Evaluating the Effectiveness and Robustness of Visual Similarity-based Phishing Detection Models

Fujiao Ji[1], Kiho Lee[1], Hyungjoon Koo[2], Wenhao You[3], Euijin Choo[3], Hyoungshick Kim[2], Doowon Kim[1]

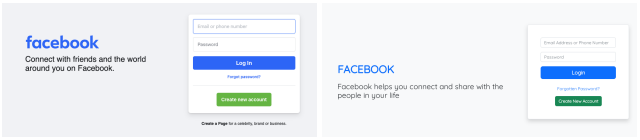
[1]*University of Tennessee, Knoxville* [2]*Sungkyunkwan University* [3]*University of Alberta*

Abstract

Phishing attacks pose a significant threat to Internet users, with cybercriminals elaborately replicating the visual appearance of legitimate websites to deceive victims. Visual similarity-based detection systems have emerged as an effective countermeasure, but their effectiveness and robustness in real-world scenarios have been underexplored. In this paper, we comprehensively scrutinize and evaluate the effectiveness and robustness of popular visual similarity-based anti-phishing models using a large-scale dataset of 451k real-world phishing websites. Our analyses of the effectiveness reveal that while certain visual similarity-based models achieve high accuracy on curated datasets in the experimental settings, they exhibit notably low performance on real-world datasets, highlighting the importance of real-world evaluation. Furthermore, we find that the attackers evade the detectors mainly in three ways: (1) directly attacking the model pipelines, (2) mimicking benign logos, and (3) employing relatively simple strategies such as eliminating logos from screenshots. To statistically assess the resilience and robustness of existing models against adversarial attacks, we categorize the strategies attackers employ into visible and perturbation-based manipulations and apply them to website logos. We then evaluate the models' robustness using these adversarial samples. Our findings reveal potential vulnerabilities in several models, emphasizing the need for more robust visual similarity techniques capable of withstanding sophisticated evasion attempts. We provide actionable insights for enhancing the security of phishing defense systems, encouraging proactive actions.

1 Introduction

Phishing attacks threaten Internet users' security through deceptive websites that mimic legitimate ones [25, 69]. Attackers replicate authentic sites of financial services or social media (*e.g.*, PayPal, Facebook), copying visual elements (*e.g.*, logos and layouts) to trick users into revealing sensitive credentials. In the ongoing battle against phishing attacks, anti-phishing systems employ multiple detection strategies. These defensive measures examine URLs [32, 34, 64], HTML structure [21, 49, 52], and visual elements [1, 3, 19, 41, 43–45] to identify fraudulent websites. The visual components (*e.g.*, logos and layouts) of websites have proven particularly critical in the phishing landscape, as attackers primarily rely on visual deception to establish credibility with potential victims. In response, visual similarity-based detection models have become an essential component of modern anti-phishing defenses, using deep learning techniques to identify fraudulent sites that closely resemble well-known target brands.



(a) Original Login Form of facebook.com (b) Adversarial Manipulation of Logo Text (Upper Case and Font)

Figure 1: **Examples of Original Login Form and Adversarial Manipulation.** An attacker changes the textual logo ('facebook') to its upper case ('FACEBOOK') and its font. The (b) example is found in our real-world phishing dataset.

Prior works [22, 43, 53] analyzed the robustness of phishing detectors. Particularly, Hao *et al.* [22] evaluated the robustness of detection models against their perturbation attacks where logo images are perturbed while preserving their semantic meaning. They found potential weaknesses in existing detection models. However, their work explored only limited perturbation techniques. Moreover, prior works evaluated their models without considering different conditions (*e.g.*datasets). Therefore, there are three major limitations: the lack of (1) systematic evaluations assessing the effectiveness and robustness of multiple detectors under consistent, fair, and large-scale real-world conditions, (2) in-depth analyses of influential factors in adversarial attacks (*e.g.*, Figure 1), and their impact on detection failures, and (3) efforts to identify specific weaknesses associated with each influential factor. These limitations impede the development of more actionable and concrete recommendations for enhancing these approaches.

Our work addresses these gaps in visual similarity-based phishing detection through comprehensive evaluations of prominent models using large-scale real-world datasets. By analyzing factors influencing adversarial attack outcomes, including image manipulation, layout changes, and color alterations, we systematically identify and categorize model weaknesses for each influential factor. This approach yields actionable recommendations for improving detection methods, guided by two key research questions: **RQ1:** Do visual similarity-based anti-phishing mechanisms maintain their *effectiveness* and *robustness* against real-world phishing attacks under the same experimental settings? **RQ2:** Are visual similarity-based anti-phishing mechanisms sufficiently resilient against adversarial strategies that manipulate visual components to evade detection?

In response to RQ1, we conduct a comprehensive performance evaluation of popular models on a large-scale dataset comprising 451k real-world phishing websites, 4,190 sampled phishing websites, and 2,500 benign samples (of Tranco Top 1000 websites (https://tranco-list.eu/)), to dig out the potential factors that are influential to the performance in phishing detection. While `PhishZoo` [3] initially appears promising with high detection accuracy (78.25%), our deeper analysis reveals significant limitations, including an elevated false positive rate (93.2%) and poor brand identification capabilities (12.78%). This indicates that these severe deficiencies may render the model impractical for real-world deployment.

We find that other models exhibit significantly lower performance compared to their original reported results on curated datasets. This discrepancy can be attributed to multiple factors, such as model structures and dataset attributes, highlighting the importance of evaluating models on real-world data to assess their actual performance. Furthermore, our study reveals that static brand reference lists used for brand-domain matching in `PhishIntention` [43] and `Phishpedia` [41] can be limited in real-world scenarios where websites regularly update their layouts and rebrand their logos. We also identify that attackers may craft phishing websites to directly attack the model pipeline, mimic legitimate websites, and use relatively simple strategies based on our analysis of failed examples. For example, simply eliminating logos will lead to the failures of logo-based methods because they can not recognize the brands of phishing webpages to verify the brand and domain.

To address RQ2, we manually analyze 6,000 detection failures to quantify key strategies attackers might employ. We then test these strategies using data from 110 popular benign websites, applying various visible and adversarial perturbations to visual components, particularly logos, to evaluate model resilience. Our findings reveal that both simple and adversarial manipulations can significantly undermine logo-based detection methods. These adversarial attacks are transferable across detection models. Although screenshot-based methods maintain stable detection, they struggle with accurately identifying brands when logos are altered. This evaluation offers crucial insights for developing more resilient models against adversarial attacks and evasion tactics.

The following summarizes our contributions.

- We conduct the first comprehensive study using a large-scale dataset of over 451k real-world phishing websites to fairly evaluate seven visual similarity-based anti-phishing systems by ensuring systems know the same brand knowledge. Our findings suggest that these systems are less effective in real-world scenarios, indicating significant performance degradation (20.7%), compared to their results on curated datasets.
- We also find three ways attackers usually employ to bypass detectors: (1) exploiting weaknesses of models' pipelines (*e.g.*, removing login forms), (2) mimicking benign logos and screenshots in the feature space, and (3) relatively simple strategies (*e.g.*, changing colors of logos).
- For robustness evaluation, we show critical limitations in visual similarity-based phishing detection models against adversarial samples.
- Based on our findings, we recommend several strategies to improve the effectiveness and robustness of visual similarity-based anti-phishing mechanisms. These include integrating text recognition with visual analysis and using preprocessing techniques such as scaling and denoising to minimize the impacts of adversarial perturbations.
- We publicly share our collected real-world phishing dataset, our manipulated dataset, code, and re-trained models at our website https://moa-lab.net/evaluation-visual-similarity-based-phishing-detection-models/.

2 Background

Phishing. Phishing is a type of social engineering attack in which attackers try to trick victims into disclosing sensitive information (*e.g.*, credentials). A phishing campaign involves fraudulent websites that mimic the appearance of legitimate websites. Victims are lured into disclosing their sensitive information to the attackers. Typically, such stolen information could be misused for further fraud or crimes.

Visual Similarity-based Phishing Detection Systems. URL-based phishing detection systems primarily rely on blocklist-based defense mechanisms (*e.g.*, Google Safe Browsing [58]) or machine learning models (*e.g.*, [33, 38, 70]) to prevent users from accessing malicious websites. However, relying solely on URLs is insufficient, as they provide limited information about a website's content, structure, or visual appearance, which are crucial for accurate phishing detection. To address these limitations, research has focused on analyzing visual components of phishing websites, such as screenshots and target brand logos. Early approaches used Earth Mover's Distance [19, 24], and SIFT [3] for

image matching, and assessments of block, layout, and style similarities [45]. Recent deep learning advancements have introduced more sophisticated methods. `VisualPhishNet` [1] uses triplet CNNs for learning visual similarities between webpage screenshots, while `Phishpedia` and `PhishIntention` combine Faster-RCNN [55] for logo recognition with a Siamese architecture for similarity comparison. `DynaPhish` [44] utilizes Google search to identify targeted brands and dynamically expand the reference lists.

Adversarial Visual Component Manipulation Attacks. To evade visual similarity-based phishing detectors, attackers manipulate visual components (*e.g.*, logos) [5, 35, 74]. Particularly, Giovanni *et al.* [5] found that phishers also bypass detectors by simply altering company name styles and stretching logos. Moreover, Ying *et al.* and Hao *et al.* [74] applied perturbations to visual components, and user study results demonstrated that these adversarial phishing techniques pose threats to both users and machine learning-based phishing website detectors. Recently, Lee *et al.* [35] developed an adversarial learning framework using imperceptible perturbation vectors based on a trained Vision Transformer (ViT) [16] and a Swin Transformer (Swin) [46]. Hao *et al.* [22] attacked the logos by changing fonts and generating adversarial logos through diffusion.

3 Problem Statement

Little effort has been made to systematically evaluate existing visual similarity-based defense models in real-world settings. Prior research [5, 35] predominantly focus on presenting novel attack models rather than systematically identifying and analyzing new inherent vulnerabilities of the models. Meanwhile, prior evaluation studies [1, 3, 41, 43] do not consider quantifying each influential factor.

