuracy in Table 6, we have N (=1,000) logos fed into the Siamese model. If k logos are matched to a logo variant of its correct brand in the target list, the logo match accuracy is computed as $\frac{k}{N}$. We observe that (i) our defense technique effectively defends against existing state-of-the-art adversarial attacks;

Table 5: Defense effect and model accuracy for adversarial attacks on the object detection model

Defense	Accuracy (mAP) without Attack	Accuracy (mAP) after Applying Adversarial Attack (DAG)
Original	59.6	12.9 (-46.7)
Transformed	58.9	58.7 (-0.02)

Table 6: Defense effect and model accuracy for adversarial attacks on the Siamese model

Defense	Logo Match Accuracy without Attack	Logo Match Accuracy After Applying Adversarial Attacks			
		i-FGSM	i-StepLL	JSMA	DeepFool
Original	93.5%	0.0%	0.1%	80.9%	0.1%
Transformed	93.5%	93.5%	93.5%	93.5%	93.5%

and (ii) the accuracy of Phishpedia is well preserved and not affected by the defense technique.

5.6.2 Experiment with Gradient-recovering Technique

While our gradient-masking based approach is effective to popular gradient-based attacks, some adversarial attacks are designed to recover the gradients to facilitate the attack. In this experiment, we adopt a state-of-the-art gradient-recovering technique, BPDA (Backward Pass Differentiable Approximation) [12], to attack Phishpedia. BDPA assumes that the gradient-masked layers in the neural network are known; it then recovers the gradient by its gradient estimation technique.

Assuming the gradient-masking layers in our model are known by an attacker, we carry out attacks on Phishpedia's Siamese model with different numbers of masked layers under the default settings of BPDA. The results are presented in Table 7, where we compare the model accuracy before and after the attacks. BPDA is seen to be effective for a small number of masked layers, but less so for a large number of masked layers. With increasing estimated layers, BPDA introduces more bias in the gradients it recovers. As a result, the adversarial attack is conducted in a biased direction when the number of masked layers increases.

6 Phishing discovery in the wild (RQ6)

We also design a phishing discovery experiment to compare Phishpedia with five phishing detection/identification solutions in literature, on their effectiveness in detecting new phishing pages in the wild (i.e., the Internet).

6.1 CertStream Service

We use CertStream service [2] which contains new domains registered from Certificate Transparency Log Network. Certificate Transparency is usually used to openly audit and monitor the event where a new domain is issued a TLS/SSL certificate. In this experimental study, we use this service to retrieve emerging new domains.

Table 7: The performance of BPDA on our Siamese model with different number of masked layers

#Masked Layers	Accuracy before	Accuracy after attack
3	93.5%	64.6%
7	93.5%	90.5%
13	93.6%	92.3%
17 (all)	93.6%	92.6%

6.2 Phishing discovery experiment

By integrating the reported emerging new domains and a phishing detector or identifier, we construct a phishing locator. We apply Phishpedia to scan and identify phishing webpages from the reported emerging domains every day. In this experiment, (as detailed in the next section) we select five known approaches in the literature to evaluate how many real-world phishing pages can they report and how precise their reported phishing pages are. We ran all the solutions for 30 days (from Sep, 10 to Oct 9, 2020). During the experiments, we record the landing URL and screenshot of each URL for postmortem analyses. For each solution, we use the configuration corresponding to the best results in Section 5.2.3; this results in each solution reporting a different number of phishing pages. Among the reported phishing URLs, we picked top reported phishing webpages (that is, the ones predicted with highest probability) for manually investigating the ground truth. The number of samples picked for each solution we evaluated is given in Table 9. Each reported phishing webpage is evaluated by two examiners independently. For those phishing webpages upon which they did not agree, we let them discuss and come to a consensus. Then, we use VirusTotal [9] to check whether it reports the same results. VirusTotal is equipped with more than 70 engines for malicious webpage detection (e.g., Google Safebrowsing). If a real phishing webpage is reported by a specific solution (i.e., one of the five baselines or Phishpedia), but none of the VirusTotal engines report it suspicious on the same day, we consider that the solution discovered a zero-day phishing webpage.

