Evaluating Robustness of Reference-based Phishing Detectors

Eunjin Roh[*]
Oregon State University
Corvallis, OR, USA
rohe@oregonstate.edu

Sungwoo Jeon[*]
KAIST
Daejeon, Republic of Korea
j0070ak@kaist.ac.kr

Sooel Son
KAIST
Daejeon, Republic of Korea
sl.son@kaist.ac.kr

Sanghyun Hong
Oregon State University
Corvallis, OR, USA
sanghyun.hong@oregonstate.edu

Abstract

Recent years have seen an unprecedented increase in phishing attacks, threatening the security of Internet users. In response, studies have explored defenses that automatically identify the intentions of suspicious websites. Most defenses work by employing machine learning models that examine different components of a website, *e.g.*, logos or areas asking login credentials. However, the robustness of this emerging approach against evasive adversaries is not well understood. In this work, we evaluate the robustness of recent phishing detectors against evasion attacks. We first conduct an extensive literature review to identify common mistakes in evaluating their adversarial robustness. We then develop evasion attacks to assess the true robustness of reference-based phishing detectors. Our evaluation of state-of-the-art phishing detectors, PhishIntention and Dynaphish, shows that they are significantly less robust in adversarial settings than demonstrated in the original studies. We also analyze the factors attributing to these vulnerabilities and suggest potential research directions for developing robust detectors.

CCS Concepts

• **Security and privacy** → **Software and application security**; • **Computing methodologies** → *Machine learning*.

Keywords

Phishing detector; Evasion attack; Machine learning

ACM Reference Format:
Eunjin Roh, Sungwoo Jeon, Sooel Son, and Sanghyun Hong. 2025. Evaluating Robustness of Reference-based Phishing Detectors. In *ACM Asia Conference on Computer and Communications Security (ASIA CCS '25), August 25–29, 2025, Hanoi, Vietnam.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3708821.3710840

[*]Both authors contributed equally to this research.

1 Introduction

An emerging approach to combat unprecedented phishing attacks is to employ reference-based phishing detectors [1, 2, 8, 14, 24, 25, 36]. These detectors compare the *visual characteristics* of phishing websites to a curated list of references, such as logos from known benign brands. Once a match is found between an inspected website and the references, they verify if the website URL is the legitimate one known for the matched brand. If the verification fails, the URL is added to the browser blocklist to safeguard Internet users.

Recent detectors employ neural networks to compare and analyze visual characteristics. While neural networks offer an increase in detection performance, they are known to be susceptible to small input perturbations [15]. This vulnerability allows an adversary to craft *adversarial examples* [15, 27]: test inputs with human-imperceptible perturbations carefully-crafted to fool a target classifier. However, the robustness of recent phishing detectors to adversarial attacks has been understudied. While prior work [1, 24–26] has shown some resilience to adversarial examples, the evaluation practice has been far from the golden standard suggested by the literature on adversasrial examples and defenses [10].

In this paper, we study the adversarial robustness of recent reference-based phishing detectors, especially those employing neural networks as a key component. We ask the question: *How vulnerable are these phishing detectors to adversarial examples, and how different is the vulnerability our work will show compared to prior research?* This is a particularly important question to answer because phishing detectors are deployed in and exposed to adversarial settings, where a motivated attacker is likely to evade detection to increase the reachability of their attacks to Internet users.

Contributions. We *first* conduct an extensive literature review on reference-based phishing detectors and categorize their common mistakes in evaluating adversarial robustness into three categories: (1) Gradient masking. Recent work [24–26] proposes a function that quantizes layer activation (i.e., StepReLU) and shows its effectiveness in rendering attacks ineffective. We find that this practice has been warned against by Athalye et al. [4] as a form of security through obscurity. (2) Weak attacks. We find that several studies employ weak adversarial examples to test their resilience. This occurs due to the incorrect implementation of existing attacks or assuming an adversary without knowledge of the target system's internals. (3) No end-to-end evaluation. No prior work conducts robustness evaluations in an end-to-end manner in a practical setting. Most prior work focuses on testing individual neural networks they employ, making it difficult to assess the holistic impact of evasive adversaries on phishing detectors.

Second, we bridge the knowledge gap between the reported robustness in prior work and the actual robustness by proposing evasion attacks. We focus on the state-of-the-art detectors, PhishIntention [25] and DynaPhish [26], composed of three key components: an object detection model, the OCR-aided brand recognition model, and the CRP classifier. We present three evasion attacks designed to fool the decisions of the three key models and demonstrate an attack success rate of 84.1–95.6%, even under scenarios where their StepReLU defense has been deployed. We highlight a stark contrast to the results reported in prior work [24, 25], which achieves approximately 0% attack success in the presence of StepReLU.

Third, we evaluate the robustness of phishing detectors in end-to-end settings to evaluate the real-world implications of our attacks. To this end, we propose three evasion attacks: the No-object attack, designed to prevent any objects from being detected in a website screenshot, the Triple-jump attack, which aims to fool all the subsequent classifiers, and the Logo masking attack, which operates in a black-box manner by masking logos. In contrast to the prior work, the first two attacks evaluate how an adversary can full the entire detection pipeline while having full knowledge of only specific models or components. We also employ techniques to enhance evasion success under practical constraints imposed by additional steps a detector employs, such as converting floating-point perturbations to pixel values or altering the resolution of screenshots before they are processed by the models. Our evaluation shows a recall dropping to 0.03–0.33, demonstrating the practicality of evasion attacks in a real-world deployment setting. It also implies that combining models with different functionalities does not necessarily offer significantly improved robustness.

We lastly study two potential countermeasures against our evasion attacks: one that makes the detection pipeline robust to human-imperceptible perturbations, and the other is the designs of HTML components, such as logos, with features that make evasion more difficult to achieve. Employing cost-effective denoising techniques, such as Gaussian blurring, before a model processes a website screenshot reduces the attack success rate, while preserving the model's performance on unperturbed inputs. We also review existing defenses designed to counter adversarial examples and explore their potential applications within the phishing detection framework. Moreover, we analyze the cases where our evasion attack fails and find that logos with simplified text and fewer embellishments make it more difficult for an adversarial attack to fool a target detector. We hope our work inspires future work on (accurately) evaluating the robustness of phishing detectors and developing defenses to prevent evasive attempts.

2 Background

Here we provide an overview of the necessary background knowledge: reference-based phishing detectors and adversarial attacks.

2.1 Neural Networks

A neural network (NN) is a function $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ that maps an input $\mathbf{x} \in \mathcal{X}$ to an output $\mathbf{y} \in \mathcal{Y}$. In supervised settings, such as classification over k classes, the output becomes a probability vector $\mathbf{y} \in [0, 1]^k$, while in generative modeling, the output $\mathbf{y} \in \mathcal{R}^d$ shares the same dimension as an input. A NN consists of a sequence of layers l_i parameterized by a set of tensors θ, including weights w_i and biases b_i. Each layer l_i runs a linear transformation of an input, defined as $l_i(\mathbf{x}_i) = \sigma(w_i \cdot \mathbf{x} + b_i)$. A non-linear activation function $\sigma(\cdot)$, such as ReLU, along with additional layers, such as pooling, dropout, or batch-norm, are applied to the output. In feed-forward networks, the last fully connected layer are referred to as the *penultimate layer*, and the input to this layer is called *features*. Since many phishing detectors use NNs for classification, our work focuses on feed-forward networks for this purpose, where the parameters θ are trained to minimize the prediction error of an input. During training, we iteratively update θ by back-propagating the error between the predicted output $y_p = \arg\max_k \mathbf{y}$ and the oracle label y_t. Training is stopped when f_θ converges to an acceptable error rate or after a sufficient number of iterations, at which point we save f along with its parameters θ. During testing (or inference), we load f and its parameters θ to compute the prediction for a given test input, which is typically not part of the training data.

2.2 Reference-based Phishing Detectors

Phishing detectors are designed to identify whether a website is not a legitimate website. Once a website is identified as phishing, the detector reports its URL to third parties. These URLs are then added to blocklists, such as Google Safe Browsing [6, 16], to prevent Internet users from browsing these websites.

Reference-based phishing detectors. These detectors work by comparing domain-brand consistencies. They utilize a predefined *reference* list of domains and brand representations, such as logos. Given a website, the detector first extracts the logos from its screenshots and then compares them with the references to identify any matches. Modern phishing detectors employ machine-learning models to extract the brand representations from a screenshot and compare them with the references [1, 2, 8, 14, 24, 25, 36].