To bridge this gap, we conduct a comprehensive evaluation of the effectiveness of visual similarity-based anti-phishing models using a large-scale real-world phishing dataset comprising 451k websites, 4,190 sampled phishing websites, and 2,500 benign samples (of the Tranco Top 1000 websites). Then, we examine how attackers manipulate visual elements and test model resilience against systematically modified logo images. Our study aims to (1) assess the effectiveness of current visual similarity-based models against real-world phishing attacks; (2) identify the root causes of the models' failures in classifying phishing websites; and (3) investigate new phishing tactics that manipulate visual components like logo images to circumvent existing visual similarity-based models.

4 Evaluation Design

4.1 Overview of Our Evaluation Methodology

We illustrate the overview of experiments in Figure 2. Our methodology includes the following key steps. *First*, we develop a web crawler that collects the screenshots and client-side resources of real-world phishing websites using phishing URLs reported by the Anti-Phishing Working Group (APWG) `eCX` [6] (❶). Note that APWG `eCX` shares real-time phishing threat intelligence (*e.g.*, phishing URLs). Then, we refine the collected phishing dataset by removing unnecessary data (*e.g.*, error pages) via clustering screenshots, as described in Section 4.2. Moreover, for the false positive evaluation, we also collect benign website samples based on the Tranco Top 1000 websites.

Table 1: **Training and Testing Reference List Dataset.** Training datasets are used to re-train the models.

	Definition	Dataset Source	Target Model	# Brand	# Image
$\boldsymbol{R}_{base}$	Baseline Ref.	PhishIntention [43]	**L-based**	277 (**B**)	3,064 (**B**)
		PhishIntention [43]	**S-based**	277 (**B**)	9,530 (**B**)
		VisualPhishNet [1]	**S-based**	155 (**B**) 155 (**P**)	9,363 (**B**) 1,193 (**P**)
$\boldsymbol{R}_{ext}$	Extended Ref.	Extended Logo	**L-based**	277 (**B**)	3,167 (**B**)
		Extended Screenshot	**S-based**	277 (**B**) 213 (**P**)	9,633 (**B**) 1,179 (**P**)

*__L-based__ = Logo-based models; __S-based__ = Screenshot-based models; __B__ = Benign; __P__ = Phishing.

(a) **Training Dataset**: Phishing Target Brand Reference Lists.

	Type	# Sample	# Domain	# Brand	# Cluster
$\boldsymbol{D}_{learn}$	Only-Learned-Brand Dataset	312,355	104,813	110	2,797
$\boldsymbol{D}_{all}$	All-Brand Dataset	451,514	163,864	270	4,190
$\boldsymbol{D}_{sample}$	Sampled Dataset	4,190	3,455	270	4,190
$\boldsymbol{D}_{benign}$	Benign Dataset	2,500	100	—	—

(b) **Testing Dataset**: Collected Phishing and Benign Websites.

Second, we standardize brand knowledge of the models using the same phishing target brand reference datasets (❷) to ensure fairness, which is $\boldsymbol{R}_{base}$, the baseline phishing target brand reference dataset from public sources. Considering the evolution of brands (*e.g.*, rebranding logos or website layouts), we further expand it and yield $\boldsymbol{R}_{ext}$ our extended reference dataset (❸). Table 1a shows the statistics of these datasets.

Third, we carefully select seven popular visual similarity-based phishing defense models considering different factors, as detailed in Section 4.3. Then, these datasets ($\boldsymbol{R}_{base}$ and $\boldsymbol{R}_{ext}$) are used to re-train the models (❹). Further details can be found in Section 4.4. *Fourth*, we evaluate the effectiveness of the seven models using our collected real-world datasets (❺): the "Only-Learned-Brand" dataset ($\boldsymbol{D}_{learn}$), the "All-Brand" dataset ($\boldsymbol{D}_{all}$), the "Sampled Dataset" ($\boldsymbol{D}_{sample}$), and the "Benign Dataset" ($\boldsymbol{D}_{benign}$) as shown in Table 1b.

Regarding the failed samples for the models (*e.g.*, changing the text logo to upper case, as shown in Figure 1(b)), we attempt to understand why the models fail to classify specific phishing attacks. We address RQ2 by designing another experiment where we manipulate visual components (*e.g.*, logo images) of phishing websites in two ways (❻): (1) visible manipulation techniques (*i.e.*, changing logo color or location) and (2) perturbation-based adversarial manipulation techniques (*i.e.*, white-box attack). Table 14 illustrates the examples of the manipulations. Then, we evaluate the robustness of models using the manipulated dataset and quantify their failures (❼), discussed in Section 4.5.

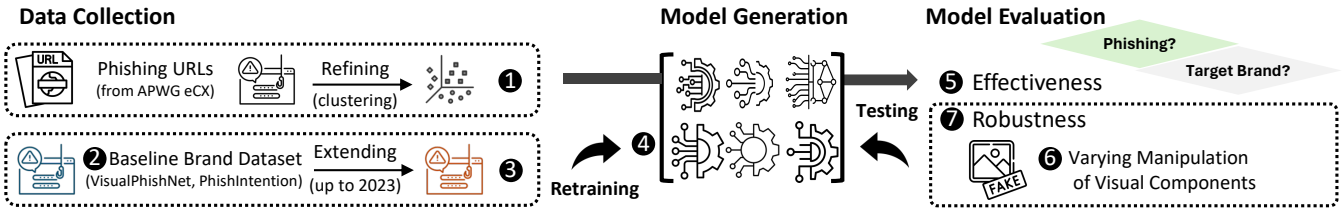


Figure 2: **Overview of Our Experiment.** We collect real-world phishing and benign websites (❶). We prepare two reference datasets (❷ and ❸). We then carefully select seven popular visual similarity-based anti-phishing models and re-train them using the prepared datasets (❹). The effectiveness and robustness of these models are systematically evaluated (❺, ❻, and ❼).

4.2 Real-world Phishing Dataset Collection

Web Crawler Design. APWG `eCX` [6] is one of the most trusted, largest repositories for real-world phishing attacks as it aggregates reports from security vendors, financial institutions, and ISPs. It is widely used to analyze and better understand phishing ecosystems [31, 40, 50, 51, 76]. As APWG `eCX` provides only phishing URLs, we newly design a web crawler that regularly (every minute) gathers (1) client-side resources (*e.g.*, logos, HTML, etc.) of phishing websites and captures (2) the screenshots. The web crawler is implemented through `Google Selenium Chrome WebDriver` [60] to simulate real user interactions with phishing websites, fully loading and rendering all client-side resources on the webpages. Additionally, `Selenium Chrome WebDriver` may assist in evading basic anti-bot techniques employed by phishing websites [4, 39].

Refining Dataset. APWG `eCX` provides a total of 15,747,193 (15.7M) real-world phishing URLs from July 2021 to July 2023 (25 months). Our crawler successfully accesses 6,118,654 (6.1M) phishing websites. 61.1% of inaccessible websites are due to server shutdowns or network errors such as DNS resolving errors. Among 6.1M samples, we exclude internal error web pages (*i.e.*, page not found) and improve label accuracy by clustering screenshots based on `Fastdup` [18], an open-source tool that is effective in identifying duplicates, outliers, and clusters of related images by calculating the edge distances inside the graph component. Specifically, we select 6,885 clusters with more than 20 screenshots, as these account for over 90% of the total 6.1M screenshots. Two security researchers independently conduct manual inspections of the clusters. They each select three representative samples from each cluster and label them. The researchers then compare their results, discuss any discrepancies, and combine the clusters. This process is iteratively repeated until a consensus on all labels is reached. The filtered dataset contains 2,160,933 samples, representing 270 brands and 4,190 clusters.

Final Phishing Dataset for Evaluation. Among the filtered dataset, a small percentage (20%) of clusters (*i.e.*, merged brands) hold the majority (80%) of total screenshots, which makes the evaluation process time-consuming and susceptible to bias. Therefore, we randomly select 1,000 samples for each cluster (if lower than 1,000, then all of them are selected) to ensure fairness. This process results in a total of 451,514 samples with 270 brands and 4,190 clusters, denoted as $\boldsymbol{D}_{all}$. Furthermore, $\boldsymbol{D}_{all}$ includes some brands that are not present in the training dataset ($\boldsymbol{R}_{base}$ and $\boldsymbol{R}_{ext}$), meaning the brands are not learned by models. We identify 110 common brands between $\boldsymbol{D}_{base}$ and $\boldsymbol{D}_{all}$. This dataset, called $\boldsymbol{D}_{learn}$, includes 312K samples with 110 learned brands. Additionally, we sample 1 example for each cluster and construct $\boldsymbol{D}_{sample}$ to test `DynaPhish` due to computational intensity and extensive Google Search API costs. In summary, $\boldsymbol{D}_{learn}$, $\boldsymbol{D}_{all}$, $\boldsymbol{D}_{sample}$, and $\boldsymbol{D}_{benign}$ are used to better understand the models' performance (*i.e.*, effectiveness and robustness) in real-world scenarios and to examine the impact of data on unlearned brands.

Benign Website Dataset for False Positive Evaluation. To evaluate false positive rates of phishing detection models, we assemble a dataset of legitimate websites. We randomly select 100 domains from the top 1,000 websites in the Tranco 1M ranking. For each domain, we collect monthly snapshots between July 2021 and July 2023 using the Internet Archive's Wayback Machine (`https://archive.org/`), capturing URLs, screenshots, and HTML content. This process yields 2,500 benign samples (100 domains $\times$ 25 months). This dataset includes 14 domains comprising 350 samples, which are associated with 12 target brands present in the training data ($\boldsymbol{D}_{learn}$).

4.3 Model Selection for Evaluation

We carefully select representative models of visual similarity-based anti-phishing techniques for comprehensive evaluations. *First*, we initially search some keywords (*i.e.*, 'anti-phishing,' 'phishing detection,' and 'visual-based similarity') at the top-tier security, computer vision, and machine learning conferences to identify model candidates. Model candidates are summarized in Table 10 and Appendix A.1.