6.3 Baselines

We select the baselines covering phishing detectors and identifiers, as shown in Table 8. URLNet [36] and StackModel [80] are the two most recent techniques reported to outperform other state-of-the-art detection techniques. Besides, they work on different inputs: URLNet uses only URL string as input, where as StackModel predicts on URL and HTML content of a given page. Based on our discussion with various industry players, we are also aware that solutions similar to the above are being considered by security vendors. For the experiments here, we train both models with our dataset of phishing (from OpenPhish) and benign (from Alexa) webpages (see Section 5.1). Furthermore, we select PhishCatcher [5] as another baseline candidate, as it is an open-source version of the commercial product PhishFinder [6] searching for phishing webpages with CertStream. Similar to other phishing detectors such as URLNet and StackModel, it also assigns suspicious score based on its predefined rules. Finally, we also consider EMD and PhishZoo in this experiment as they are state-of-the-art phishing identification approaches. Note, LogoSENSE is not selected as it can support only a limited number of brands, leading to unfair comparison.

Table 8: Solutions for searching new phishing pages

Tool	Category	Input	Description
PhishCatcher	Deteciton	URL	A rule-based phishing detector to compare how similar a new domain (e.g., foceb00k.com) is with an existing legitimate domain (e.g., facebook.com).
URLNet	Deteciton	URL	A CNN-based approach that predicts on a given URL.
StackModel	Deteciton	URL+HTML	A tree-model consisting of multiple layers of random forest which takes input features extracted from URL and HTML code.
EMD	Identification	URL+Screenshot	See Section I.
Phishzoo	Identification	URL+Screenshot	See Section I.
Phishpeida	Identification	URL+Screenshot	See Section III.

Table 9: Phishing discovery results

Tool	Category	#Reported Phishing	#Top Ranked Samples	#Real Phishing	#Zero-day Phishing
PhishCatcher	Deteciton	1,421,323	1000	5	4
URLNet	Deteciton	422,093	1000	13	3
StackModel	Deteciton	327,894	1000	9	6
EMD	Identification	299,082	1000	3	2
Phishzoo	Identification	9,127	1000	8	5
Phishpeida	Identification	1,820	1000	939	623

6.4 Results

Table 9 summarizes the results on discovered phishing webpages. All discovered phishing webpages and their reports are published at [7]. We observe that, compared to other baseline approaches, Phishpedia reports far more accurate phishing results. Indeed, among all the reported 1,820 phishing webpages by Phishpedia, the total number of real phishing webpages is 1,704. Of these identified by Phishpedia, 1,133 are new real phishing webpages that are considered as benign by VirusTotal. These discovered phishing webpages range over 88 brands. Figure 16 shows the top 20 brands phishing webpages. Following the suggested practice of using VirusTotal [52], we conducted a postmortem analysis on all discovered real phishing webpages after one week, finding that 74.6% of them are still not reported by VirusTotal.

6.4.1 Why does Phishpedia outperform the baselines?

Based on the experiment results, we also have two observations for Phishpedia's advantage over the baseline approaches: **Observation 1: Plausible URL/domain is not a strong indicator for phishing.** PhishCatcher reports highest num-

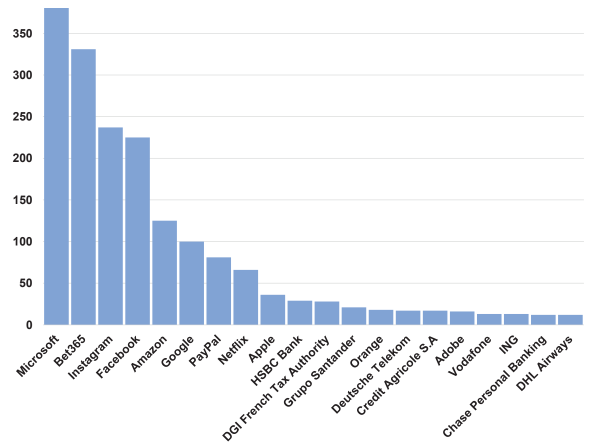
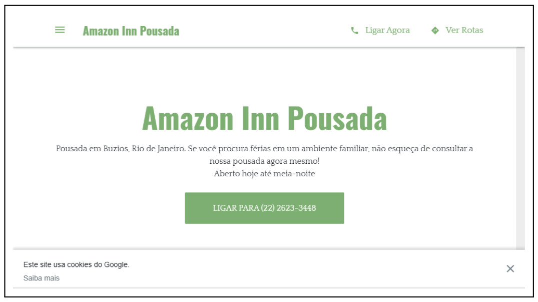


Figure 16: Top 20 brands from the found phishing webpages



Amazon Inn Pousada

Ligar Agora

Ver Rotas

Amazon Inn Pousada

Pousada em Buzios, Rio de Janeiro. Se você procura férias em um ambiente familiar, não esqueça de consultar a nossa pousada agora mesmo!