Figure 1 illustrates the workflow of the most recent system for reference-based phishing detection, DynaPhish, proposed by Liu et al. [26]. DynaPhish employs the phishing detectors presented by the prior work: Phishpedia [24] and PhishIntention [25], and on top of them, proposes additional mechanisms for keeping the reference list up-to-date. The backbone of its detection pipeline is composed of three neural network classifiers: an object detector, an OCR-Siamese model, and a credential-requiring page (CRP) classifier. The object detector uses the Faster R-CNN architecture [32] and identifies all components, such as logos or forms for user inputs, from a website's screenshot. The OCR-Siamese model identifies the brand by determining whether the logos detected by the object detector match those in the reference list. Is uses the OCR model $\mathcal{A}(\cdot)$ and the Siamese model $\mathcal{R}(\cdot)$, which is based on the ASTER encoder architecture [33] and the ResNetV2-50 architecture [18, 21], respectively. Unlike typical Siamese models that identifies how *similar* two different inputs are, this model compares the similarity of the output embedding e of the logo x with the predefined embeddings in the reference list. The OCR-Siamese model computes e as follows:

$$e = \mathcal{R}(x) \oplus \mathcal{A}(x),$$

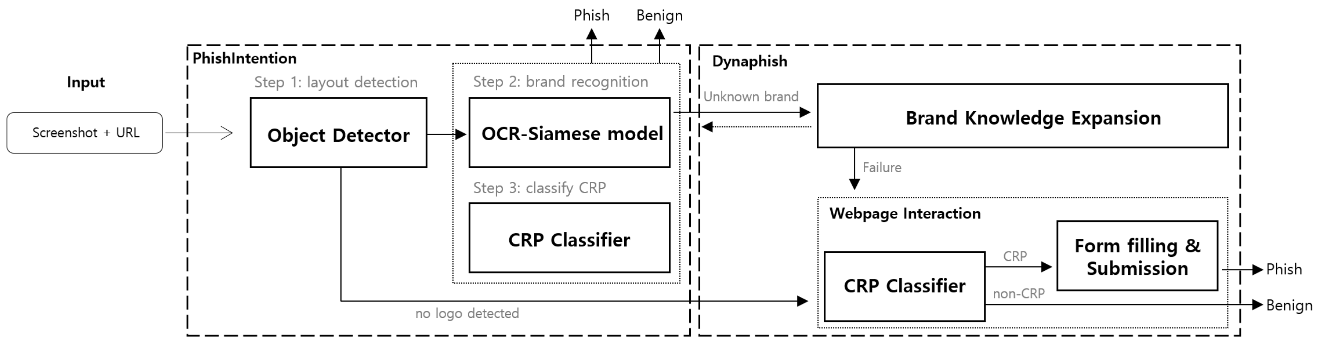


Figure 1: Illustration of how a state-of-the-art reference-based phishing detector inspects a suspicious website.

where x is the logo, $\mathcal{R}$ is the ResNetV2-50 model, and $\mathcal{A}$ is the Aster encoder model. The model concatenates the outputs of $\mathcal{R}$ and $\mathcal{A}$ for the logo x. The model then uses cosine similarity to compare the embeddings. If there is a match, the system compares the domain of the website with the known domain of the logo brand. If the domains differ, the system reports the website as phishing. Otherwise, the system continues its inspection using the CRP classifier. It checks whether the website asks for a user's credentials. If the website does ask, the system performs a further investigation to identify its intention. The CRP classifier, based on the same ResNetV2-50 architecture, takes both the screenshot and the detected objects from the object detector. Please refer to Appendix A.1 for more details about the OCR-Siamese model.

DynaPhish introduces two additional mechanisms to keep the reference list up-to-date: brand knowledge expansion and web interaction. If the OCR-Siamese model cannot find a match from the reference list, the brand knowledge expansion module performs a Google search with the detected logo and stores the logos from a few website screenshots appearing in the Google search. Upon the addition, the system runs its detector, PhishIntention, again with the updated reference list. The web interaction module checks whether a website accepts any random credentials. It uses the components identified by the CRP classifier, such as forms for user ID and password, and the log-in button. In the case of a website showing only the log-in button, it clicks through the button a few times until the website displays the log-in forms. The module automatically fills the forms with random credentials, and if the log-in is successful, DynaPhish reports the website as phishing.

Our work studies the adversarial robustness of these representative, state-of-the-art reference-based phishing detectors.

2.3 Adversarial Attacks on Neural Networks

Neural networks are vulnerable to small perturbations to their inputs [35]. This vulnerability enables *adversarial attacks*: an adversary can craft human-imperceptible perturbations and add them to a target network's input to alter its classification result for evasion [10, 11, 15, 20, 27]. The process of crafting such *adversarial examples* $x + \delta$ can be formulated as follows:

$$\arg\max_{|\delta| \le \varepsilon} \mathcal{L}(\theta, x + \delta, y),$$

where x is the test input, θ is the model parameters of a target model f, y is the oracle label, $\mathcal{L}$ is the loss function, and ϵ is the maximum perturbation bound to keep the human-imperceptibility. We consider the PGD and DAG attacks because PGD is the strongest known adversarial attack, while DAG is used in the prior studies [25, 26] to evaluate adversarial robustness of their detectors.

PGD. Projected gradient descent (PGD) [27] is a representative algorithm for crafting adversarial examples, described as follows:

$$x^{(t+1)} = \Pi_\epsilon\Big(x^t + \alpha \cdot \texttt{sign}\Big(\nabla_x \mathcal{L}\big(\theta, x^t, y\big)\Big)\Big),$$

where x^t is the adversarial example step t, α is the step size, and Π_ϵ is the projection operator that forces the constraints on the perturbation size δ. The perturbation size is bounded by the ℓ_p-norm, and the typical choice of p is in $\{1, 2, \infty\}$.

DAG. A dense adversary generation (DAG) is an adversarial-example crafting algorithm designed to alter the classification result of an object detector [37]. The object detector consists of two components: a component for semantic segmentation and an object classifier. Given an image, the segmentation component identifies proposals with rectangular areas containing objects, and the classifier outputs the class among N target objects for each proposal. DAG attack crafts δ such that, when δ is added to an input x, it decreases the confidence score of the correct label while increasing that of the incorrect target label. The objective for crafting an adversarial example $x + \delta$ is as follows:

$$\mathcal{L}(\theta, x + \delta, y) = \sum_{i=1}^{N} \Big[f_{y_i}(x + \delta, t_i) - f_{y'_i}(x + \delta, t_i) \Big],$$

where y'_i is the incorrect label an adversary chooses, y_i is the correct label, t_i is each target object for $x + \delta$, and $f_{y_i}(x + \delta, t_i)$ is a confidence score of the class y_i for t_i. To generate the adversarial perturbation, the attacker generates proposals more densely and chooses 'positive' ones: those where the intersection over unions (IoU) matches the ground truth among the objects identified by semantic segmentation component. DAG also chooses the target label y'_i, different from the original label y_i for all positive proposals. The attack finally employs PGD to optimize the objective $\mathcal{L}$.

Minimizing $\mathcal{L}$ involves ensuring that every object an adversary target is misclassified into the incorrect target label y_i'.

Our work adapts both PGD and DAG attacks to test the actual robustness of PhishIntention and DynaPhish, a state-of-the-art reference-based phishing detector. We propose a series of non-trivial adaptations to these algorithms: the design of new attack objectives (§4), the conversion of perturbations in floating-point numbers into integer-valued pixels (§5), and their extension to Expectation over Transformations [13, 22] (§5).

3 Existing Robustness Evaluation

Now we start by identifying common issues in evaluating the robustness of reference-based phishing detectors, particularly those employing neural networks. We first adapt the threat model commonly used in prior research on adversarial attacks. We then categorize three common mistakes we identify from previous studies that lead to limited evaluation of adversarial robustness.

3.1 Threat Model

We consider an adversary who aims to evade a target reference-based phishing detector. The adversary's goal is to cause the classifiers in the target detector to misclassify the input website, leading the detector to incorrectly identify a phishing website as benign. Considering the nature of a phishing campaign, the adversary is required to use a phishing logo that is visually near-identical to an authentic service vendor logos (e.g., Google and PayPal). We note that our adversary differs from the evasive attackers studied in prior research on phishing blocklist evasion [7, 29, 30, 41, 42] (e.g., attackers who exploit cloaking or URL redirection).

Capabilities. We assume that the adversary leverages small perturbations to cause misclassifications of their phishing website. For this, the adversary can manipulate their phishing website by changing the web page layout and introducing perturbations into their phishing logo and the display area. For example, they can inject adversarial perturbations into a logo image or use `iframe` components to overlay them on top of the web page rendering area. In a few cases, the perturbations applied by our adversary may be visually-perceptible. But since phishing detectors typically operate automatically with minimal human intervention, the presence of human-perceptible perturbations does not pose a significant issue. Moreover, we assume that the attacker can employ further techniques to evade human detection, such as making `iframe` components containing the perturbations disappear shortly after the phishing website is rendered. It ensures that screenshots are taken while humans remain unaware of the evasive perturbations.

Knowledge. Our objective is to *audit* adversarial robustness of the state-of-the-art reference-based phishing detector. We thus mainly assume a white-box adversary who knows the internals of phishing detectors, such as the detection pipeline, model architectures, and model parameters, such as weights and biases. We also test black-box attacks, such as logo masking, shown in prior work [3] as techniques that practical adversaries are likely to employ.

3.2 Undesirable Practices in Prior Work

We conduct an extensive review of the literature on reference-based phishing detectors. We find *undesirable* practices in evaluating adversarial robustness, especially those warned by prior work on adversarial attacks [4, 10]. We categorize them into three: defenses leveraging gradient masking (P1), evaluating with weak attacks (P2), and no end-to-end evaluation (P3). Table 1 shows the categorization.

Table 1: Summary of common issues in evaluating the adversarial robustness, found in prior work on phishing detectors.