From these candidates, we choose models with publicly available code to ensure fidelity to the original papers. Furthermore, considering the utilized information and pipeline structures, we select three recent popular logo-based phishing detection models, `DynaPhish` [44], `PhishIntention` [43], and `Phishpedia` [41]. They use URLs, screenshots, and HTML

as input, employing cropped logos to match the logos in the reference list for brand detection and identification. Additionally, `PhishZoo` [3] uses the same inputs but matches between screenshots and logos. For a broader comparison, we select `Involution` [66], a model not specifically tailored for phishing detection. We are also interested in whole image comparison and thus select `VisualPhishNet` [1], which detects phishing using screenshots, and `EMD` (Earth Mover's Distance) [19] which is a static model for screenshot-based phishing detection. Finally, detailed explanations of the models, including their descriptions and inputs, are provided in Appendix A.1 and Appendix A.2.

4.4 Re-training Models & Evaluation Plan

We aim to rigorously evaluate the effectiveness of the carefully selected seven visual similarity-based anti-phishing models with our extensive dataset of real-world phishing websites. Initially, training under varying conditions and with diverse reference lists can significantly impact the evaluation outcomes. Additionally, based on our evaluation, the presence of either outdated or new visual elements, such as rebranded logo images or login forms, can profoundly affect model performance, as these elements might not have been adequately captured during initial training. For example, updates (*i.e.*, refreshes) to Facebook's login form, user interface, or icons could potentially adversely impact the model's performance. To ensure a more equitable and cautious approach in evaluations and model performance comparisons, we *re-train* the models by incorporating them with the same reference knowledge of brands, taking these factors into account.

Two Variants of Re-trained Models. Our objective in developing two variants of re-trained models is to assess the impact of brand variations (*e.g.*, refreshed or outdated logos) because our initial evaluation of the all-brands dataset with the base reference list demonstrates the limitations of the outdated reference list (*i.e.*, refreshed or outdated logos are not included). The distinction between the two variants lies in the reference lists used: (1) the baseline phishing target brand reference dataset ($\boldsymbol{R}_{base}$) and (2) our expanded reference dataset ($\boldsymbol{R}_{ext}$), as detailed in Table 1a. $\boldsymbol{R}_{base}$ is the same brand list as `PhishIntention` (L-based) for logo-based anti-phishing models and `PhishIntention` (S-based) for screenshot-based models. $\boldsymbol{R}_{ext}$ is obtained by expanding the $\boldsymbol{R}_{base}$ by adding a newly updated logo and screenshot variance from Archive between 2016 and 2023. Note that we train `VisualPhishNet` only on the $\boldsymbol{R}_{ext}$ screenshot dataset, as it requires both benign and phishing data during the training phase and ensures the brand knowledge consistent. During evaluation, we integrate the trained model with `PhishIntention` (S-based) and $\boldsymbol{R}_{ext}$ screenshot dataset as the reference lists for baseline and extended results, respectively.

4.5 Manipulating Visual Component (Logo)

Through our evaluation experiment, we analyze the manipulation tactics employed by phishing attackers. We find that there are four primary components typically exploited by adversaries in phishing attacks, including logos, pop-ups, login forms, and others, as presented in Table 12 of Appendix A.4. Upon randomly selecting images from the failure results, we discern that logos are prevalent targets used by adversaries to circumvent detection mechanisms. Furthermore, logos serve as the indicators for both users and detection mechanisms to recognize the websites. Consequently, this study focuses primarily on the logo component.

4.5.1 Manipulation Methods

Phishing attackers not only aim to mimic legitimate target brand websites to deceive potential victims closely but also aim to evade detection by anti-phishing systems, particularly those based on visual similarity. To achieve this, they have developed various adversarial visual component manipulation strategies. We broadly categorize such strategies into (1) visible manipulation techniques and (2) perturbation-based adversarial manipulation techniques.

Visible Manipulation Methods. The visible manipulation techniques involve substantial, noticeable changes to the original visual appearance, such as altering the image color [30], text [72], and UI design patterns [62]. For instance, logos (*e.g.*, changing the Facebook logo's font and converting the letters to uppercase) are manipulated, as illustrated in Figure 1. Based on the failure samples discussed in Section 5, we identify 13 types of manipulations used by adversaries in real datasets and choose SRNet [71] as an additional deep learning-based method in this category. For descriptions of SRNet, please refer to Appendix A.5. Subsequently, we craft logos corresponding to these manipulations and attach them to the original screenshots. Specifically, we use 'remove.bg' (`https://www.remove.bg/`) to eliminate the background of logos. If users are not looking at the logos carefully, they readily overlook it and can be readily lured. The manipulations and crafted logos are shown in Table 14. Note that we do not combine more than two visible manipulation ways.

Perturbation-based Manipulation Methods. Perturbation-based adversarial manipulation techniques introduce subtle manipulations [11, 20, 47, 63] that are difficult for humans to detect visually. We introduce perturbations to the logos using popular white-box and black-box attack techniques. The perturbed logos are then returned to their original positions on the screenshots, ensuring a seamless integration into the visual context. For descriptions of selected models and perturbated logos, please refer to Appendix A.5 and Figure 4.

4.5.2 Evaluation Plan for Manipulation

All models used to evaluate robustness are trained on the 'Extended Ref.' ($\boldsymbol{R}_{ext}$) or 'Baseline Ref.' ($\boldsymbol{R}_{base}$) datasets along with the original needed datasets. We focus on the learned 110 brands ($\boldsymbol{D}_{learn}$) to obtain more accurate results. Furthermore, to investigate the impact of different factors (URLs, Logo, and HTML), we conduct an ablation study for four models that highly depend on the three factors. Specifically, we manually collect the URLs, HTML, and screenshots of the login page or main page of 110 brands' websites. Then, we use `typosquatter` (https://github.com/typosquatter/ail-typosquatting) to generate various domain typos to replace their original domain within URLs. Finally, the total dataset contains 6,569 visible manipulated screenshots, 544 perturbated screenshots, 110 benign URLs, and 1,321 squatted URLs for 110 brands. These URLs are then paired with corresponding altered images to curate the phishing testing dataset.

5 Evaluation of Model Performance on Real-world Phishing Datasets

We first assess the effectiveness of seven models using our real-world phishing and benign datasets in phishing detection and computational costs using FLOPs and parameters (Section 5.1), false positive rates (Section 5.2), and phishing brand identification (Section 5.3). Additionally, we further analyze the reasons for the failures of the models (Section 5.4). Finally, we conduct ablation studies on logo, URL, and HTML features to understand their contributions to the detection performance (Section 5.5).

Settings. $\boldsymbol{D}_{learn}$, $\boldsymbol{D}_{all}$, $\boldsymbol{D}_{sample}$, and $\boldsymbol{D}_{benign}$ in Table 1b are leveraged to evaluate the performance of the models. We define a *true positive* as correctly detecting phishing and a *false positive* as incorrectly classifying a benign website as phishing. For phishing detection, let N_p be the number of phishing testing samples, N_{tp} (*tp* stands for true positive) be the number of correctly classified phishing samples, and I_{tp} be the number of correctly identified target brands. For benign samples, let N_b be the total number, N_{fp} (*fp* stands for false positive) be the number falsely classified as phishing, and I_{fp} be the number with incorrectly identified target brands.

Using these metrics, we calculate six rates: (1) the true positive rate (N_{tp}/N_p), measuring phishing detection accuracy; (2) the phishing identification rate (I_{tp}/N_p) and (3) identified phishing accuracy (I_{tp}/N_{tp}) for brand identification; (4) the false positive rate (N_{fp}/N_b), measuring benign misclassification; (5) the false identification rate (I_{fp}/N_{fp}) and (6) the overall false brand rate (I_{fp}/N_b) for incorrect brand identification.

Thresholds. Threshold values are obtained by prior work [42] and our further check with their datasets. Specifically, we set the thresholds as 0.83, 0.83, 0.83, 40, 1, 0.94, and 0.7 for `DynaPhish`, `PhishIntenion`, `Phishpedia`, `PhishZoo`, `VisualPhishNet`, `EMD`, and `Involution`, respectively, to identify potential phishing instances effectively.

5.1 Result: Detection Effectiveness

Phishing detection refers to the capability to correctly classify websites as phishing or legitimate. Table 2 shows the phishing detection and brand identification results on large-scale real-world datasets (D_{learn} and D_{all}). Table 3 provides the phishing detection and brand identification results of D_{sample}. Table 4 describes the false positive and cost results.

General Performance. All seven models demonstrate lower performance in phishing detection, compared to their originally reported results. Specifically, when trained on $\boldsymbol{R}_{base}$, six models in Table 2 failed to detect 38.62% of the 312,355 phishing samples from their learned target brands ($\boldsymbol{R}_{learn}$). Even with expanded training on $\boldsymbol{R}_{ext}$, these models still miss 33.8% of phishing websites.

Logo-based phishing detection models experience significant performance degradation on comprehensive datasets. True positive rates of `PhishIntention`, `Phishpedia`, and `Involution` drop by 13–19% when tested on the whole-brand dataset (D_{all}, R_{ext}). Although `PhishZoo` reaches 77.22% accuracy on limited datasets (D_{learn}) and 78.25% on D_{all}, the false positive reaches 93.92% on D_{benign}. Their primary weakness stems from poor resilience to logo variations and heavy reliance on static reference lists for similarity matching, making them vulnerable to evasion through logo modifications not present in their reference lists.

> **Takeaway 1:** Reference list-based models can introduce weaknesses. Logos or screenshots not included in the reference list but known to users may mislead the detection models. This highlights the necessity of expanding and continuously updating the reference lists and detection models.