Aberto hoje até meia-noite

LIGAR PARA (22) 2623-3448

Este site usa cookies do Google.

Saiba mais

Figure 17: A benign website with suspicious name.

ber of pages as phishing, but it has very low accuracy. We note that, PhishCatcher reports high suspiciousness score for domains containing plausible brand name, such as "https://www.amazon-voucher.com/" and "http://amazoninnpousada.com/". Figure 17 shows an example of the latter. Several works in literature [31, 67, 80] make an assumption that a domain address looking similar to that of a legitimate website is more prone to be phishing. However, our phishing discovery experiment does not support this assumption, and we find less correlation between name plausibility and phishing suspiciousness. While such a conclusion is counter-intuitive, it is statistically sound given that PhishCatcher reports very few real phishing webpages.

Observation 2: Overfitting or the learned bias is a fatal drawback of machine learning approaches. We find that machine-learning based approaches do not perform well in such a real discovery study, even though they tend to show very accurate results on experimental datasets [36, 80]. StackModel [80] is a tree-based model, which allows us to generate the feature importance to explain why the model considers a webpage as phishing. Given a benign webpage, say, "https://www.httpspro-afld-amazon2020.cu.ma", we find that the StackModel reports it as phishing because it has small HTML length and low domain occurrence i.e., the frequency of domain name appearances in the HTML text. We observe that, in the OpenPhish dataset, those two features (i.e., HTML code length, and domain occurrence) are strong indicators for phishing. Nevertheless, there is no causality between these two features and the phishing intention. However, the bias learned by the model causes a large number of false positives in the phishing discovery experiment. Overall, machine learning models usually learn more of association than causality from the dataset, which is risky for their application on real-world scenario.

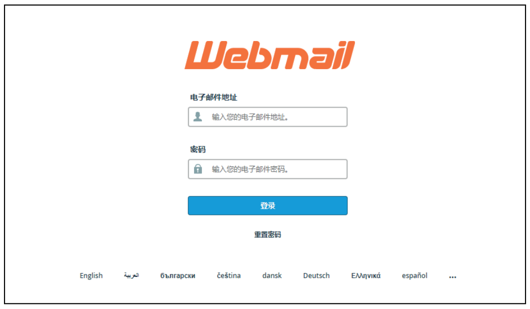


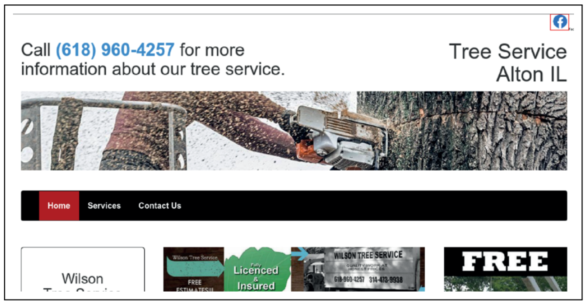
Figure 18: A website constructed through Webmail system (http://webmail.eventgiftshop.com/).

6.4.2 Investigating False Positives

Next, we investigate the false positives reported by Phishpedia during this discovery experiment; these are due to two reasons: (i) template-based websites and (ii) benign websites with a logo of some of the biggest and very popular companies such as Google, Facebook, or LinkedIn.

Template-based websites. We find that most false positives are due to some websites built with templates provided by web hosting services (e.g., https://www.cpanel.net/). After setup, the website usually has a secondary web domain such as "webmail.eventgiftshop.com". However, the web administrator preserves the default logo as shown in Figure 18. Phishpedia reports it as phishing in this experiment. Arguably, given such a webpage design, even a human user would find it difficult to decide whether it is a phishing page. As a quick remedy, we could set up a white-list to suppress the warning of Phishpedia to report webmail-based webpages. However, such websites may be considered as having a bad UI design from a security point of view, for provide phishers with a chance to construct indistinguishable phishing webpages.

Benign websites with logos of big company. We also observe that Phishpedia sometimes mistakes a benign website having a logo of a large well-known company such as Google, Facebook, LinkedIn, etc. We refer to them as *plausible websites* for Phishpedia. Such logos appear for the purposes of advertisement or Single Sign-On (SSO) used for convenient registration. Figure 19 and Figure 20 present two examples. Given that a plausible website renders a big-company logo on its screenshot, Phishpedia might interpret the screenshot as a page of that big company and report it as a phishing webpage.