Detector	Problems		
	P1 (Masking)	P2 (Weak attacks)	P3 (No E2E)
VisualPhishNet [1]	-	✓	-
Phishpedia [24]	✓	✓	✓
PhishIntention [25]	✓	✓	✓
DynaPhish [26]	✓	-	✓

P1. Gradient masking. As phishing detectors will be deployed in adversarial settings, recent work [24–26] evaluates the robustness of these detectors under adversarial pressure and proposes defenses to mitigate such risks. These detectors adapt a commonly used activation function, ReLU, $f(x) = max(0, x)$ to StepReLU, $f(x) = max(0, \alpha \cdot \lceil \frac{x}{\alpha} \rceil)$. StepReLU quantizes the activation values, making it difficult for an adversary to compute the input gradients through back-propagation. However, the effectiveness of such gradient-masking relies on the assumption that the attacker is not aware of its presence. If the attacker knows that a model uses StepReLU, they can develop adaptive attacks that bypass this defense mechanism. Prior work [4] pointed out that obfuscating gradient (e.g., gradient masking) offer a false sense of security. Sophisticated adversaries can circumvent the obfuscation, leaving the phishing detectors vulnerable despite them seemingly robust to adversarial attacks.

P2. Using weak attacks. Prior work [1, 24, 25] evaluates adversarial robustness using weak attacks. Liu et al. [25], for example, uses the DAG attack but with an incorrect implementation; our correct implementation of the DAG attack shows zero robustness, in contrast to their results indicating some robustness. We highlight these implementation differences in Appendix A.2. The evaluation often contains attacks like FGSM [15] or DeepFool [28], which are known to be weaker than more recent attacks such as C&W [10] or PGD [27]. Moreover, most prior work does not consider an adaptive adversary who can either design a crafting objective tailored to a target model or is aware of the defenses deployed to the target. This oversight can lead to an underestimation of the model's vulnerabilities, as adaptive adversaries are likely to be more effective in bypassing defenses and fooling the phishing detectors [5, 19].

P3. No end-to-end attacks. DynaPhish's design principle is to make the detection work in adversarial settings by incorporating components that keep the reference list up-to-date. However, the robustness evaluation only focuses on the components employing neural networks, while their end-to-end (E2E) detection capability has remained unknown. This limited scope of evaluation fails to provide a comprehensive understanding of how user-facing defense systems will perform under adversarial pressure, potentially overlooking the difficulty of adversaries fooling a target.

4 Our Adversarial Attacks

We first follow the common practices in the original studies of PhishIntention and DynaPhish on evaluating adversarial robustness: individually testing the robustness of key components. Our evaluation focuses on their three key components: the object detector, the OCR-Siamese model, and the CRP classifier.

4.1 Evading Object Detection

Most phishing detectors employ object detectors, such as Faster R-CNN [32], to identify various components in a website screenshot, including logos, buttons, or forms. Because the subsequent workflow heavily depends on the identified objects, we first test the robustness of detectors against evasion attacks specifically designed to fool these object detectors. We design our attack to address two common problems, P1 and P2, as outlined below.

- **P1:** We modify the DAG attack, to employ the backward pass differentiable approximation (BPDA) [4]. This technique is typically used to break gradient masking defenses, such as StepReLU. BPDA approximates gradients through non-differentiable components, enabling the attack to effectively bypass defenses that rely on gradient masking and thereby evaluate the true robustness of the object detectors.
- **P2:** The original DAG attack aims to cause misclassification of an object into a target class as well as misidentification of the object area in an image. However, in the context of phishing detection, we found that the attack is much stronger and more effective when they make it impossible for a target object detector to identify any object from an input image. To this end, we develop a new objective function designed to achieve this goal.

Object-removal attack. We first use the ReLU activation function during the backward pass as a differentiable proxy for StepReLU. We then devise the objective function for crafting adversarial perturbations to remove objects from being detected, as follows:

$$\arg\min_{\delta} \sum_{i=1}^{N} \sum_{c \in C} f_c(x + \delta, t_i),$$

where C is the set of target object classes that the attacker aims to evade detection for. This objective is designed to minimize confidence scores across all classes in C for each target object t_i. Unlike the original DAG, we optimize all proposals generated by semantic segmentation because the goal is to erase all detected target class objects regardless of the position of the object.

Methodology. We randomly choose 901 website screenshots from the validation dataset of the object detector provided by Liu et al. [25]. For each screenshot, we run the object-removal attack, which aims to evade all class outputs. We measure the attack success when the number of objects that a detector identifies from a screenshot becomes zero. We vary the number of attack iterations in $[100, 800]$. The attack hyper-parameter γ is set to 0.5.

Table 2: Result from object-removal attacks. We show the increase in attack success over multiple attack iterations. The perturbations are added to the entire screenshots. We set the non-maximum suppression threshold of the region proposal network to 0.9 and the perturbation weight γ to 0.5.

Iteration N	# Screenshots	No object
No attack	901	0%
100	472	47.6%
200	162	82.0%
400	47	94.8%
600	49	94.6%
800	40	95.6%

Results. Table 2 summarizes our attack results. After running 400 attack iterations, our attack achieves a success rate of 94.8%, rendering the object detector ineffective most of the time. We further increase the attack iterations, but there is a marginal increase in attack success (95.6%). We want to highlight that, in contrast to the results from the original study, our work demonstrates that StepReLU does not provide *any* adversarial robustness to the object detector. Figure 2 illustrates a successful attack: the left figure shows the detector identifying multiple objects in a screenshot, while the right figure demonstrates our adversarial perturbation causing the detector to fail to identify any objects. Replacing the backward propagation function in BPDA from the default ($y = x$) to ReLU is sufficient to break the robustness of the object detector.

4.2 Evading Logo Classification

The next step is typically brand recognition. Recent detectors, such as PhishIntention and DynaPhish, propose the OCR-aided brand recognition model (henceforth referred to as the OCR-Siamese model), which employs a two-step detection process. The first step is to extract visual and textual features from a logo using ResNetV2-50 [18, 21] and the OCR model [33], respectively. Then they feed these features into fully-connected layers to extract a single, consolidated feature. The detectors then compare the feature computed from logos in a screenshot with the pre-computed features of various known-benign logos in the reference list. They run this comparison only with the top-3 features that are closest to those computed on the identified logo. Here, we develop an adversarial attack to tackle the two common problems in prior work, P1 and P2, thereby testing the true robustness of this detection process.

- **P1:** Our attack operates under the assumption that StepReLU activations in a model can be replaced with ReLU. When attacking, we consider adversarial perturbations crafted on the model with this adaptation against the original model with the StepReLU defense. We surprisingly find that this adaptation does not guarantee robustness against our attack.
- **P2:** Unfortunately we could not find how the original study [25] adapts existing adversarial attacks to the OCR-Siamese model from their paper and source code. However, we hypothesize that the attacks used in prior work are weaker than our proposed attack. Similar to our object-removal attack, we develop a new

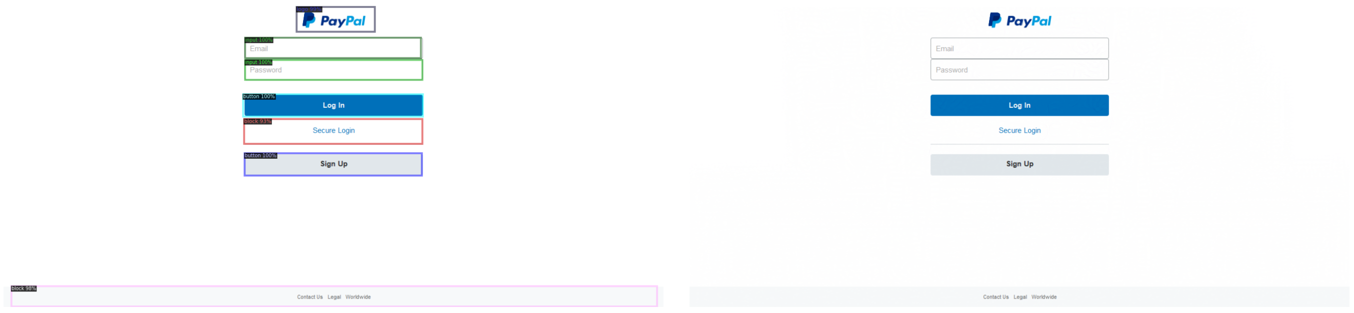


Figure 2: An example demonstrating the success of the object-removal attack. The left shows the detector identifies objects in a clean screenshot, while the right shows the detector failing to identify as we add our adversarial perturbations.

objective designed to reduce the similarity between the features computed from identified logos and those in the reference list.

Evasion attack. To bypass the OCR-Siamses model, we design the objective function for crafting adversarial examples as follows:

$$\arg\min_{|\delta|\leq\epsilon} \sum_{i=1}^{N} cos(f(x+\delta), f(x_i))$$

where f is the OCR-Siamese model, $x+\delta$ is an adversarial example, x_i is the i-th logo in our reference list, N is the cardinality of the reference list (i.e., 3061), and $cos(\cdot)$ is the cosine similarity function. Our objective is designed to minimize cosine similarity scores between the features across all the logos x_i in the reference list. We run our attack using the PGD algorithm [27] as our optimization with an ℓ_∞-norm of 16 pixels to craft adversarial examples.

Methodology. We evaluate our attack on the 30k phishing websites collected and shared by Liu et al. [25]. For each website screenshot, we run our evasion attack and measure whether there is a match between the identified logo and the brand in our reference list. If no match is found, we consider it an attack success. We use the same threshold of 0.87 in the cosine similarity score between two features as the original study when determining a match between the two logos. To set the attack hyper-parameters, we run our attack on 100 randomly selected screenshots with attack iterations ranging from 100–300. As shown in our initial investigation in Table 3, we observe that increasing attack iterations beyond 150 provides no additional benefit. We therefore set N to 150.