Learned Vs. Unlearned Testing Dataset. When deployed in real-world environments, phishing detection models are highly likely to encounter unlearned brands. To further investigate the models' readiness for real-world deployment with new, unknown phishing websites, we compare the results between $\boldsymbol{D}_{learn}$ (containing only learned brands) and $\boldsymbol{D}_{all}$ (also including unlearned, new brands).

The results, detailed in Table 2, show a decline in detection performances in more challenging scenarios ($\boldsymbol{D}_{all}$). Particularly, `PhishIntention`, `Phishpedia`, and `Involution` (logo-based models) decrease in the detection rate from 66.22% to 52.68% (13.54%↓), from 87.97% to 70.47% (17.5%↓), and from 84.77% to 66.67% (18.1%↓), respectively with R_{ext}. The three models rely on identifying and comparing logo similarities with their target brand reference list. If either these models fail to recognize the logo or the brand does not appear in the reference list, the similarity score will be lower than the threshold, leading to detection failures.

`PhishZoo` shows consistent but unreliable performance across datasets, with low identification rates (9.86% on

Table 2: **Phishing Detection Results** on 312,355 ($\boldsymbol{D}_{learn}$) and 451,514 ($\boldsymbol{D}_{all}$) testing samples from APWG. **Phishing Brand Identification Results** on 312,355 ($\boldsymbol{D}_{learn}$). The bold numbers denote the better detection or identification rate.

Model	Phishing Detection				Phishing Brand Identification (with $\boldsymbol{D}_{learn}$)					
	Only-Learned Brands (312,355) $\boldsymbol{D}_{learn}$		All Brands (451,514) $\boldsymbol{D}_{all}$		Baseline Ref. ($\boldsymbol{R}_{base}$)			Ext. Ref. ($\boldsymbol{R}_{ext}$)		
	Baseline Ref. ($\boldsymbol{R}_{base}$)	Ext. Ref. ($\boldsymbol{R}_{ext}$)	Baseline Ref. ($\boldsymbol{R}_{base}$)	Ext. Ref. ($\boldsymbol{R}_{ext}$)	$\boldsymbol{I}_{tp}$[1]	$\boldsymbol{I}_{tp}/\boldsymbol{N_p}$[2]	$\boldsymbol{I}_{tp}/\boldsymbol{N}_{tp}$[3]	$\boldsymbol{I}_{tp}$[1]	$\boldsymbol{I}_{tp}/\boldsymbol{N_p}$[2]	$\boldsymbol{I}_{tp}/\boldsymbol{N}_{tp}$[3]
PhishIntention	204,880 (65.59%)	206,846 (66.22%)	235,838 (52.23%)	237,861 (52.68%)	200,134	64.07%	97.68%	202,123	64.71%	97.72%
Phishpedia	232,572 (74.46%)	**274,779 (87.97%)**	275,292 (60.97%)	318,196 (70.47%)	222,860	71.34%	95.82%	**265,627**	**85.04%**	96.67%
Involution	**253,965 (81.31%)**	264,782 (84.77%)	289,058 (64.02%)	301,035 (66.67%)	**253,090**	**81.03%**	**99.66%**	263,835	84.47%	**99.64%**
PhishZoo	241,206 (77.22%)	269,748 (86.36%)	**353,292 (78.25%)**	**389,585 (86.28%)**	30,829	9.86%	12.78%	89,724	28.73%	33.26%
VisualPhishNet	122,106 (39.09%)	126,762 (40.58%)	181,177 (40.13%)	186,606 (41.33%)	81,119	25.97%	66.43%	83,697	26.80%	66.03%
EMD	95,632 (30.62%)	97,880 (31.34%)	133,241 (29.51%)	136,697 (30.28%)	22,478	7.20%	23.50%	22,426	7.18%	22.91%

[1] I_{tp} = The number of phishing samples brands correctly identified; [2] I_{tp}/N_p = The phishing target brand identification rate out of the total phishing testing samples;
[3] I_{tp}/N_{tp} = The phishing target brand identification rate out of the only samples detected as phishing by each model.

Table 3: **Phishing Detection and Identification Results on $\boldsymbol{D}_{sample}(4,190)$ from APWG trained with $\boldsymbol{R}_{base}$.**

Model	Detection	Identification $\boldsymbol{I}_{tp}$ ($\boldsymbol{I}_{tp}/\boldsymbol{N}_{tp}$)
PhishIntention	2,056 (49.07%)	**2,027 (98.56%)**
Phishpedia	2,395 (57.16%)	2,212 (92.36%)
Involution	2,538 (60.57%)	2,470 (97.32%)
PhishZoo	**3,190 (76.13%)**	306 (9.59%)
VisualPhishNet	1,418 (33.84%)	773 (54.51%)
EMD	1,150 (27.45%)	236 (20.42%)
DynaPhish	923 (22.03%)	904 (97.94%)

D_{learn} with R_{base}, 9.59% on D_{sample}) and high false positives (93.92% on D_{benign}). This indicates misclassification rather than accurate detection, stemming from its HTML/URL keyword selection and SIFT feature comparison methodology. Screenshot-based models (`EMD`, `VisualPhishNet`) demonstrate better resilience to unlearned brands compared to logo-based approaches. `EMD`'s detection rate slightly decreases from 31.34% to 30.28% (1.06%↓), while `VisualPhishNet` shows a marginal increase from 40.58% to 41.33% (0.75%↑) on R_{ext}. However, these models generally achieve lower detection rates than logo-based detectors (`PhishIntention`, `Phishpedia`, `Involution`), highlighting a trade-off between resilience to unlearned brands and overall detection performance.

Takeaway 2: Logo-based models heavily rely on their pre-established brand reference lists, leading to degraded detection performance when encountering unlearned brands. In contrast, screenshot-based models demonstrate better resilience to unlearned brands, though they generally achieve lower detection rates than logo-based models.

Baseline Vs. Extended Reference List. Recall that the Extended Reference List Dataset ($\boldsymbol{R}_{ext}$) is curated by manually adding more logo variance and screenshots of their learned target brands to the baseline dataset ($\boldsymbol{R}_{base}$). Typically, the new logos are slightly changed from their prior logos.

We observe a significant performance increase in both `Phishpedia` and `PhishZoo` when being tested on $\boldsymbol{D}_{learn}$. Specifically, the phishing detection accuracy of `Phishpedia` and `PhishZoo` increased from 74.46% to 87.97% (13.51%↑) and from 77.22% to 86.36% (9.14%↑), respectively. The performance gain for `Phishpedia` is attributed to the recent logo updates in the dataset, highlighting the importance of comprehensive and regularly updated logo collections in phishing detection model design. In contrast, the apparent improvement in `PhishZoo` does not truly reflect its effectiveness in phishing detection. Considering its low identification results, it appears that logos among reference lists mislead `PhishZoo` to incorrectly recognize brands of phishing websites. Moreover, `VisualPhishNet` and `EMD` demonstrate resilience to changes in their reference lists, as newly added screenshots often share visual styles with existing samples. However, these models face a fundamental limitation: the challenge of capturing the complete diversity of webpage layouts and designs, which constrains their overall performance.

These findings reveal a potential weakness in reference-based models that attackers could exploit using newly updated logos or various designs not included in the reference list. While increasing reference data (regularly adding new logos) improves detection performance, it also prolongs computing time, presenting a crucial trade-off. This trade-off underscores the need for a strategic model design, where balancing detection efficacy and computational efficiency is vital.

Takeaway 3: Updating the reference list dataset to include varied logos, as well as retaining outdated logos, significantly improves the performance of logo-based models. However, expanding the reference dataset also increases computation time, necessitating a balance between detection accuracy and efficiency in model design.

Influence of Model Architecture. `PhishIntention`, `Phishpedia`, and `DynaPhish` share a common architectural foundation for phishing detection yet exhibit distinct performance characteristics. `Phishpedia` demonstrates a better detection rate (57.16%) on the D_{sample} compared to `DynaPhish` (22.03%) and `PhishIntention` (49.07%). This performance divergence stems from the enhanced verification mechanisms implemented in `PhishIntention`, specifically their analy-

Table 4: **False Positive Result and Performance Comparison.** Bold text indicates the best performance for each metric.

Model	N_{fp}/N_b	InferTime*	FLOPs	Parameter
DynaPhish	**0**	13.36s**	215.66G	88.92M
PhishIntention	**0**	0.24s	215.66G	88.92M
Phishpedia	406 (16.24%)	0.34s	212.35G	65.40M
Involution	99 (3.96%)	**0.1s**	212.67G	53.04M
PhishZoo***	2,348 (93.92%)	16 mins	—	—
VisualPhishNet	338 (13.52%)	2.27s	**92.49G**	**21.27M**
EMD***	659 (26.36%)	23.54s	—	—

*__InferTime__: Inference time taken for each sample.
**This time includes online search via Google Search APIs.
***Note that FLOPs and parameters are unavailable for PhishZoo and EMD.

sis of credential forms in HTML and screenshots. Notably, `DynaPhish` achieves the lowest detection rate (22.03%) compared to architecturally similar models (`PhishIntention` and `Phishpedia`). Our analysis reveals that 43.94% of samples are flagged due to forbidden words appearing in the searched page titles. This filtering mechanism significantly impairs the model's overall detection performance.

`PhishZoo`, leveraging extracted keywords from URLs and HTML sources to mitigate false positives and employing the SIFT feature to calculate similarity in the target list, has reported promising results in phishing detection tasks. However, the performance on brand identification reveals that the high accuracy initially indicated may be an overestimation of its true capabilities. We reveal that the keyword selection approach based on TF-IDF scoring does not accurately capture brand-specific keywords that are highly indicative of phishing attempts. Specifically, we identify some examples where common words like 'the' and 'in' influence classification decisions, exposing a limitation in the feature engineering process.

Furthermore, the results indicate that screenshot-based reference methods are more stable for brand changes in both testing and reference data compared to logo-based methods, though their overall performance is worse. `EMD` measures the distribution distance between testing samples and the reference list, while `VisualPhishNet` utilizes a triplet Convolutional Neural Network to compare two screenshots. Unlike logos, the wide variety of screenshots presents a significant challenge in covering the full range of variations in the reference target list. Additionally, screenshots not present in the target list but sharing similar designs or features exhibit a high degree of resemblance to those in the target list. This highlights a potential flaw in screenshot-based methods due to the vast diversity and similarity among web designs.