Call (618) 960-4257 for more information about our tree service.

Tree Service Alton IL

Figure 19: A benign website mis-reported by Phishpedia. The screenshot has only one logo - that of Facebook.

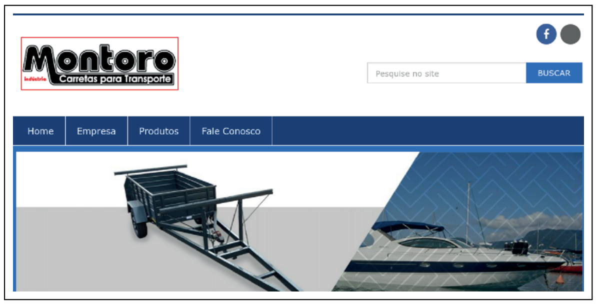


Figure 20: A benign website correctly reported by Phishpedia, which aims to report identity logo instead of arbitrary logos.

In order to further evaluate how Phishpedia perform on these plausible webpages, we additionally collected 131,975 URLs from CertStream, and experimented Phishpedia on the webpages with logos of Google, Facebook, and LinkedIn. As a result, we found 47 (0.036%) such webpages, and our manual validation confirms that four of them are real phishing webpages. Among the 47 webpages, Phishpedia reports 7 of them as phishing; the precision is $\frac{4}{7}$ and the recall is $\frac{4}{4}$.

Intuitively, Phishpedia is robust to such websites because it recognizes *identity logo* instead of arbitrary logos. When Faster-RCNN model reports multiple logos, Phishpedia uses the logo with highest confidence (see Section 3.1). Nevertheless, such webpages may still cause false positives. We will address them in our future work.

6.4.3 Investigating False Negatives

We also investigate the false negatives of Phishpedia in the phishing discovery experiment. Note that the metric recall is hard to obtain as the ground truth can only be validated manually, which is laborious for large-scale evaluations. In this experiment, we sample 1,500 CertStream URLs. Our manual evaluation found no phishing URLs. Therefore, we further used PhishCatcher to select 1,500 CertStream URLs and we confirm 16 real phishing webpages among them.

Taking the phishing label of the above 1,500 CertStream URLs reported by PhishCatcher as ground truth, we compare VirusTotal, EMD, PhishZoo, URLNet, StackModel, and Phishpedia for their precision and recall. The results are given in Table 10. Phishpedia achieves a good balance between the precision and the recall, in comparison to other baselines.

Table 10: Precision and recall of Phishpedia and baselines on the URLs filtered by PhishCatcher.

Solution	Precision	Recall
VirusTotal	28.00%	43.75%
EMD	1.00%	43.00%
PhishZoo	1.20%	43.75%
URLNet	1.22%	93.75%
StackModel	1.30%	100.0%
Phishpedia	87.50%	87.50%

7 Discussions

7.1 Webpage semantics

Our vision for Phishpedia is to identify the semantics of a webpage screenshot so that we can compare its rendered intention with its real domain. We achieve this by recognizing identity logos and brands via Faster-RCNN and Siamese model. While our experiments demonstrate promising results, the semantics can sometimes go beyond logo-domain inconsistency. For example, a benign webpage might have a Google icon as its content, which can causes confuse Phishpedia. In our future work, we will explore webpage layout information or extract topic model from a webpage content to infer the identity of a screenshot in a more confident way.

7.2 Application and deployment scenarios

Scenario 1: URL access interception. One of the most common channels for delivering URLs of phishing webpages is email [32]. Vendors can have multiple options for deploying e-mail security gateway. i) All URLs in an e-mail are sent to Phishpedia; and the results are used to classify the mail as phishing or to deliver to the user. ii) Every URL in an e-mail is transformed and prefixed with a cloud-service link, so that anytime a user clicks on the link, Phishpedia service in the cloud analyses the URL. In this case, Phishpedia fits in with negligible additional delay.

Scenario 2: Complementing phishing detectors. Phishpedia can also be used for providing explanations to existing phishing detectors. A typical example is an analyst at a SOC (security operations centre) going through a list of URLs that have been classified as phishing by multiple phishing detectors. Phishpedia can then be used to identify the phishing target and provide visual explanation on webpage screenshot.