Table 3: Result of out evasion attack on OCR-Siamese model. We use different attack iterations, ranging from 100–300.

Iteration N	Before alignment		After alignment	
	Phish	Benign	Phish	Benign
No attack	88	12	89	11
100	0	100	27	72
150	0	100	16	84
200	0	100	16	84
300	0	100	16	84

Table 4: Results of evasion attack on the OCR-Siamese model. We add human-imperceptible adversarial perturbations to phishing website screenshots. We use 150 attack iterations, and the perturbations are bounded to ℓ_∞-norm of 16 pixels.

Iteration N	# Screenshots	Before alignment			After alignment		
		Phishing	Benign	Recall	Phishing	Benign	Recall
No attack	29,496	26,436	3,060	0.90	25,753	3,743	0.87
Ours	28,856	670	28,186	0.02	3,486	25,370	0.12

Results. Table 4 summarizes our results. PhishIntention (that is also used as a module in DynaPhish) runs the detection in two steps: It first compares the feature computed from the logo in a screenshot with the pre-computed features in the reference list. If the detector fails to identify the logo's brand, they *align* the resolution of the identified logo with the logos in the reference list and rerun the comparison. We denote the attack success in the first step as "Before alignment" and in the final result as "After alignment".

Without any adversarial perturbations, out of 29,496 phishing websites, 25,753 (87.3%) websites are identified as phishing after the second step. The remaining 3,743 websites (12.7%) are classified as benign. Logos are not detected in 640 websites, which are also classified as benign. It is interesting to observe that the detection is more successful, identifying 26,436 websites (89.6%) as phishing.

Under our evasion attack, out of 28,856 phishing websites that have logos, 25,370 websites are detected as benign after the second step, demonstrating an attack success of 87.9%. Only 12.1% (3,486 websites) are correctly classified as phishing under our attack. The alignment slightly helps the detection. Before the alignment, 97.7% phishing websites are classified as benign, while 2.3% are still detected as phishing. Our results are in contrast to the findings in the original study, which claimed that StepReLU completely mitigates adversarial attacks. To understand this gap, we further analyze when our attacks become successful/unsuccessful in §6.

4.3 Evading CRP Classification

If the object detector cannot find any logos, the detectors perform the last step: web interaction. The key component of this module is the credential-requiring page (CRP) classifier. Given a website screenshot, the CRP classifier takes the detected bounding boxes and their class predictions from the object detector as input and identifies whether the website asks for user credentials or not. If the classifier identifies that the website is requesting the user's login credentials, the web interaction module provides random credentials to see if the website allows a login. Otherwise, the module finds links that potentially transit to credential-requesting pages. The module then runs the same step with the CRP classifier to determine if a login can be achieved with random credentials. We develop an attack to test how robust the CRP classifier is against bypassing attempts. Our attack exploits the common problem: P1.

- **P1:** The same StepReLU is used by the prior work, when testing their robustness to existing adversarial attacks. We assumed that the StepReLU activation function is used in the model. We craft adversarial perturbations on the model with this adaptation.

Methodology. We evaluate by running our attack optimizing with the PGD [27] algorithm on the CRP classifier, substituting StepReLU with ReLU. We use the same 30k phishing website dataset used in §4.2. We consider an attack successful if the classifier fails to identify a screenshot with adversarial perturbation as a CRP. We set N to 30, as this was found to be the most effective according to Table 5, and the perturbation bound to 16 pixels in ℓ_∞-norm.

Table 5: Result of out evasion attack on the CRP classification model. We perform our attack with 10–35 iterations.

Iteration N	Non-CRP	CRP
No attack	0	100
10	49	51
15	66	34
20	77	23
25	83	17
30	88	12
35	87	13

Table 6: Result of the evasion attack on the CRP classifier. The same model is used in PhishIntention and the DynaPhish's Web Interaction module. We use 30 attack iterations and the perturbations are bounded to ℓ_∞-norm of 16 pixels.

	Iteration N	Not a CRP	CRP
No attack	-	122 (0.4%)	29,374 (99.6%)
PGD	30	24,802 (84.1%)	4,694 (15.9%)

Results. Table 6 summarizes our results. We attack the CRP classifier model employed by the web interaction module in PhishIntention and DynaPhish. Before the attack, 122 out of 29,496 phishing websites are classified as non-CRP websites (0.4%), while 29,374 are classified as CRP websites (99.6%). Under the iteration of 30, the model classifies 24,802 websites as non-CRP websites, showing an attack success of 84.1%. In contrast to the results demonstrated by the original study, our findings show that the CRP classifier is not robust to adversarial attacks, even in the presence of StepReLU.

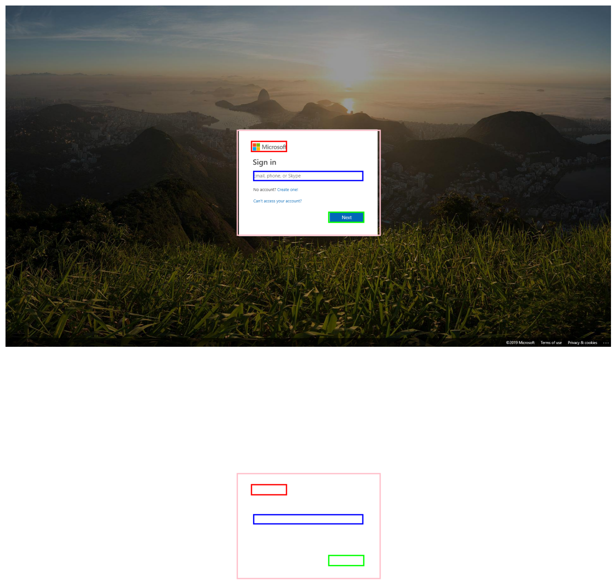


Figure 3: An example illustrating the weakness of the CRP classifier. Both screenshots are classified as CRP. We take the bounding boxes from the upper screenshot and apply them only to a blank image with the same resolution.

We further analyze the factors contributing to the susceptibility of the CRP classifier to adversarial attacks. As observed from our object-removal attack (§4.1), the adversarial perturbations can make it difficult for an object detector to identify several objects in a screenshot. The disappearance of these objects from the input for the CRP classifier is not particularly a problem in detection. However, we find that the classifier heavily relies on a specific pattern among the objects identified in the scene. Figure 3 illustrates this weakness. The upper figure shows a screenshot where the classifier correctly identifies it as a CRP. The bounding boxes are located around objects such as the logo, the forms requesting user account information, and the login button. We then remove the backgrounds and keep only the bounding boxes, as shown in the lower figure. We find that this screenshot, with only the bounding boxes, is still classified as a CRP. This implies that the classifier depends more on the location of the bounding boxes rather than "what these bounding boxes are." If an adversary successfully removes a few bounding boxes from the patterns commonly observed in login forms, they may easily bypass the CRP classifier. Inspired by this observation, we show the effectiveness of our object-removal attack in bypassing the CRP classifier in §5.7, especially in end-to-end settings. Our results suggest that while bounding box locations are sufficient to identify the CRP in benign settings, this heavy reliance can be exploited by adversaries for evasion. We leave further exploration of strategies to mitigate this reliance as future work.

5 Attacks on End-to-End Detection

In the previous section, we presented evasion attacks on each of the three key components of recent reference-based phishing detectors. We now turn our attention to how an adversary can bypass the entire detection system that combines these key components in an end-to-end (E2E) manner. We present three E2E attacks and use them to evaluate the detectors holistically.

5.1 Attack Overview

Figure 4 illustrates how our E2E attacks are expected to work. Our first attack (no-object attack, whose workflow is shown in red) aims to fool the object detector by making it difficult to recognize any objects in a website screenshot. This detection failure can potentially impact the functionality of the subsequent CRP classifier and cause the misclassification of a phishing website as benign.

Our second attack (triple-jump attack, whose workflow is shown in blue) bypasses three mechanisms. First, the object detector recognizes a logo in a screenshot. Next, the attack fools the OCR-Siamese model into classifying the identified logo as a different, perceptually dissimilar logo or making the model impossible to find the matching logo. The attack is finally designed to bypass the CRP classifier, ensuring it is not detected as a credential-requiring page.

Our third attack (logo masking attack, in green) operates in a black-box manner and partially masks the logo in the screenshot. This type of attack is increasingly common, as adversaries seek to simplify their approach with little or no knowledge of the target detector [3]. The process begins with the object detector locating the logo in the screenshot. The logo is then partially masked with white in a random location and replaced with the masked version. This modification can disrupt the OCR-Siamese model, making it difficult for the model to find a matching logo in the reference data.

5.2 Challenges in E2E Evaluation

We identify challenges in end-to-end evaluation against reference-based phishing detectors and show how we address them.

Overlaying adversarial perturbations. To apply adversarial perturbation on a real-world phishing website, we leverage HTML features, such as `iframes`. Using such features, it is easy for an adversary to overlay the perturbation onto a website at a lower cost than directly manipulating all the components in the website's HTML code. Because our gradient-based attacks iteratively minimize (or maximize) the attack objectives, we incorporate the following additional overlaying steps in each iteration:

$$x + \delta = x * (1 - \alpha) + \delta * \alpha,$$

where α is the blending factor we use for overlaying, and x, δ are the original screenshot and the perturbation at each step, respectively. At each iteration, we additionally clip and round the perturbation δ to the nearest pixel values, which are integers between 0–255.