Takeaway 4: Inaccurate keyword extraction can degrade performance, while diverse screenshots and similar web designs present challenges to screenshot-based methods. Selecting an appropriate model structure is crucial to optimizing performance and mitigating these weaknesses.

Costs for Models. We evaluate model efficiency through inference time, FLOPs, and parameter size using a dataset of 2,500 benign samples (D_{benign}), with results presented in Table 4; detailed performance metrics for key components are provided in Table 11 and Appendix A.3.

`Involution` demonstrates the fastest inference time (0.1 seconds/sample), utilizing 212.67G FLOPs and 53.04M parameters. While `PhishIntention` and `DynaPhish` share identical architecture and computational requirements, `DynaPhish`'s additional Google Search verification process results in 56 times longer inference time.

Despite `Phishpedia`'s lower computational requirements compared to `PhishIntention`, its inference time is slower (0.34 seconds/sample) due to 97% of samples bypassing CRP detection. `PhishZoo` shows the longest inference time (16 minutes/sample) due to sequential keyword comparisons and SIFT feature extraction. Though `VisualPhishNet` uses fewer computational resources than `Phishpedia`, its full-screenshot analysis approach leads to 6.68 times longer inference times compared to `Phishpedia`'s logo-focused analysis. Similarly, `EMD`'s screenshot-based approach results in significantly longer processing times.

5.2 Result: False Positive Analysis

Even models achieving high true positive rates may prove impractical if they generate excessive false alarms (*i.e.*, incorrectly flagging benign websites as phishing). Such misclassifications can severely impact system efficacy in production environments, potentially undermining user trust and increasing operational overhead.

Our analysis in Table 4 reveals significant performance variations across models. `PhishZoo` demonstrates poor reliability with a 93.92% false positive rate due to screenshot-logo mismatches during brand identification. While `DynaPhish` and `PhishIntention` achieve 0% false positive rates, consistent with their original papers. Their superior performance is due to additional verification steps but at the cost of true positives, 22.03% and 49.07% respectively, as shown in Table 3. Their performance is notably influenced by dataset characteristics, where D_{sample} contains 2,155 samples that lack CRPs and 1,841 samples fall within forbidden domains for `DynaPhish`. `Phishpedia` shows moderate performance with a 16.24% false positive rate and 57.16% phishing detection accuracy. Its limited knowledge scope, covering only 14 domains across 12 brands in D_{benign}, leads to poor generalization and frequent misclassification of legitimate websites from unfamiliar brands, significantly impacting its real-world applicability.

Takeaway 5: Detection models face trade-offs between false positives and detection accuracy. Importantly, most models mistakenly recognize benign websites outside knowledge scope as existing brands, highlighting significant deployment challenges in open set recognition.

5.3 Result: Phishing Brand Identification

Phishing brand identification refers to identifying brands that phishing websites attempt to impersonate. Table 2 shows the results on the learned brand dataset $\boldsymbol{D}_{learn}$ and Table 3 contains the results on $\boldsymbol{D}_{sample}$.

General Performance. Our analysis reveals significant differences in performance between logo-based and screenshot-based approaches for brand identification. The logo-based models (`Phishpedia` and `Involution`) demonstrate superior performance (84%–88% detection rate while 96%–99.64% on for identification on D_{learn} with R_{ext}) across both tasks, highlighting the critical role that logo recognition plays in accurately identifying target brands and detecting phishing.

As shown in Table 2 and Table 3, with R_{base} reference dataset on D_{learn}, `PhishIntention`, `Phishpedia`, and `Involution` achieve identification rates of 97.68%, 95.82%, and 99.66% respectively. Similarly, on D_{sample}, they maintain strong performance with rates of 97.94% (`DynaPhish`), 98.56% (`PhishIntention`), 92.36% (`Phishpedia`), and 97.32% (`Involution`). This consistency across datasets indicates suitable reference logo coverage for recognized brands.

In contrast, screenshot-based models (`VisualPhishNet` and `EMD`) demonstrate inferior performance, with detection rates of 40.58% and 31.34%, and identification rates of 66.03% and 22.91% respectively for correctly detected samples on D_{learn} with R_{ext}. This underperformance stems from their broader analysis of webpage elements rather than focus logo detection, complicated by the challenge of maintaining current screenshot datasets amid dynamic webpage layouts. However, these approaches maintain consistent effectiveness with unfamiliar brands where logos are unavailable, offering a valuable advantage despite lower overall performance.

`PhishZoo` achieves detection rates of 76.13% on D_{sample} and 86.36% on D_{learn} with R_{ext}, but its brand identification rate is only 9.59% on D_{sample}. This disparity indicates misleading performance metrics, stemming from `PhishZoo`'s SIFT-based methodology that struggles with screenshot-to-logo matching in its brand database. In contrast, `VisualPhishNet` outperforms `EMD` in identification by employing triplet CNN to learn intra-brand similarities and inter-brand differences, while `EMD`'s effectiveness diminishes when brand screenshot distributions show high similarity.

Takeaway 6: Logo-based models currently offer the most reliable approach for standard phishing detection and brand identification, but they are susceptible to additional checking steps, used features, and logo components. Screenshot-based models struggle with web design diversity but may serve as a complementary solution for scenarios involving unknown or emerging brands.

Baseline Vs. Extended Reference List. The six evaluated models show significant brand identification failure rates: approximately 51% with Extended Reference dataset ($\boldsymbol{R}_{ext}$) and 57% with Baseline Reference dataset ($\boldsymbol{R}_{base}$). Logo-based methods demonstrate improved performance with $\boldsymbol{R}_{ext}$ versus $\boldsymbol{R}_{base}$, revealing generalization limitations. Specifically, when switching from $\boldsymbol{R}_{base}$ to $\boldsymbol{R}_{ext}$, `Phishpedia`'s identification rate increases from 71.34% to 85.04% (13.7%↑), `Involution` from 81.03% to 84.47% (3.44%↑), and `PhishZoo` from 9.86% to 28.73% (18.87%↑). This indicates that logo variations significantly enhance `Phishpedia` and `Involution`'s identification capabilities. Conversely, `PhishIntention`, `VisualPhishNet`, and `EMD` maintain stable performance across reference changes, attributed to CRP-based filtering and screenshot-based methods' inherent resilience to limited reference variations.

Table 5: **Failure Statistics of Different Evasion Strategies.**

	Strategy	PhishIntention	Phishpedia	PhishZoo	Involution
Similar	WrongLogoArea	68 (6.8%)	19 (1.9%)	16 (1.6%)	173 (17.3%)
	CorrectLogoArea	377 (37.7%)	245 (24.5%)	633 (63.3%)	67(6.7%)
	Total	445 (44.5%)	264 (26.4%)	649 (64.9%)	240 (24.0%)
Visible	Elimination	44 (4.4%)	88 (8.8%)	37 (3.7%)	48 (4.80%)
	BrokenImage	1 (0.1%)	1 (0.1%)	3 (0.3%)	1 (0.1%)
	ColorReplace	52 (5.2%)	58 (5.8%)	45 (4.5%)	48 (4.8%)
	LogoBackground	26 (2.6%)	157 (15.7%)	26 (2.6%)	7 (0.7%)
	ImageBackground	28 (2.8%)	0 (0.0%)	4 (0.4%)	71 (7.1%)
	Popup/Blurring	63 (6.3%)	36 (3.6%)	2 (0.2%)	37 (3.7%)
	Integration	63 (6.3%)	54 (5.4%)	48 (4.8%)	117 (11.7%)
	Re-position	43 (4.3%)	36 (3.6%)	12 (1.2%)	56 (5.6%)
	Outdated	154 (15.4%)	194 (19.4%)	8 (0.80%)	102 (10.2%)
	CaseConversion	9 (0.9%)	32 (3.2%)	15 (1.5%)	43 (4.3%)
	TextAsLogo	5 (0.5%)	14 (1.4%)	9 (0.9%)	3 (0.3%)
	ScalingOrResizing	8 (0.8%)	0 (0.0%)	18 (1.8%)	4 (0.4%)
	FontReplace	4 (0.4%)	9 (0.9%)	1 (0.1%)	8 (0.80%)
	Omission	7 (0.7%)	23 (2.3%)	38 (3.8%)	32 (3.20%)
	Shape	3 (0.3%)	16 (1.6%)	0 (0.0%)	18 (1.8%)
	ImageAddText	36 (3.6%)	13 (1.3%)	79 (7.9%)	153 (15.3%)
	LogoAddText	2 (0.2%)	4 (0.4%)	0 (0.0%)	7 (0.7%)
	Replacement	0 (0.0%)	1 (0.1%)	6 (0.6%)	5 (0.5%)
	Blocked	1 (0.1%)	0 (0.0%)	0 (0.0%)	0 (0.0%)
	Total	555 (55.5%)	736 (73.6%)	351 (35.1%)	760 (76.0%)
Total		1,000 (100%)	1,000 (100%)	1,000 (100%)	1,000 (100%)

Table 6: **Analysis of Detection Failures for Visually Similar Phishing Logos in 'CorrectLogoArea.'**

Model	T.*	#Sample	Feat. Sim.**	SSIM (< 0.7)	PSNR (> 4)
PhishIntention	0.83	28	27 (0.6–0.83)***	27	11
Phishpedia	0.83	206	190 (0.6–0.83)	206	205
PhishZoo	40	633	633 (0–40)	569	628
Involution	0.7	67	62 (0.6–0.7)	66	60

*: Threshold. **: Feature Similarity. ***: One sample is misclassified.