Scenario 3: Threat intelligence gathering. With its high precision, Phishpedia can run as an independent service, to discover new phishing pages on the Internet. This live threat intelligence can be used to maintain dynamic black lists for users to block access to phishing pages.

8 Threats to Validity

In our experiments, an internal threat is that we re-implemented all the baseline approaches because their implementations are not publicly available. While this may result in not obtaining the best performance of these models, we emphasize that we experimented the baselines with multiple thresholds. For example, for LogoSENSE, we evaluated multiple versions and report the results of best performance. We also publish all our baseline implementations in [7] for replicating the experiments. An external threat is that, VirusTotal engines can be adversely affected due to cloaking of phishing websites. Therefore, it cannot be determined whether improved detection comes from Phishpedia, or the crawling infrastructure that Phishpedia runs on.

9 Related Works

Phishing webpage detection. Current phishing detection approaches can be classified according to their input, i.e., URL, HTML content, and visual screenshot. URL features have proved well on the datasets collected from some open phishing platform such as PhishTank and OpenPish [36, 55, 60, 62]. Rakesh *et al.* [55] explored features such as URL length, frequency of suspicious symbols and characters, etc., and they showed that their selected features have better performance on a variety of machine learning models. Guang *et al.* [36] proposed URLNet which uses character-level and token-level convolutional neural network for prediction. Researchers also explored detecting phishing based on HTML features [10, 17, 27, 31, 38, 50, 69, 80]. Ke *et al.* [31] used frequency of keywords appearing in specific HTML tags and that of brand names as features, and use three traditional classifiers to make the prediction. Other works used both URL and HTML contents to achieve a better prediction accuracy. Cantina [27] and Li *et al.* [80] enhanced traditional URL and HTML features by introducing IP addresses and top name domain. Moreover, visual analysis (e.g., OCR technique) is often used as a complementary technique to extract text in images to enhance HTML features [31, 62, 63, 82]. We refer to surveys for more details [32, 61].
Phishing target identification. Existing identification techniques detected phishing target via search engine [44, 63, 79] and by employing target brand list [11, 13, 21, 74]. Samuel *et al.*'s Know-Your-Phish work [63] is representative for search-engine based approach. They extracted dominant keywords from HTML content (including text recognized by OCR) and applied search engine (e.g., Google) to return the most likely targets. However, repetitive network connections can incur huge runtime overhead and it is also a challenge to select appropriate keyword for search engine.

Fu *et al.* [21] first proposed the idea of using a target brand list. They compared the screenshot of a suspicious webpage with that of all websites in the target brand list, subsequently reporting the phishing target if the similarity is above a threshold. As an alternative to screenshot, Medvet *et al.* [46] and Rosiello *et al.* [59] explored techniques to compare page content such as text, images, and layout. Following their work, Afroz *et al.* and Wang *et al.* considered logo as a more reliable invariant to compare, and pioneered logo-based approaches such as Phishzoo [11] and Verilogo [74], which locate logos on the screenshot based on SIFT. As discussed above, the performance of SIFT limits the accuracy of the approach.

10 Conclusion

In this work, we proposed Phishpedia to identify phishing webpage with visual explanation. Phishpedia well solves the challenging problems of logo detection and brand recognition. Our evaluation shows that Phishpedia performs better than state-of-the-art approaches on experiments using real datasets as well as the ability to discover new online phishing pages. In our future work, we will address Phishpedia's false positive issue in benign webpage with logos of big company. Moreover, we will extend Phishpedia into an online phishing monitoring system to collect active phishing kits, on which we will apply state-of-the-art program analysis techniques [40–43] to gain more insights into the phishing campaigns.

Acknowledgement

We thank the anonymous reviewers and the shepherd whose comments helped to improve this work. This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its Corporate Laboratory@University Scheme, National University of Singapore, and Singapore Telecommunications Ltd.

References

[1] Alexa Ranking. https://www.alexa.com/siteinfo, October 2020.

[2] CertStream. https://certstream.calidog.io, October 2020.

[3] Google Transparency Report: Google Safe Browsing. https://transparencyreport.google.com/safe-browsing/overview, 2020.

[4] OpenPhish. https://www.openphish.com/, October 2020.

[5] PhishCatcher. https://github.com/x0rz/phishing_catcher, October 2020.

[6] PhishFinder. https://phishfinder.io/, October 2020.

[7] Phishpedia. https://sites.google.com/view/phishpedia-site/home, October 2020.