Incorporating data augmentations. Phishing detectors, such as PhishIntention, often resize a screenshot to various resolutions and perform detection. Under such data augmentations, adversarial attacks become less effective [34] than those operating in the floating point space, and so do our attacks. To address this challenge, in the optimization process, we convert the resolution of an input screenshot to the resolutions a detector will examine. We then upsample the gradients to the original resolution of the screenshot using the `Resize()` function with bilinear interpolation supported by PyTorch library. This adaptation can be viewed as a simplified form of Expectation over Transformation (EoT), as proposed in prior work [13, 22]. It is worth noting that this adaptation can be extended to include additional data augmentation methods. We leave the exploration of such generalizations as future work.

Practicality of the experimental setup. Real-world adversaries often lack complete knowledge of the target detector. Our attacks are designed to show that evasion attacks can remain effective even when the attacker lacks full knowledge of the target detectors. Our first two attacks—No Object and Triple-Jump attacks—only require the full knowledge of only specific components. For example, the No Object attack assumes the knowledge of the object detector, such as Fast R-CNN, which is publicly available on the Internet. The Logo Masking attack does not require any knowledge.

5.3 Experimental Setup

An end-to-end evaluation of recent phishing detectors, such as DynaPhish, requires actual phishing websites hosted under their original URLs. Because these phishing websites have already become inactive, this is not feasible in our settings. We thus exclude the following detection processes from our evaluation:

- **CRP transition:** In the web interaction module, DynaPhish first checks whether the current website is a CRP. If not, the module searches for buttons or hyperlinks that will redirect a user to a CRP. Because we do not have the HTML code for the redirected websites, we run our evasion attacks on CRPs.
- **Credential verification:** The same module automatically examines whether a CRP runs verification of user credentials (such as username and/or password). If the website allows a login with a random credential, the module marks the website as phishing. We assume the process will always be successful in any CRPs.

This enables us to provide a conservative estimate of evasion attack success since we do not account for attack success resulting from the failures in the CRP transition or credential verification steps.

Metrics. We measure *attack success* as the percentage of phishing websites misclassified as benign after perturbations are applied. Because our dataset for evaluating end-to-end attacks consists only of phishing websites, attack success can also be interpreted as *recall*.

5.4 No-object E2E Attack

Methodology. We run our object-removal attack in an E2E manner. We evaluate on 5k phishing website randomly chosen from the 30k phishing websites [25]. We first craft an adversarial perturbation for each screenshot and store it as a `.png` file. We then load each file and overlay to the actual website screenshot.

Results. Table 7 summarizes the results of the no-object attack. When processing the screenshot image, PhishIntention and DynaPhish resize the image to the resolution the model receives as input. Under the condition that the input screenshot image is resized as the model's resolution, 4,895 out of 5,000 phishing websites are classified as benign. We observe a significant decrease in recall,

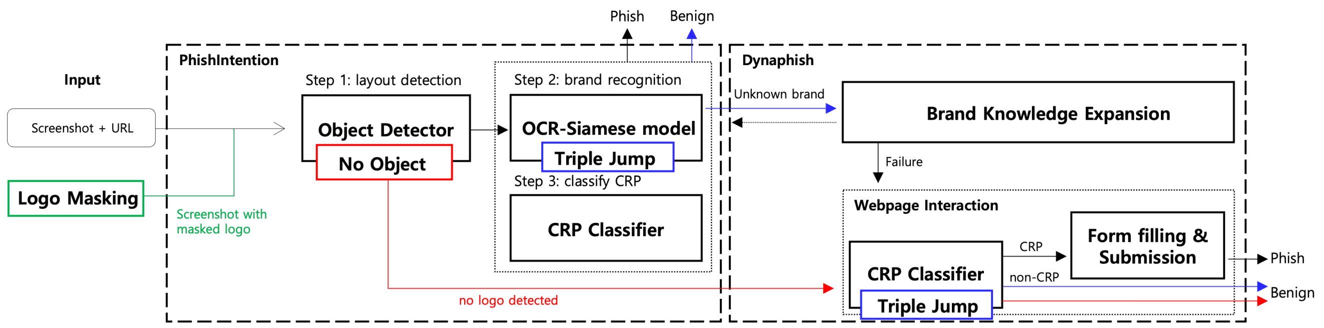


Figure 4: Illustration of how our end-to-end attacks render the key components of PhishIntention [25] and DynaPhish [26], ineffective. The red, blue, and green lines represent the workflow of our first, second, and third attacks respectively.

Table 7: Results of no-object E2E attack on DynaPhish. We set an overlay factor to 0.2.

E2E Attacks	Phishing	Benign	Recall
None	3,812	1,188	0.76
No Object (overlay)	105	4,895	0.02
No Object (overlay and upsampling)	174	4,826	0.03

dropping from 0.76 to 0.02. When we use the input screenshot and the perturbation whose resolution has been restored to the original screenshot size (third row in Table 7), 4,826 out of 5,000 phishing websites are classified as benign. We show a reduced recall, dropping to 0.03, which is slightly higher than the previous result that only considers overlaying. This is because resizing the perturbation, in which the original size will be the model's resolution, to the size of the original screenshot image can cause an unintended adjustment to the perturbation.

5.5 Triple-jump E2E Attack

Methodology. We next adapt our component-wise attacks for E2E evaluation and propose Triple-jump attack. We add the two objectives that we use for evading the OCR-Siamese and CRP classification models and craft the adversarial perturbations. We run our Triple-jump attack on randomly-chosen 2,000 websites.

Table 8: Results of Triple-jump E2E attack on DynaPhish. We set an overlay factor to 0.5.

E2E Attacks	Phishing	Benign	Recall
No attack	1,554	446	0.78
Triple-jump	654	1,346	0.33

Results. We show the results of our Triple-jump attack in Table 8. Out of the 2,000 phishing websites we manipulate, DynaPhish identifies 1,346 as benign, showing an attack success rate of 67.3%. Additionally, the recall dropped significantly from 0.78 to 0.33. Compared to the scenario where we directly attack the OCR-Siamese (§4.2) or the CRP classification model (§4.3), the attack success rate is decreased by 21–32%. However, we see the attack success rate is still high, implying that an adversary with the objective of bypassing automated phishing detection (e.g., DynaPhish) will be successful in real-world deployment settings.

5.6 Logo Masking E2E Attack

Methodology. We lastly test a logo masking attack that partially obscures the logo in a screenshot, targeting the OCR-Siamese model in a more straightforward and cost-effective manner while visually easier to identify the attack artifacts. The masking area is randomly selected, with sizes tested at $\frac{1}{6}$, $\frac{1}{4}$, $\frac{1}{3}$, and $\frac{1}{2}$ of the original logo dimensions. The width and height of the masking area, denoted as M_w and M_h are computed as follows:

$$M_w = \left\lfloor \frac{L_w}{w} \right\rfloor \text{ and } M_h = \left\lfloor \frac{L_h}{h} \right\rfloor$$

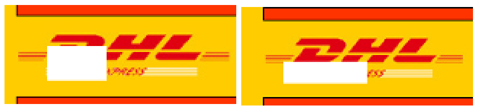
where L_w and L_h are the width and height of the original logo, respectively. The values of w and h are integers selected to satify the condition $w \times h \in \{2, 3, 4, 6\}$. Our attack was evaluated on a randomly chosen set of 1,963 phishing websites. This dataset is the same as that used in the Triple-jump attack experiment, excluding 37 websites where the object detector failed to detect the logo.

Table 9: Results of the logo masking E2E attack on DynaPhish. The size of the masking area is determined by the formula 5.6, with its location randomly selected.

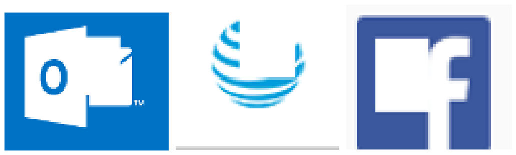
Masked area					
$w \times h$	w	h	Phishing	Benign	Recall
No Masking			1,554	409	0.79
6	3	2	1,528	435	0.78
	2	3	1,533	430	0.78
4	2	2	1,498	465	0.76
3	3	1	1,341	622	0.68
	1	3	1,500	463	0.76
2	2	1	945	1,018	0.48
	1	2	1,448	515	0.74

Results. Table 9 summarizes the result of the logo masking attack experiment. The overall attack success rate was slightly higher than the ground truth of 20.8%, with an average success rate of 29.2%. The highest success rate was observed when the masking area covered half of the logo, with $w = 2$ and $h = 1$, achieving a 51.9% success rate (1,018 out of 1,963).

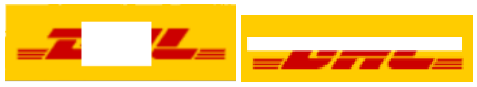
We further analyze the masked logos that successfully bypassed DynaPhish. Figure 5 illustrates the masked logos that are classified as either phishing or benign. As shown in Table 9, we observe that the success rate is higher when w is greater than h. Moreover, the attack is more likely to succeed when the masked logo consists of a single icon without any accompanying brand name or explanatory text (see Figure 5b). When the masking area obscures portions of the logo text across multiple letters, rather than fully concealing a single random letter, the detector is more likely to misclassify the input website as benign (as shown in Figure 5c).



(a) `DHL` logo with masking applied at $w = 2$, $h = 3$, and $w = 3$, $h = 2$, respectively. The website with the logo on the left is detected as phishing, while the website with the logo on the right is detected as benign. Websites where the logo masking has $w > h$ are more likely to be classified as benign.