5.4 In-depth Analysis of Detection Failures

We find that real-world phishing attackers frequently modify four main visual components (logo, popup, login, and others) to evade phishing detection systems. These strategies and descriptions are summarized in Table 12 of Appendix A.4. To quantify the prevalence of evasion strategies, we conduct a systematic manual review of 6,000 phishing samples from D_{learn} that evaded detection, randomly selecting 1,000 failed samples from each model. We exclude `DynaPhish` due to its structural similarity to `PhishIntention`.

(a) Phishing Logo (b) LIME Analyzed Logo

Figure 3: Analysis of a Facebook phishing logo using LIME. Black means negative or no contributions.

In our examination, we observe varying degrees of visual modifications, from obvious alterations detectable by humans to subtle changes that are challenging to identify. To categorize these modifications, we establish two primary classification terms: 'similar' and 'visible.' For samples classified as 'similar,' the models successfully identify the correct logo placement designated as 'CorrectLogoArea,' while incorrect placements are termed 'WrongLogoArea.' The comprehensive results of our logo-based model analysis are presented in Table 5, with detailed performance matrices for 'CorrectLogoArea' documented in Table 6.

5.4.1 Logo-based Methods

Table 5 reveals that 44.5%, 26.4%, 64.9%, and 24.0% of failed samples appear similar logos to brand target lists for `PhishIntention`, `Phishpedia`, `PhishZoo`, and `Involution`, respectively. Among these, 6.8%, 1.9%, 1.6%, and 17.3% of samples fail to locate accurate logos ('WrongLogoArea') with the top-1 predicted boundary box without additional filtering.

Analysis of Similarity-based Evasion. We further analyze phishing samples that visually mimic legitimate target logos and where models correctly identify logo regions but fail in detection ('CorrectLogoArea' failures). Samples failing to bypass validations (CRPs and logo ratio checks) are excluded. The remaining sample sizes are as follows: `PhishIntention` (28 samples), `Phishpedia` (206 samples), `PhishZoo` (633 samples), and `Involution` (67 samples). Most of the feature similarities between samples and reference lists fall just below each model's detection threshold—the majority of `PhishIntention` and `Phishpedia` samples at 0.6–0.83 (threshold: 0.83), all samples of `PhishZoo` at 0–40 (threshold: 40), and most `Involution` samples between 0.6–0.7 (threshold: 0.7).

The visual quality metrics SSIM [67] and PSNR [27] suggest that these samples maintain a reasonable visual similarity to legitimate logos, although the specific target logo images are unknown and calculations are based on researchers' selection. Additionally, a small subset of samples demonstrates high quality with high values of SSIM and PSNR. LIME [56] analysis of a Facebook phishing example (Figure 3) provides additional insight: while the logo appears authentic, modifications appear in areas that may affect model detection but remain less noticeable to humans. These findings could indicate potential adversarial manipulation in real-world phishing attacks, suggesting attackers might be developing methods to maintain visual similarity while avoiding detection thresholds.

DynaPhish, PhishIntention, and Phishpedia. The systems `PhishIntention` and `DynaPhish` share a core structure of logo detection, brand verification, and domain checking, with some distinctions: `PhishIntention` adds CRP checking and OCR-aided features, while `DynaPhish` employs Google Search APIs for brand-domain verification. Several vulnerabilities exist in these approaches: Faster-RCNN's imprecise logo detection (as seen in Figure 6(c)), the assumption that phishing sites require CRPs (missing alternatives like QR codes), and challenges with borderline logo similarity cases (Figure 6(d)). Attackers can exploit these weaknesses by manipulating URLs or adjusting brand similarities. For non-targeted brand logos, attackers maintain semantic meaning while controlling similarity through strategies like 'Elimination' (*e.g.*, Figure 6(e)), background modifications, or text additions. The OCR integration in `PhishIntention` improves textual logo detection, as demonstrated in Figure 6(f), where it successfully identifies Facebook (similarity: 0.92) while `Phishpedia` fails (similarity: 0.78, below its 0.83 threshold). Additional examples appear in Figure 6.

Involution. The logo area detection and Involution model are used as the pipeline. Table 5 reveals that only 24% of samples have matching logos, while 76% bypass detection through basic modifications. Specifically, logo removal causes 4.80% of failures, text additions to screenshots impair Faster-RCNN's logo region detection in 15.30% of cases, and alternative logo integration (*e.g.*, Figure 6(i)) accounts for 11.7% of failures.

PhishZoo. The system shows strong phishing detection but has limitations in identification due to SIFT's poor performance in matching screenshots and logos. The keyword selection process from parsed URLs and HTML is problematic – for example, in Figure 6(g), it selects generic terms like "Page" and "Password," while for legitimate AT&T content, it chooses irrelevant words like "arrowmenu" and "verse." Neither set captures phishing-specific markers, reducing accuracy. Additionally, 35.10% of failures occur when samples differ from target appearances, with text additions significantly impacting results, as shown in Figure 6(h).

5.4.2 Screenshot-based Methods

Identifying the exact cause of failure is challenging due to the multiple visual components in screenshots. Therefore, we examine distance or similarity metrics. We find that 93.40% of failed samples for `EMD` have distances ranging from 0.90 to 0.94, close to its detection threshold (0.94), indicating that these samples are particularly challenging to distinguish without additional context. Additionally, 64.40% of failed samples fall within the 0.93 to 0.94 range, while 6.60% display zero EMD values from the target list. Figure 6(a) illustrates an example where the EMD value is zero, and the target reference list only contains outdated screenshots. For `VisualPhishNet`, distances vary from 1.00 to 1.99, with 83.30% of samples having distances between 1.00 and 1.50 and only 16.70% ranging from 1.5 to 2.0. Figure 6(b) shows

Table 7: **Ablation Study Result.**

Model	Component			Result	
	Logo	URL	HTML	**Detection**	**Identification**
PhishIntention	**C**[1]	**C**	**C**	3/110 (2.73%)	3/3 (100.00%)
	C	**C**	**M**[2]	2/110 (1.82%)	2/2 (100.00%)
	M	**C**	**C**	358/7,113 (5.03%)	165/358 (46.09%)
	C	**M**	**C**	296/1,321 (22.41%)	296/296 (100.00%)
Phishpedia	**C**	**C**	**C**	8/110 (7.27%)	7/8 (87.50%)
	C	**C**	**M**	—	—
	M	**C**	**C**	991/7,113 (13.93%)	323/991 (32.59%)
	C	**M**	**C**	906/1,321 (68.58%)	894/906 (98.69%)
DynaPhish	**C**	**C**	**C**	3/110 (2.73%)	3/3 (100.00%)
	C	**C**	**M**	2/110 (1.82%)	2/2(100.00%)
	M	**C**	**C**	1,372/7,113 (19.29%)	1,218/1,372 (88.78%)
	C	**M**	**C**	77/1,321 (5.83%)	77/77 (100.00%)
PhishZoo	**C**	**C**	**C**	81/110 (73.64%)	4/81 (4.94%)
	C	**C**	**M**	24/110 (21.82%)	3/24 (12.50%)
	M	**C**	**C**	6,916/7,113 (97.23%)	1,472/6,916 (21.28%)
	C	**M**	**C**	973/1,321 (73.66%)	48/973 (4.93%)

[1]*C* = Controlled component; [2]*M* = Modified Components: manipulated logos typo-squatting URLs, and empty HTMLs.

an example very similar to a screenshot in the reference list but slightly above the threshold (1.0). These observations suggest that setting fixed thresholds can be risky because attackers design adversarial images to closely mimic benign ones while slightly exceeding the threshold of detectors.

Takeaway 7: Analysis reveals three key weaknesses in current phishing detection systems: (1) pipeline exploitation through logo manipulation and CRP circumvention, (2) visually plausible modifications that remain convincing to humans while falling below detection thresholds, and (3) straightforward evasion techniques such as text overlay and logo removal. These findings indicate an overreliance on static feature matching and predetermined thresholds, underscoring the necessity for detection methods that incorporate dynamic, contextual awareness.

5.5 Ablation Study

Study Plan. We conduct a detailed ablation study evaluating the performance of four models (`PhishIntention`, `Phishpedia`, `DynaPhish`, and `PhishZoo`) across distinct scenarios. The first scenario uses completely legitimate inputs, incorporating authentic logos, URLs, and HTML structure. This establishes our baseline for normal website behavior. In the second scenario, we modify the input by maintaining legitimate logos and HTML while introducing typosquatted URLs, allowing us to assess the impact of URL manipulation. The third scenario uses legitimate logos and URLs but combines them with an empty HTML structure, enabling us to isolate the role of HTML content in detection accuracy.

Results. Table 7 presents the performance of phishing detection and identification in our ablation study for `PhishIntention`, `Phishpedia`, `DynaPhish`, and `PhishZoo`. In Table 7, controlled original legitimate components are denoted as **C** and modified components as **M**.

Our analysis comparing configurations with and without HTML components ('CCC' and 'CCM') reveals varying impacts on false positive rates. `PhishIntention` and `DynaPhish` showed a minor improvement, reducing false positives from 3 to 2 cases due to their handling of inadequately maintained domains. More notably, `PhishZoo` demonstrated substantial improvement, with false positives decreasing from 81 to 24 when HTML was removed, suggesting that its HTML analysis through keyword matching may impair detection accuracy and require methodology refinement.

The analysis reveals significant variations in phishing detection performance when comparing original versus modified logos ('CCC' and 'MCC'). `PhishIntention`, `Phishpedia`, and `DynaPhish` showed relatively low detection rates of 5.03%, 13.93%, and 19.29% respectively with modified logos, indicating substantial vulnerability to logo-based evasion tactics. While `PhishZoo` reported a 97.23% detection rate, this figure appears inflated due to poor identification accuracy and high false positives. `DynaPhish`'s superior performance in detecting modified logos (19.29% vs `PhishIntention`'s 5.03%) can be attributed to its real-time web search capability, demonstrating the value of incorporating online search features in phishing detection systems.

Under modified URL configurations ('CCC' and 'CMC'), `Phishpedia` achieves a 68.58% detection rate, demonstrating both resilience and limitations in its dual URL-logo analysis approach. This performance indicates the need to enhance domain verification through expanded databases or alternative domain-brand authentication methods.