[8] Python Implementation of Perceptual Hashing. https://pypi.org/project/ImageHash/, October 2020.

[9] VirusTotal. https://www.virustotal.com/gui/home/upload, October 2020.

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

[11] Sadia Afroz and Rachel Greenstadt. Phishzoo: Detecting phishing websites by looking at them. In *IEEE Int'l Conf. on Semantic Computing*, pages 368–375, 2011.

[12] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, pages 274–283, 2018.

[13] Ahmet Selman Bozkir and Murat Aydos. LogoSENSE: A Companion HOG based Logo Detection Scheme for Phishing Web Page and E-mail Brand Recognition. *Computers & Security*, 2020.

[14] Ahto Buldas, Andres Kroonmaa, and Risto Laanoja. Keyless signatures' infrastructure: How to build global distributed hash-trees. In *Nordic Conf. on Secure IT Systems*, pages 313–320, 2013.

[15] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*, 2018.

[16] Ee Hung Chang, Kang Leng Chiew, Wei King Tiong, et al. Phishing detection via identification of website identity. In *Int'l Conf. on IT convergence and security (ICITCS)*, pages 1–4, 2013.

[17] Kang Leng Chiew, Choon Lin Tan, KokSheik Wong, Kelvin S.C. Yong, and Wei King Tiong. A new hybrid ensemble feature selection framework for machine learning-based phishing detection system. *Information Sciences*, 484:153 – 166, 2019.

[18] Dahjung Chung, Khalid Tahboub, and Edward J Delp. A two stream siamese convolutional neural network for person re-identification. In *Proc. IEEE ICCV*, pages 1983–1991, 2017.

[19] Cyren. Evasive Phishing Driven by Phishing-as-a-Service. https://www.cyren.com/blog/articles/evasive-phishing-driven-by-phishing-as-a-service.

[20] Mira Dontcheva, Steven M Drucker, David Salesin, and Michael F Cohen. Changes in webpage structure over time. *UW, CSE*, 2007.

[21] Anthony Y. Fu, Wenyin Liu, and Xiaotie Deng. Detecting phishing web pages with visual similarity assessment based on earth mover's distance (emd). *IEEE Trans. on Dependable and Secure Computing*, 3(4):301–311, Oct 2006.

[22] Sujata Garera, Niels Provos, Monica Chew, and Aviel D Rubin. A framework for detection and measurement of phishing attacks. In *Proceedings of the 2007 ACM workshop on Recurring malcode*, pages 1–8, 2007.

[23] Ross Girshick. Fast R-cnn. In *Proceedings of the IEEE ICCV*, pages 1440–1448, 2015.

[24] Ian Goodfellow, Nicolas Papernot, and Patrick D. McDaniel. cleverhans v0.1: an adversarial machine learning library. *arXiv preprint arXiv:1610.00768*, 1, 2016.

[25] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[26] Group-IB. How much is the phish? Underground market of phishing kits is booming — Group-IB. 2019.

[27] Xiang Guang, Hong Jason, P. Rose Carolyn, and Cranor Lorrie. Cantina+: A feature-rich machine learning framework for detecting phishing web sites. In *ACM Trans. on Information and Sys. Sec.*, pages 1–28, 2011.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE CVPR*, pages 770–778, 2016.

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV*, pages 630–645, 2016.

[30] Luca Invernizzi, Kurt Thomas, Alexandros Kapravelos, Oxana Comanescu, Jean-Michel Picod, and Elie Bursztein. Cloak of visibility: Detecting when machines browse a different web. In *IEEE S&P*, 2016.

[31] Tian Ke, T.K Steve, Jan, Hu Hang, Y. Danfeng, and W. Gang. Needle in a haystack: Tracking down elite phishing domains in the wild. In *Proc. IMC*, 2018.

[32] M. Khonji, Y. Iraqi, and A. Jones. Phishing Detection: A Literature Survey. *IEEE Communications Surveys Tutorials*, 15(4):2091–2121, 2013.

[33] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, 2015.

[34] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[35] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[36] Hung Le, Quang Pham, Doyen Sahoo, and Steven CH Hoi. URLNet: learning a URL representation with deep learning for malicious URL detection. *arXiv preprint arXiv:1802.03162*, 2018.

[37] Hyeungill Lee, Sungyeob Han, and J. Lee. Generative adversarial trainer: Defense to adversarial perturbations with GAN. *arXiv preprint arXiv:1705.03387*, 2017.