(b) `Outlook`, `AT&T`, and `Facebook` logos with masking applied at $w = 3$ and $h = 2$. All three websites featuring these logos are detected as benign. Websites with icon-only logos are more likely to be classified as benign.



(c) `DHL` logo with masking applied at $w = 1$, $h = 3$, and $w = 3$, $h = 1$, respectively. The website with the logo on the left is detected as phishing, while the website with the logo on the right is detected as benign. Websites where the logo masking partially obscures multiple letters of the text are more likely to be classified as benign compared to those where a single letter is fully covered.

Figure 5: Logo images with random masking. The logo on the left in Figure 5a and Figure 5c is detected as phishing, while the logo on the right in both figures is detected as benign. All three logos in Figure 5b are detected as benign.

5.7 Impact on Additional Mechanisms

Brand Knowledge Expansion. DynaPhish employs the Brand Knowledge Expansion module to keep the reference list up-to-date. Once DynaPhish fails to find a match between the identified logo and those in the reference list, the module runs Google's OCR and Logo detection models to identify the brand name. Using the brand name, they use it as a keyword for Google search and download a screenshot of the first website appearing in the search results. The module then crops the logo from the screenshot and stores the logo and its brand name in the reference list. We test if the Brand Knowledge Expansion module can help render our attacks ineffective.

We run our attacks tailored to evading the OCR-Siamese model (§4.2) on the Brand Knowledge Expansion module. Table 10 summarizes our results. Without the module, the attack success rate is 87.9% (i.e., 25,370 websites are detected as benign out of 28,856 in total). When we add the module, the attack success rate becomes 84.2% and the recall improves to 0.16. We observe that there is only a marginal decrease (3.7%) in the attack success rate and a modest increase in the recall (0.04). This implies that the newly added mechanism, Brand Knowledge Expansion, does not particularly enhance the detection performance, especially in adversarial settings.

Table 10: Results of our attack on the Brand Knowledge Expansion (BKE) module. We use the attack presented in §4.2.

	Iteration N	Phishing	Benign	Recall
PhishIntention	150	3,486	25,370	0.12
PhishIntention with BKE	150	4,553	24,303	0.16

We analyze the factors attributing to our observation. We find that the Brand Knowledge Expansion does not necessarily identify the correct logo from Google search. For instance, the module performs a Google search for the name of a small bank. But the search returns a large company's logo as the company collaboratively runs an event with the small bank. What is eventually stored in the reference list for the brand name is not the bank's logo, but the logo of the large company. Even if we run the OCR-Siamese detection with the updated reference list, the result will be still no match.

Web Interaction. The Web Interaction module heavily relies on the results from the object detector. It runs the CRP classification model with the objects identified by the object detector model. We thus ask: *Can we render the module ineffective by suppressing the identification of objects in a screenshot?* To answer this question, we select 384 screenshots with a layout from the validation dataset of the object detector and CRP classifier [25], all of which were labeled as CRP. We then craft the adversarial perturbations to remove object classes gradually from detection and measure the attack success as the performance of the target CRP classifier.

Table 11 shows our results on removing specific objects. We observe that the Logo class is not critical in the CRP classification: even without detecting logos, the classifier correctly detects 372 as CRP out of 384 websites. We find that the Input and Button classes are important. Without the both, the classifier is only able to detect 144 websites as CRPs. The Block objects are less critical for detection. Removing the block objects is only able to flip the classifier's decision for a single website. The last row shows that our adversary can create perturbations to remove all objects from detection, rendering the CRP classification completely ineffective.

We further run our attack that removes all the objects on 10k phishing webpage screenshots classified as CRPs. Our attack can

Table 11: Results of object-removal attack remove specific classes. We measure the attack's impact on the CRP classifier. ✗indicates the object class we optimize to remove.

Class removed					Detection		
Logo	Input	Button	Label	Block	CRP	Non-CRP	Recall
					376	8	0.98
✗					372	12	0.97
✗	✗	✗			144	240	0.38
✗	✗	✗	✗		1	383	0.00
✗	✗	✗	✗	✗	0	384	0.00

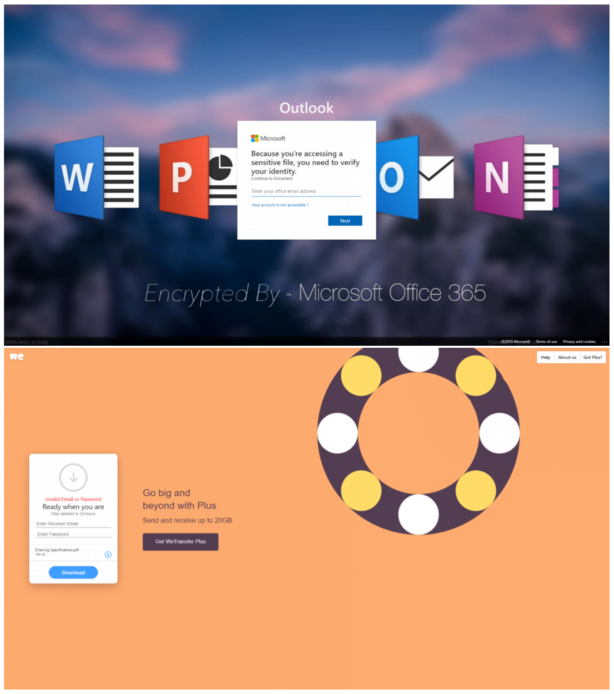

Outlook
Microsoft
Because you're accessing a sensitive file, you need to verify your identity.
Next
Encrypted By - Microsoft Office 365
Go big and beyond with Plus
Send and receive up to 20GB
Get WeTransfer Plus
Download

Figure 6: Illustrating two screenshots with perturbation where our attack fails to evade the CRP classifier even if the object detector fails to identify *any* object in the images.

evade the CRP classification with a success rate of 99.8% (9,975 of them are classified as non-CRPs). Our attack works on most screenshots, but there are 25 cases where the CRP classifier identifies the screenshots as CRPs. These cases are interesting because, although we made all objects disappear from the screenshots, the CRP classifier still correctly identified them. We showcase two cases in Figure 6. We hypothesize that in most cases, the CRP classifier focuses on the object layout. But there are a few cases where the model focuses on patterns in the backgrounds.

6 Discussion

We now discuss the effectiveness of potential countermeasures and future work directions for safeguarding Internet users. In the context of machine learning, extensive studies have explored defenses against adversarial attacks [9, 12, 27, 31, 39]. However, it remains unclear whether those defenses are effective in phishing detection. Prior work particularly discusses the shortcomings of existing defenses, such as performance degradation or computational overheads, which, when integrated into the phishing detectors processing millions of suspicious websites daily, could increase the operational overhead substantially (Please refer to the detailed discussion in Appendix A.3). We thus discuss two lightweight defensive strategies to minimize the aforementioned shortcomings.

6.1 Designing Robust Detectors

Because our attacks leverage human-imperceptible perturbations, we test if one can remove these perturbations while preserving detection performance. We focus on *certified* defenses proposed by Cohen et al. [12], which leverage randomized smoothing techniques. Recent work [9] shows that one can reduce its computational overhead by selecting a randomized smoothing method wisely (e.g., using off-the-shelf diffusion denoising models). Inspired by this prior work, we employ a lightweight randomized smoothing technique: Gaussian blurring. This technique can be integrated into existing phishing detectors by applying Gaussian blurring to screenshots before feeding them into the detector.

Methodology. We test 25,753 phishing logos with adversarial perturbations. We denoise them using Gaussian blur. We implement the denoiser with OpenCV and set the filter size to (3, 3). We feed the denoised logos and measure the reduction in evasion success.

Table 12: Detection results after denoising in the E2E setting.

	Iteration N	Kernel Size	Phishing	Benign
Perturbed	150	-	383 (1.5%)	25,370 (98.5%)
Denoised	150	(3, 3)	18,771 (72.9%)	6,982 (27.1%)

Results. Table 12 shows the effectiveness of our denoising. Before denoising, the attack success rate is 98.5%: only 383 logos in our 26k screenshots are detected as phishing. However, when we apply denoising, the attack success significantly drops to 27.1%. Considering that DynaPhish classifies 87.3% screenshots without perturbations as phishing, our cost-effective denoising is slightly below (~14.4% below) under our evasion attacks.

Table 13: Performance of the object detector after denoising.

Attack	Defense	Kernel size	mAP	No object
No attack	**No defense**	-	56.8	-
	Denoising	(3, 3)	53.3	-
Object-removal	**No defense**	-	0.0	95.6%
	Denoising	(3, 3)	34.4	0.3%

We also examine the effectiveness of denoising against our object-removal attacks. We run this evaluation on 901 webpage screenshots used in §4.1. We perturb them with 800 attack iterations. Table 13 summarizes our results. We first show that denoising does not reduce detection performance when there is no adversary. There is a slight decrease in mAP from 56.8 to 53.5. Against the screenshots with perturbations, our denoising increases mAP from 0.0 to 34.4. The **No object** column shows the percentage of screenshots where the detector fails to identify any objects. We observe that the percentage is drastically reduced from 95.6% to 0.3%.