Takeaway 8: Logos, URLs, and HTML are critical components that significantly influence the results. The current simple processing of HTML for `PhishZoo` has detrimental impacts on its performance.

6 Evaluation of Model Robustness against Manipulated Visual Components

To further investigate why visual similarity-based anti-phishing models fail, we categorize possible reasons into (1) visible manipulations, where simple modifications have the possibility to be detectable by humans; and (2) perturbed manipulations, where phishing logos closely resemble target brands. Representative manipulations are selected from Table 5 and Table 12. We employ several white-box and black-box attack methods on the benign screenshots to imitate the perturbed manipulations. Details can be found in Appendix A.5. The robustness results for seven models on crafted datasets, covering phishing detection and phishing brand identification rates, are detailed in Table 8 and Table 9.

Evaluation Plan. The crafted samples are equipped with original benign URLs and HTML for evaluation by default. To assess the impact of URLs with crafted visual images, we

Table 8: **Evaluation Results of Phishing Detection with Manipulated Visual Components.** Models, trained on $\boldsymbol{R}_{ext}$, are used for evaluation. The row means manipulating type; the original is the brand's recent benign webpage, visible manipulation, and perturbation-based manipulation, which refers to the dataset crafted for robustness testing. The column for different models means URL type, benign refers to the original benign URL, while squatted means the created URLs. Default benign URLs are used. Note that the row of 'Original' with benign URLs means *misclassified*.

	Manipulation Name	PhishIntention		Phishpedia		DynaPhish	Involution	PhishZoo	VisualPhishNet	EMD
		Benign	Squatted	Benign	Squatted					
	Original	3/110 (2.73%)	316/1,321 (23.92%)	8/110 (7.27%)	894/1,321 (67.68%)	3/110 (2.73%)	8/110 (7.27%)	81/110 (73.64%)	30/110 (27.27%)	55/110 (50.00%)
Visible Manipulation	Elimination	0/110 (0.0%)	10/1,321 (0.76%)	10/110 (9.09%)	151/1,321 (11.43%)	1/110 (0.91%)	3/110 (2.73%)	103/110 (93.64%)	28/110 (25.45%)	54/110 (49.09%)
	ColorReplace	4/580 (0.69%)	858/6,965 (12.32%)	44/580 (7.59%)	1,248/6,965 (17.92%)	56/580 (9.66%)	327/580 (56.38%)	562/580 (96.90%)	125/580 (21.55%)	275/580 (47.41%)
	Resizing	29/877 (3.30%)	2,230/10,532 (21.17%)	65/877 (7.41%)	6,383/10,532 (60.61%)	200/877 (22.81%)	696/877 (79.36%)	856/877 (97.61%)	242/877 (27.59%)	421/877 (48.00%)
	Rotation	36/1,320 (2.73%)	3,612/15,852 (22.79%)	96/1,320 (7.27%)	10,442/15,852 (65.87%)	326/1,320 (24.70)	1,085/1,320 (82.20%)	1,283/1,320 (97.20%)	387/1,320 (29.32%)	671/1,320 (50.83%)
	Integration	22/369 (5.96%)	817/4,431 (18.44%)	58/369 (15.72%)	2,624/4,431 (59.22%)	77/369 (20.88)	217/369 (58.81%)	362/369 (98.10%)	95/369 (25.75%)	188/369 (50.95%)
	Re-position	24/879 (2.73%)	2,035/10,556 (19.28%)	66/879 (7.51%)	6,278/10,556 (59.47%)	190/879 (21.62%)	587/879 (66.78%)	870/879 (98.98%)	216/879 (24.57%)	432/879 (49.15%)
	Flipping	4/220 (1.82%)	459/2,642 (17.37%)	16/220 (7.27%)	1692/2,642 (64.04%)	36/220 (16.36%)	28/220 (12.73%)	216/220 (98.18%)	62/220 (28.18%)	107/220 (48.64%)
	Replacement	177/1,006 (17.59%)	2,202/12,081 (18.23%)	444/1,006 (44.14%)	5,545/12,081 (45.90%)	145/1006 (14.41%)	502/1,006 (49.90%)	967/1,006 (96.12%)	235/1,006 (23.36%)	468/1,006 (46.52%)
	Blurring	1/110 (0.91%)	122/1,321 (9.24%)	3/110 (2.73%)	426/1,321 (32.20%)	7/110 (6.36%)	32/110 (29.09%)	102/110 (92.73%)	29/110 (26.36%)	51/110 (46.36%)
	Scaling	20/550 (3.64%)	1,530/6,605 (23.16%)	41/550 (7.45%)	4,536/6,605 (68.68%)	144/550 (26.18%)	461/550 (83.82%)	538/550 (97.82%)	147/550 (26.73%)	274/550 (49.82%)
	Omission	2/96 (2.08%)	114/1,152 (9.90%)	11/96 (11.46%)	451/1,152 (39.15%)	5/96 (5.21%)	52/96 (54.17%)	93/96 (96.88%)	34/96 (35.42%)	42/96 (43.75%)
	FontReplace	4/186 (2.15%)	294/2,232 (13.17%)	23/186 (12.37%)	774/2,232 (34.68%)	26/186 (13.98%)	115/186 (61.83%)	179/186 (96.24%)	56/186 (30.11%)	90/186 (48.39%)
	CaseConversion	6/225 (2.67%)	299/2,700 (11.07%)	36/225 (16.00%)	960/2,700 (35.56%)	22/225 (9.78%)	76/225 (33.78%)	214/225 (95.11%)	73/225 (32.44%)	113/225 (50.22%)
	Total	329/6,528 (5.04%)	14,582/78,390 (18.60%)	913/6,528 (13.99%)	41,510/78,390 (52.95%)	1,235/6,528 (18.92%)	4,181/6,528 (64.05%)	6,345/6,528 (97.20%)	1,729/6,528 (26.49%)	3,186/6,528 (48.81%)
	SRNet	11/41 (26.83%)	114/492 (23.17%)	34/41 (82.93%)	347/492 (70.53%)	8/41 (19.51%)	25/41 (60.98%)	39/41 (95.12%)	11/41 (26.83%)	20/41 (48.78%)
Perturbation-based	[35]-ViT	3/110 (2.78%)	266/1,321 (20.14%)	8/110 (7.27%)	696/1,321 (52.69%)	26/110 (23.64%)	82/110 (74.55%)	106/110 (96.36%)	34/110 (30.91%)	52/110 (47.27%)
	[35]-Swin	3/110 (2.73%)	296/1,321 (22.41%)	10/110 (9.09%)	820/1,321 (62.07%)	28/110 (25.45%)	86/110 (78.18%)	108/110 (98.18%)	34/110 (30.91%)	55/110 (50.00%)
	FGSM	4/108 (3.70%)	279/1,297 (21.51%)	8/108 (7.41%)	776/1,297 (59.83%)	25/110 (22.73%)	80/108 (74.07%)	106/108 (98.15%)	30/108 (27.78%)	52/108 (48.15%)
	PGD	4/108 (3.70%)	259/1,297 (19.97%)	8/108 (7.41%)	756/1,297 (58.29%)	25/108 (23.15%)	79/108 (73.15%)	106/108 (98.15%)	30/108 (27.78%)	51/108 (47.22%)
	CW	4/108 (3.70%)	269/1,297 (20.74%)	10/108 (9.26%)	790/1,297 (60.90%)	25/108 (23.15%)	80/108 (74.07%)	106/108 (98.15%)	30/108 (27.78%)	52/108 (48.15%)
	Total	18/544 (3.31%)	1,369/6,533 (20.96%)	44/544 (8.09%)	3,838/6,533 (58.75%)	129/544 (23.71%)	407/544 (74.82%)	532/544 (97.79%)	158/544 (29.04%)	262/544 (48.16%)

use squatted domains (*e.g.*, faceb00k .com) to replace the original benign URLs for `PhishIntention` and `Phishpedia`, as they employ second-level domains to verify legitimacy. The performance represents the upper bound for methods that are not equipped with domain checks.

Settings. We use domains from `PhishIntention` and the $\boldsymbol{R}_{ext}$ as reference lists. Metrics are the same as in Section 5.

6.1 Result: Robustness Evaluation

Legitimate Samples (False Positive). Legitimate samples (legitimate screenshots, URLs, and HTML) are expected to be correctly identified as benign. Incorrectly labeling benign samples as potential phishing attempts is defined as a false positive error. As shown in Table 8, `PhishIntention` detects 3 benign samples (with legitimate domains) as phishing, 8 for `Phishpedia`, and 3 for `DynaPhish`. The misclassification of `PhishIntention` and `Phishpedia` stems from the reliance on brand-domain verification. Some legitimate domains, such as 'santanderbank,' are not included in the list, although 'santander' and 'santanderresearch' exist in the list. This oversight highlights the limitations of relying on incomplete reference lists for verification purposes. The error for `DynaPhish` comes from the "forbidden words" maintained by the authors.

Takeaway 9: Models that rely on incomplete reference lists and static word matching to verify the legitimacy of logos and domains are prone to false positive errors, incorrectly flagging legitimate websites as phishing threats.

Visible and Perturbation-based Manipulation Methods. From Table 8 and Table 9, we observe that visible and perturbation-based strategies impact the identification result, but they do not affect the detection result significantly for `PhishZoo`. Specifically, it achieves a 97.79% detection rate on the perturbation-based adversarial manipulation dataset and 96.16% on the visible manipulation dataset. Logo elimination, blurring, replacing font manually or by `SRNet`, and converting cases are critical factors in the model's phishing detection capability, whereas replacing logos markedly influences identification results. `PhishZoo` is less sensitive to the combined logos but sensitive to the white-box attack in phishing identification, which means it is not robust on the perturbation-based manipulations. For instance, Vit and Swin-based methods achieve 25.47% and 32.41% while FSGM, PGD, and CW only achieve 17.92%, 13.21%, and 17.92%, respectively. For other logo-based approaches, manipulations such as logo deletion, flipping, blurring, and case conversion substantially affect detection results. Meanwhile, changing colors, combining logos, or replaced with other logos play important roles in identification. Although logo text font greatly affects `Phishpedia`, it does not significantly impact `Involution`. `DynaPhish` is effective and robust in recognizing brands but may make mistakes when there are two logos.