[38] Jehyun Lee, Pingxiao Ye, Ruofan Liu, D.M. Divakaran, and M.C. Chan. Building robust phishing detection system: an empirical analysis. In *NDSS MADWeb*, 2020.

[39] G. L'Huillier, A. Hevia, R. Weber, and S. Ríos. Latent semantic analysis and keyword extraction for phishing classification. In *IEEE Int'l Conf. on Intelligence and Security Informatics*, pages 129–131, 2010.

[40] Yun Lin, Jun Sun, Gordon Fraser, Ziheng Xiu, Ting Liu, and Jin Song Dong. Recovering fitness gradients for interprocedural boolean flags in search-based testing. In *ISSTA*, pages 440–451, 2020.

[41] Yun Lin, Jun Sun, Lyly Tran, Guangdong Bai, Haijun Wang, and Jinsong Dong. Break the dead end of dynamic slicing: Localizing data and control omission bug. In *ASE*, pages 509–519, 2018.

[42] Yun Lin, Jun Sun, Yinxing Xue, Yang Liu, and Jinsong Dong. Feedback-based debugging. In *ICSE*, pages 393–403. IEEE, 2017.

[43] Yun Lin, Zhenchang Xing, Yinxing Xue, Yang Liu, Xin Peng, Jun Sun, and Wenyun Zhao. Detecting differences across multiple instances of code clones. In *ICSE*, pages 164–174, 2014.

[44] G. Liu, B. Qiu, and L. Wenyin. Automatic detection of phishing target from phishing webpage. In *IEEE CVPR*, pages 4153–4156, Aug 2010.

[45] Justin Ma, Lawrence K Saul, Stefan Savage, and Geoffrey M Voelker. Beyond blacklists: learning to detect malicious web sites from suspicious URLs. In *Proc. ACM KDD*, pages 1245–1254, 2009.

[46] Eric Medvet, Engin Kirda, and Christopher Kruegel. Visual-similarity-based phishing detection. In *Proc. SecureComm*, 2008.

[47] Tyler Moore. Phishing and the economics of e-crime. *Infosecurity*, 4(6):34–37, 2007.

[48] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *IEEE CVPR*, pages 2574–2582, 2016.

[49] A. Oest, P. Zhang, B. Wardman, E. Nunes, J. Burgis, A. Zand, K. Thomas, A. Doupé, and G.J Ahn. Sunrise to Sunset: Analyzing the End-to-end Life Cycle and Effectiveness of Phishing Attacks at Scale. In *USENIX Security Symposium*, 2020.

[50] Alina Oprea, Zhou Li, Robin Norris, and Kevin Bowers. Made: Security analytics for enterprise threat detection. In *Proc. ACSAC*, pages 124–136, 2018.

[51] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE S&P*, pages 582–597, 2016.

[52] Peng Peng, Limin Yang, Linhai Song, and Gang Wang. Opening the blackbox of VirusTotal: Analyzing online phishing scan engines. In *Proc. IMC*, 2019.

[53] Lorien PRATT. Reuse of neural networks through transfer. *Connection Science (Print)*, 8(2), 1996.

[54] Rajat Raina, Andrew Y Ng, and Daphne Koller. Constructing informative priors using transfer learning. In *Proc. ICML*, pages 713–720, 2006.

[55] Verma Rakesh and Dyer Keith. On the character of phishing urls: Accurate and robust statistical learning classiers. In *Proc. ACM Conf. on Data and Application Security and Privacy*, pages 111–122, 2015.

[56] Waseem Rawat and Zenghui Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation*, 29(9):2352–2449, 2017.

[57] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[58] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NeuIPS*, 2015.

[59] Angelo PE Rosiello, Engin Kirda, Fabrizio Ferrandi, et al. A layout-similarity-based approach for detecting phishing pages. In *Proc. Int'l Conf. on Security and Privacy in Communications Networks and the Workshops-SecureComm*, pages 454–463, 2007.

[60] Ozgur Koray Sahingoz, Ebubekir Buber, Onder Demir, and Banu Diri. Machine learning based phishing detection from URLs. *Expert Systems with Applications*, 117:345 – 357, 2019.

[61] Doyen Sahoo, Chenghao Liu, and Steven CH Hoi. Malicious URL detection using machine learning: A survey. *arXiv preprint arXiv:1701.07179*, 2017.