6.2 Designing Robust Reference Lists

In §4.2, we find that some of our attacks fail to manipulate the OCR-Siamese model despite increased attack iterations. We focus on these logos and analyze if there are unique characteristics that particularly make the logos resilient to our attacks. Identifying such characteristics can help popular brands design their logos to be robust against adversary evasion efforts, thereby making it easier for detectors to recognize phishing attempts using their logos.

We showcase such resilient logos in Figure 7. The left column shows the logos successfully fooling the model, whereas the right column shows those that do not. Across the board, we observe that logos with symbols (which are often larger than the brand names in text) are more vulnerable to our adversarial attacks. All logos that fail to cause misclassification of the OCR-Siamese model has the brand name with fonts that are easily human-perceptible, symbols smaller than the brand names, and single-colored backgrounds. For example, in the third row, the `Runescape` logo contains its brand name adorned with a curved line, while the `Rakuten` logo does not have such embellishments. In the fourth row, we also contrast the `Adobe Acrobat` and `PayPal` logos. The Adobe logo, with its somewhat unrelated backgrounds, is vulnerable to our adversarial perturbations. While these brands do not consider phishing attacks when designing their logos, we will share these interesting findings with them. This may encourage these brands to update their logos to a simpler format, similar to how `Instagram` changed their logo from a detailed camera picture to a simpler outline version (even if their intention is not robustifying against phishing attempts). In summary, our results suggest that for brand logos to be robust against evasion, they should adhere to the following design principles: (1) use simple, unembellished fonts, (2) minimize or eliminate the use of logo icons, and (3) employ a solid, uniform background.

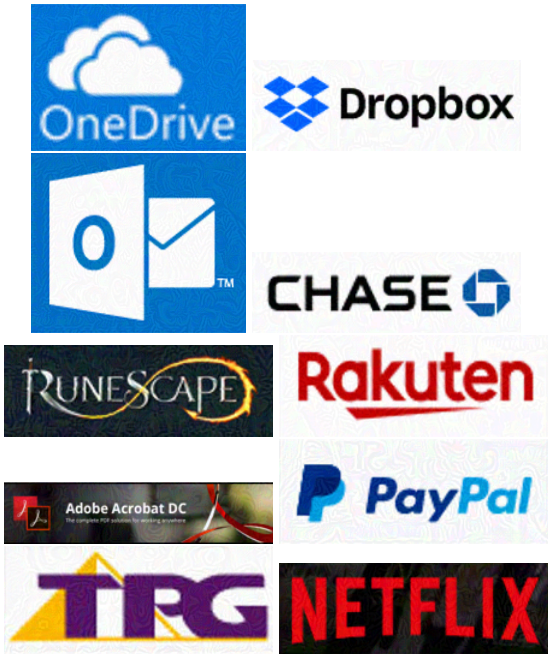


Figure 7: Examples of logos with perturbations. The logos on the left row are the logos with similarities lower than the threshold (Non-robust), and the logos on the right row are the logos with similarities higher than the threshold (Robust).

7 Conclusion

This paper studies the adversarial robustness of reference-based phishing detectors, which widely used machine learning models to automate the detection process. In contrast to the best-known practices in the literature on adversarial attacks, we identify three common mistakes in evaluating the adversarial robustness: gradient masking, employing weak attacks, and not conducting end-to-end evaluations. Our work follows the best-known practices and proposes three evasion attacks. We evaluate these attacks on state-of-the-art phishing detectors and demonstrate that, unfortunately, they are not as robust to adversarial attacks as previously claimed. We analyze factors contributing to these weaknesses and discuss potential ways to improve the resilience of detection systems.

We conclude by outlining future research directions for evaluating robustness of phishing detectors deployed in practice and designing defensive strategies against evasion efforts:

- **Measuring the impact of our attacks on real-world phishing detectors.** A key area for future work is evaluating the robustness of real-world phishing detectors against our attacks. This is a challenging question due to the *black-box nature* of many protection methods in the real-world against Internet threats [6]. However, we believe that because these detectors may adapt pre-trained models publicly available on the Internet, our attacks are likely to transfer and serve as a reasonable starting point of such evaluations. We also note that any such evaluations must be ethically sound and designed with extra care, as highlighted in relevant work [30]. Conducting evasion attacks on real-world detectors may have serious implications for the safety of Internet users.
- **Designing practical defenses against evasion.** While countermeasures against adversarial attacks have been extensively studied, it has been unclear how compatible these proposals with phishing detection. As shown in Sec. 6.1, approaches like certified defenses increase the time it takes to examine a suspicious website. It is thus important future work on how to balance the trade-off between defense effectiveness and operational overhead. In addition, defenses could be built upon lightweight image transformations, such as Gaussian blur we study or JPEG compression shown in [39], particularly when detectors take the screenshot of suspicious websites. Moreover, it is a promising direction to develop measurable metrics and practical guidelines for designing robust logos.

Acknowledgments

We thank our shepherd and the anonymous reviewers for their feedback. We also thank Zane Ma for helpful discussions. This work was partially supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-00153, Penetration Security Testing of ML Model Vulnerabilities and Defense). The findings and conclusions in this work are those of the author(s) and do not necessarily represent the views of the funding agency.

References

[1] Sahar Abdelnabi, Katharina Krombholz, and Mario Fritz. 2020. VisualPhishNet: Zero-Day Phishing Website Detection by Visual Similarity. In *Proceedings of the ACM Conference on Computer and Communications Security*. 1681–1698.

[2] Sadia Afroz and Rachel Greenstadt. 2011. PhishZoo: Detecting Phishing Websites by Looking at Them. In *Proceedings of the IEEE International Conference on Semantic Computing*.

[3] G. Apruzzese, H. S. Anderson, S. Dambra, D. Freeman, F. Pierazzi, and K. Roundy. 2023. "Real Attackers Don't Compute Gradients": Bridging the Gap Between Adversarial ML Research and Practice. In *2023 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*. IEEE Computer Society, Los Alamitos, CA, USA, 339–364. doi:10.1109/SaTML54575.2023.00031

[4] Anish Athalye, Nicholas Carlini, and David Wagner. 2018. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. In *Proceedings of the International Conference on Machine Learning*. 274–283.

[5] Osbert Bastani, Yani Ioannou, Leonidas Lampropoulos, Dimitrios Vytiniotis, Aditya Nori, and Antonio Criminisi. 2016. Measuring neural net robustness with constraints. In *Proceedings of the Advances in Neural Information Processing Systems*. 2613–2621.

[6] Simon Bell and Peter Komisarczuk. 2020. An analysis of phishing blacklists: Google safe browsing, openphish, and phishtank. In *Proceedings of the International Conference on Frontiers in Handwriting Recognition*. 1–11.

[7] Marzieh Bitaab, Haehyun Cho, Adam Oest, Zhuoer Lyu, Wei Wang, Jorij Abraham, Ruoyu Wang, Tiffany Bao, Yan Shoshitaishvili, and Adam Doupé. 2023. Beyond Phish: Toward Detecting Fraudulent e-Commerce Websites at Scale. In *Proceedings of the IEEE Symposium on Security and Privacy*. 2566–2583.

[8] A.S. Bozkir and M. Aydos. 2020. LogoSENSE: A Companion HOG based Logo Detection Scheme for Phishing Web Page and E-mail Brand Recognition. *Computers & Security* (2020).

[9] Nicholas Carlini, Florian Tramer, Krishnamurthy Dj Dvijotham, Leslie Rice, Mingjie Sun, and J Zico Kolter. 2023. (Certified!!) Adversarial Robustness for Free!. In *Proceedings of the International Conference on Learning Representations*.

[10] Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In *Proceedings of the IEEE Symposium on Security and Privacy*. 39–57.

[11] Shuyu Cheng, Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. 2019. Improving Black-box Adversarial Attacks with a Transfer-based Prior. In *Proceedings of the Advances in Neural Information Processing Systems*.

[12] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. 2019. Certified adversarial robustness via randomized smoothing. In *Proceedings of the International Conference on Machine Learning*. 1310–1320.

[13] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. 2018. Robust Physical-World Attacks on Deep Learning Visual Classification. In *Computer Vision and Pattern Recognition (CVPR)*.

[14] Anthony Y. Fu, Liu Wenyin, and Xiaotie Deng. 2006. Detecting Phishing Web Pages with Visual Similarity Assessment Based on Earth Mover's Distance (EMD). *IEEE Transactions on Dependable and Secure Computing* 3, 4 (2006), 301–311.

[15] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. In *Proceedings of the International Conference on Learning Representations*.

[16] Google LLC. 1999. *Safe Browsing - Google Safe Browsing*. https://safebrowsing.google.com/

[17] Anandbabu Gopatoti, Kiran Kumar Gopathoti, Sai Prasanna Shanganthi, and Chappali Nirmala. 2018. Image Denoising using spatial filters and Image Transforms: A Review. *International Journal for Research in Applied Science and Engineering Technology* 6 (2018), 3447–3452. https://api.semanticscholar.org/CorpusID:57598681

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Identity Mappings in Deep Residual Networks. In *Computer Vision – ECCV 2016*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (Eds.). Springer International Publishing, Cham, 630–645.

[19] Dan Hendrycks and Thomas Dietterich. 2019. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. *Proceedings of the International Conference on Learning Representations* (2019).

[20] Andrew Ilyas, Logan Engstrom, and Aleksander Madry. 2019. Prior Convictions: Black-box Adversarial Attacks with Bandits and Priors. In *Proceedings of the International Conference on Learning Representations*.