[62] Marchal Samuel, François Jérôme, State Radu, and Engel Thomas. PhishStorm: Detecting phishing with streaming analytics. In *IEEE Trans. Netw. Service Manag.*, 2014.

[63] Marchal Samuel, Saari Kalle, Singh Nidhi, and Asokan N. Know your phish: Novel techniques for detecting phishing sites and their targets. In *IEEE ICDCS*, 2016.

[64] Aécio SR Santos, Nivio Ziviani, Jussara Almeida, Cristiano R Carvalho, Edleno Silva de Moura, and Altigran Soares da Silva. Learning to schedule webpage updates using genetic programming. In *Int'l Symposium on String Proc. and Info. Retrieval*, 2013.

[65] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE CVPR*, 2015.

[66] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *NeuIPS*, pages 3358–3369, 2019.

[67] Hossein Shirazi, Bruhadeshwar Bezawada, and Indrakshi Ray. "kn0w thy doma1n name" unbiased phishing detection using domain name based features. In *Proc. ACM Symposium on Access Control Models and Technologies*, pages 69–75, 2018.

[68] Hossein Shirazi, Bruhadeshwar Bezawada, Indrakshi Ray, and Charles Anderson. Adversarial sampling attacks against phishing detection. In Simon N. Foley, editor, *Data and Applications Security and Privacy XXXIII*, pages 83–101, 2019.

[69] Alrwais Sumayah, Yuan Kan, A. Eihal, Li Zhou, and Wang XiaoFeng. Understanding the dark side of domain parking. In *USENIX Security Symposium*, 2014.

[70] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *IEEE CVPR*, pages 1701–1708, 2014.

[71] Kurt Thomas, Frank Li, Ali Zand, Jacob Barrett, Juri Ranieri, Luca Invernizzi, Yarik Markov, Oxana Comanescu, Vijay Eranti, Angelika Moscicki, et al. Data breaches, phishing, or malware? understanding the risks of stolen credentials. In *ACM CCS*, 2017.

[72] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

[73] Sara Vicente, Joao Carreira, Lourdes Agapito, and Jorge Batista. Reconstructing pascal voc. In *IEEE CVPR*, pages 41–48, 2014.

[74] Ge Wang, He Liu, Sebastian Becerra, Kai Wang, Serge J Belongie, Hovav Shacham, and Stefan Savage. *Verilogo: Proactive phishing detection via logo recognition*. Department of Computer Science and Engineering, University of California, 2011.

[75] Jing Wang, Weiqing Min, Sujuan Hou, Shengnan Ma, Yuanjie Zheng, Haishuai Wang, and Shuqiang Jiang. Logo-2K+: A Large-Scale Logo Dataset for Scalable Logo Classification. In *AAAI*, pages 6194–6201, 2020.

[76] Liu Wenyin, Ning Fang, Xiaojun Quan, Bite Qiu, and Gang Liu. Discovering phishing target based on semantic link network. *Future Generation Computer Systems*, 26(3):381 – 388, 2010.

[77] Yuxin Wu, Alexander Kirillov, F. Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[78] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. Adversarial examples for semantic segmentation and object detection. In *Proceedings of the IEEE ICCV*, pages 1369–1378, 2017.

[79] H. Yuan, X. Chen, Y. Li, Z. Yang, and W. Liu. Detecting Phishing Websites and Targets Based on URLs and Webpage Links. In *Proc. ICPR*, pages 3669–3674, 2018.

[80] Li Yukun, Yang Zhenguo, Chen Xu, Yuan Huaping, and Liu Wenyin. A stacking model using url and html features for phishing webpage detection. In *Future Generation Computer Systems*, pages 27–39, 2019.

[81] Penghui Zhang, Adam Oest, Haehyun Cho, RC Johnson, Brad Wardman, Shaown Sarker, Alexandros Kpravelos, Tiffany Bao, Ruoyu Wang, Yan Shoshitaishvili, Adam Doupé, and Gail-Joon Ahn. CrawlPhish: Large-scale Analysis of Client-side Cloaking Techniques in Phishing. In *IEEE S&P*, 2021.

[82] Wei Zhang, Qingshan Jiang, Lifei Chen, and Chengming Li. Two-stage elm for phishing web pages detection using hybrid features. *WWW*, 20(4):797–813, Jul 2017.

[83] Zhengxia Zou, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *arXiv preprint arXiv:1905.05055*, 2019.