[21] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. 2020. Big Transfer (BiT): General Visual Representation Learning. In *Computer Vision – ECCV 2020*, Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (Eds.). Springer International Publishing, Cham, 491–507.

[22] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. 2016. Adversarial examples in the physical world. *ArXiv* abs/1607.02533 (2016). https://api.semanticscholar.org/CorpusID:1257772

[23] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. 2018. Noise2Noise: Learning Image Restoration without Clean Data. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 2965–2974. https://proceedings.mlr.press/v80/lehtinen18a.html

[24] Yun Lin, Ruofan Liu, Dinil Mon Divakaran, Jun Yang Ng, Qing Zhou Chan, Yiwen Lu, Yuxuan Si, Fan Zhang, and Jin Song Dong. 2021. Phishpedia: A Hybrid Deep Learning Based Approach to Visually Identify Phishing Webpages. In *Proceedings of the USENIX Security Symposium*. 3793–3810.

[25] Ruofan Liu, Yun Lin, Xianglin Yang, Siang Hwee Ng, Dinil Mon Divakaran, and Jin Song Dong. 2022. Inferring Phishing Intention via Webpage Appearance and Dynamics: A Deep Vision Based Approach. In *Proceedings of the USENIX Security Symposium*. 1633–1650.

[26] Ruofan Liu, Yun Lin, Yifan Zhang, Penn Han Lee, and Jin Song Dong. 2023. Knowledge Expansion and Counterfactual Interaction for Reference-Based Phishing Detection. In *Proceedings of the USENIX Security Symposium*. 4139–4156.

[27] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *Proceedings of the International Conference on Learning Representations*.

[28] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. 2016. DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2574–2582.

[29] Adam Oest, Yeganeh Safaei, Adam Doupé, Gail-Joon Ahn, Brad Wardman, and Kevin Tyers. 2019. PhishFarm: A Scalable Framework for Measuring the Effectiveness of Evasion Techniques against Browser Phishing Blacklists. In *Proceedings of the IEEE Symposium on Security and Privacy*. 1344–1361.

[30] Adam Oest, Penghui Zhang, Brad Wardman, Eric Nunes, Jakub Burgis, Ali Zand, Kurt Thomas, Adam Doupé, and Gail-Joon Ahn. 2020. Sunrise to Sunset: Analyzing the End-to-end Life Cycle and Effectiveness of Phishing Attacks at Scale. In *Proceedings of the USENIX Security Symposium*. 361–377.

[31] Aaditya (Adi) Prakash, Nick Moran, Solomon Garber, Antonella DiLillo, and James A. Storer. 2018. Deflecting Adversarial Attacks with Pixel Deflection. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), 8571–8580. https://api.semanticscholar.org/CorpusID:4528012

[32] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Proceedings of the Advances in Neural Information Processing Systems*. 91–99.

[33] Baoguang Shi, Mingkun Yang, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. 2019. ASTER: An Attentional Scene Text Recognizer with Flexible Rectification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41, 9 (2019), 2035–2048. doi:10.1109/TPAMI.2018.2848939

[34] Chawin Sitawarin, Florian Tramèr, and Nicholas Carlini. 2023. Preprocessors Matter! Realistic Decision-Based Attacks on Machine Learning Systems. In *Proceedings of the International Conference on Machine Learning*.

[35] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *Proceedings of the International Conference on Learning Representations*.

[36] Thij van den Hout, Thymen Wabeke, Giovane C.M. Moura, and Cristian Hesselman. 2022. LogoMotive: Detecting Logos on Websites to Identify Online Scams - A TLD Case Study. In *Proceedings of the International Conference on Passive and Active Measurement*. 3–29.

[37] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. 2017. Adversarial Examples for Semantic Segmentation and Object Detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 1369–1378.

[38] Yutong Xie, Mingze Yuan, Bin Dong, and Quanzheng Li. 2023. Unsupervised Image Denoising with Score Function. In *Thirty-seventh Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=d6LShzSTOP

[39] Weilin Xu, David Evans, and Yanjun Qi. 2018. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*. Internet Society, San Diego, CA, USA. https://doi.org/10.14722/ndss.2018.23198

[40] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. 2017. Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising. *IEEE Transactions on Image Processing* 26, 7 (2017), 3142–3155. doi:10.1109/TIP.2017.2662206

[41] Penghui Zhang, Adam Oest, Haehyun Cho, Zhibo Sun, RC Johnson, Brad Wardman, Shaown Sarker, Alexandros Kapravelos, Tiffany Bao, Ruoyu Wang, Yan

Shoshitaishvili, Adam Doupé, and Gail-Joon Ahn. 2021. CrawlPhish: Large-scale Analysis of Client-side Cloaking Techniques in Phishing. In *Proceedings of the IEEE Symposium on Security and Privacy.* 1109–1124.

[42] Penghui Zhang, Zhibo Sun, Sukwha Kyung, Hans Walter Behrens, Zion Leonahenahe Basque, Haehyun Cho, Adam Oest, Ruoyu Wang, Tiffany Bao, Yan Shoshitaishvili, Gail-Joon Ahn, and Adam Doupé. 2022. I'm SPARTACUS, No, I'm SPARTACUS: Proactively Protecting Users from Phishing by Intentionally Triggering Cloaking Behavior. In *Proceedings of the ACM Conference on Computer and Communications Security.* 3165–3179.

A Appendix

A.1 OCR-Siamese Model

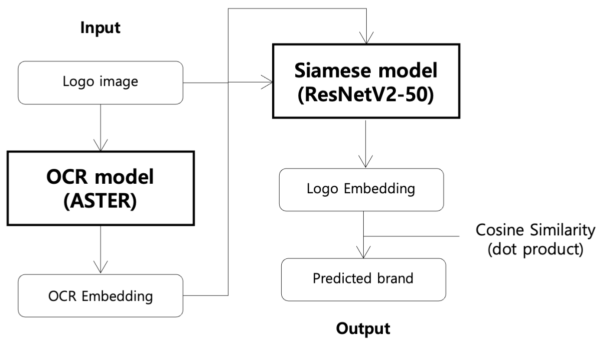


Figure 8: Illustration of the OCR-Siamese model's operation. The input is a logo image, and the output is the predicted brand associated with the logo.

Figure 8 illustrates the detailed internal structure of the OCR-Siamese model. The logo image is initially cropped from the original screenshot using an object detector before being fed into the model. The pre-trained OCR model, which is based on the ASTER architecture [33], extracts text features by recognizing digits and characters. The dimensionality of the OCR embedding matches that of the original input logo feature. These features are then passed to the Siamese model, which is based on the ResNetV2-50 architecture [18, 21], where the input consists of the OCR embeddings and the original input logo image. The Siamese model generates embeddings for both the input logo and the predefined brand list, subsequently calculating the cosine similarity between them using the dot product. The final output is the predicted brand name and its associated similarity score, corresponding to the logo with the highest similarity. If all brand similarity scores fall below a predefined threshold (0.87 in our case), the detector fails to predict a brand and returns None.

A.2 Incorrect Implementation of DAG

When we ran the DAG implemented in PhishIntention [25] and fed the generated adversarial examples back into the model, the normal output was often restored without any adversarial effect. Upon analyzing the code, we found that the DAG structure was implemented as Algorithm 1 shows.

The main problem is that during each iteration, the process of updating **target_boxes** overwrites the variable completely, as shown in the code above. This causes indices excluded in previous iterations to be ignored entirely, leading to incomplete results.

Algorithm 1: DAG Iterative Process

```
Data: inputs, number of iterations $n\_iter$
Result: Updated adversarial labels
...
$target\_boxes, target\_labels$ = get_targets(batched_inputs);
for $i \leftarrow 1$ to $n\_iter$ do
    $logits \leftarrow$ get_predictions(batched_inputs);
    $active\_cond \leftarrow$ argmax(logits) == $target\_labels$;
    ...
    $target\_boxes \leftarrow target\_boxes[active\_cond]$;
    ...
end
```

A.3 Discussion about Adversarial Defenses

Currently, numerous machine learning-based approaches have been proposed to defend against adversarial attacks. Some approaches focus on detecting and mitigating adversarial perturbations in images as a defense method against adversarial attacks. Pixel Deflection, for example, randomizes pixel positions within the input image, thereby neutralizing changes introduced by adversarial attacks [31]. Additionally, Feature Squeezing compresses the input feature space and compares the outputs generated from the original and squeezed inputs. Significant discrepancies between the two outputs indicate the likelihood of adversarial characteristics in the input image [39]. However, these methods must be directly integrated into the phishing detection framework, further increasing its complexity.

Also, in addition to Gaussian blur, numerous traditional and state-of-the-art methods are employed for image denoising. Techniques such as Mean, Median, and Wiener filters calculate pixel values based on the neighborhood pixels, while methods like Wavelet, Contourlet, and Curvelet Transforms provide filtering capabilities for both stationary and non-stationary signals. These approaches are well-established and computationally efficient. However, they often underperform when dealing with images affected by specific types of noise [17].

Recently, advancements in machine learning have introduced new denoising methodologies. For instance, Zhang et al. [40] proposed a deep convolutional neural network (CNN) model for image denoising by training on paired datasets with clean and noisy images. Expanding on this, Lehtinen et al. [23] introduced Noise2Noise, a framework enabling denoising without requiring clean reference images during training. More recently, Xie et al. [38] developed a denoising model based on CNNs that employs multiple score functions to address diverse noise types effectively. Such methods can be applied as a preprocessing step before integrating the images into phishing detection frameworks.