Although the influence of perturbation-based attacks is not as great as the visible manipulation, they reveal weaknesses in the models: `PhishIntenion` and `Phishpedia` are sensitive to the `ViT-based` black-box attack and the `PGD` white-box attack. `Involution` is not robust on white-box attacks, and `PhishZoo` is susceptible to both attacks. For screenshot-based methods, detection performance remains stable across various manipulations, with perturbation-based manipulation strategies even improving detection rates in

Table 9: **Evaluation Results of Phishing Brand Identification with Manipulated Visual Components.** Models, trained on $\boldsymbol{R}_{ext}$, are used for evaluation. The percentage is the correctly identified brands out of the predicted phishing number by default.

	Manipulation Name	PhishIntention		Phishpedia		DynaPhish	Involution	PhishZoo	VisualPhishNet	EMD
		Benign	Squatted	Benign	Squatted					
	Original	3/3 (100.0%)	316/316 (100.0%)	7/8 (87.50%)	882/894 (98.66%)	3/3 (100.0%)	7/8 (87.50%)	4/81 (4.94%)	16/30 (53.33%)	11/55 (20.0%)
Visible Manipulation	Elimination	0/0 (0.0%)	10/10 (100.0%)	0/10 (0.0%)	30/151 (19.87%)	1/1 (100.0%)	3/3 (100.0%)	7/103 (6.80%)	6/28 (25.43%)	9/54 (16.67%)
	ColorReplace	4/4 (100.0%)	858/858 (100.0%)	6/44 (13.64%)	792/1,248 (63.46%)	56/56 (100.0%)	322/327 (98.47%)	69/562 (12.28%)	39/125 (31.20%)	49/275 (17.82%)
	Resizing	29/29 (100.0%)	2,230/2,230 (100.0%)	42/65 (64.62%)	6,112/6,383 (95.75%)	200/200 (100.0%)	688/696 (98.85%)	160/856 (18.69%)	140/242 (57.85%)	80/421 (19.00%)
	Rotation	36/36 (100.0%)	3,612/3,612 (100.0%)	81/96 (84.38%)	10,262/10,442 (98.28%)	326/326 (100%)	1,072/1,085 (98.80%)	389/1,283 (30.32%)	202/387 (52.20%)	121/671 (18.03%)
	Integration	9/22 (40.91%)	661/817 (80.91%)	23/58 (39.66%)	2,206/2,624 (84.07%)	61/77 (79.22%)	188/217 (86.64%)	119/362 (32.87%)	38/95 (40.00%)	37/188 (19.68%)
	Location	24/24 (100.0%)	2,035/2,035 (100.0%)	47/66 (71.21%)	6,051/6,278 (96.38%)	190/190 (100.0%)	578/587 (98.47%)	288/870 (33.10%)	103/216 (47.69%)	79/432 (18.29%)
	Flipping	4/4 (100.0%)	459/459 (100.0%)	13/16 (81.25%)	1,656/1,692 (97.87%)	36/36 (100.0%)	28/28 (100.0%)	28/216 (12.96%)	34/62 (54.84%)	19/107 (17.76%)
	Replacement	0/177 (0.0%)	70/2,202 (3.18%)	0/444 (0.0%)	210/5,545 (3.79%)	7/145 (4.83%)	27/502 (5.38%)	63/967 (6.51%)	63/235 (26.81%)	67/468 (14.32%)
	Blurring	1/1 (100.0%)	122/122 (100.0%)	1/3 (33.33%)	402/426 (94.37%)	7/7 (100.0%)	23/32 (71.88%)	5/102 (4.90%)	17/29 (58.62%)	11/51 (21.57%)
	Scaling	20/20 (100.0%)	1,530/1,530 (100.0%)	35/41 (85.37%)	4,466/4,536 (98.46%)	144/144 (100.0%)	455/461 (98.70%)	155/538 (28.81%)	89/147 (60.54%)	48/274 (17.52%)
	Omission	2/2 (100.0%)	114/114 (100.0%)	3/11 (27.27%)	355/451 (78.71%)	5/5 (100.0%)	47/52 (90.38%)	20/93 (21.51%)	8/34 (23.53%)	5/42 (11.90%)
	FontReplace	4/4 (100.0%)	294/294 (100.0%)	5/23 (21.74%)	558/774 (72.09%)	26/26 (100.0%)	111/115 (96.52%)	24/179 (13.41%)	25/56 (44.64%)	17/90 (18.89%)
	CaseConversion	3/6 (50.0%)	263/299 (87.96%)	5/36 (13.89%)	588/960 (61.25%)	22/22 (100.0%)	74/76 (97.37%)	25/214 (11.68%)	36/73 (49.32%)	20/113 (17.70%)
	Total	136/329 (41.34%)	12,258/14,582 (84.06%)	261/913 (28.59%)	33,688/41,510 (81.16%)	1,081/1,235 (87.53%)	3,616/4,181 (86.49%)	1,352/6,345 (21.31%)	800/1,729 (46.27%)	562/3,186 (17.64%)
	SRNet	11/11 (100.0%)	114/114 (100.0%)	34/34 (100.0%)	347/347 (100.0%)	8/8 (100.0%)	25/25 (100.0%)	6/39 (15.38%)	5/11 (45.45%)	3/20 (15.00%)
Perturbation-based	[35]-ViT	3/3 (100.0%)	266/266 (100.0%)	4/8 (50.00%)	648/696 (93.10%)	26/26 (100.0%)	81/82 (98.78%)	27/106 (25.47%)	18/34 (52.94%)	10/52 (19.23%)
	[35]-Swin	3/3 (100.0%)	296/296 (100.0%)	6/10 (60.00%)	772/820 (94.15%)	28/28 (100.0%)	85/86 (98.84%)	35/108 (32.41%)	17/34 (50.00%)	12/55 (21.82%)
	FSGM	4/4 (100.0%)	279/279 (100.0%)	6/8 (75.00%)	752/776 (96.91%)	25/25 (100%)	79/80 (98.75%)	19/106 (17.92%)	15/30 (50.00%)	9/52 (17.31%)
	PGD	4/4 (100.0%)	259/259 (100.0%)	6/8 (75.00%)	732/756 (96.83%)	25/25 (100%)	78/79 (98.73%)	14/106 (13.21%)	14/30 (46.67%)	7/51 (13.73%)
	CW	4/4 (100.0%)	269/269 (100.0%)	6/10 (60.00%)	742/790 (93.92%)	25/25 (100%)	79/80 (98.75%)	19/106 (17.92%)	15/30 (50.00%)	9/52 (17.31%)
	Total	18/18 (100.0%)	1,369/1,369 (100.0%)	28/44 (63.64%)	3,646/3,838 (95.00%)	129/129 (100.0%)	402/407 (98.77%)	114/532 (21.43%)	79/158 (50.00%)	47/262 (17.94%)

`VisualPhishNet`. However, these methods struggle with accurately identifying the target brand. Additionally, decreased performance observed when logos are deleted, replaced, or divided in identification results underscores the crucial role of logos in brand recognition within screenshot-based methods. We mention that the attacked logos are obtained based on one model and transferred to test other models. The results indicate the transferability of the attacks.

Takeaway 10: Simple visible and perturbation-based manipulations significantly disrupt logo-based methods. Both of them are transferable. Screenshot-based methods maintain stable detection but struggle with identifying brands when logos are altered.

Benign Vs. Squatted Domains. `PhishIntention` shows varied performance shifts when tested with squatted versus benign URLs: +13.56% for manual visible manipulations, -3.66% for SRNet, and +17.65% for perturbation-based manipulations. `Phishpedia`'s detection rates change from 13.99% to 52.95% for manual visible manipulations, 82.93% to 70.5% for SRNet, and 8.09% to 58.75% for perturbation-based manipulations.

These results highlight the critical role of domain validation in both models' detection mechanisms. By comparing detected brand domains with parsed URL domains, the models become vulnerable to URL manipulation attacks. Attackers can potentially bypass detection by using squatted domains that match benign domains (e.g., 'www.capitalone.aaa' targeting 'www.capitalone.com'). URL structure parsing also presents vulnerabilities, as demonstrated by `tldextract` (`https://github.com/john-kurkowski/tldextract`) parsing 'https://home.barclays/' as 'home' rather than the more relevant 'barclays'.

Takeaway 11: Models that rely on brand domain checking heavily depend on the structure of URLs, the URL parsing method, and the comparison against the maintained second-level domain.

6.2 Case Study of Failures

`PhishIntention` leverages OCR to incorporate textual information, outperforming `Phishpedia` on textual logos. For example, `PhishIntention` correctly identifies the brand of Figure 5, while `Phishpedia` misclassifies it as 'timeweb,' highlighting the crucial role of OCR in logo character recognition. We further check the manipulated 'YouTube' logos in Table 14 against brand reference lists. `Phishpedia` shows lower similarity scores (0.5–0.6) for 'Elimination,' 'Color Replace,' and 'Integration,' while other manipulations are around 0.9. `PhishIntention` considers all as benign due to the absent CRP in HTML. `PhishZoo` successfully identifies candidate keywords with varying similarity scores across manipulations. `Involution` mostly fails to recognize the brand (similarity around 0.57), except misidentifying the 'Case Conversion' example as 'AOL.' `VisualPhishNet` misclassifies all examples as other brands (scores 1.1–1.3), while `EMD` predicts 'Airbnb' (distance 0.96). The results indicate the difficulty in setting appropriate similarity score thresholds for each solution.

Takeaway 12: Textual information of visual elements and appropriate similarity thresholds significantly impact performance. Future models should integrate advanced OCR, human-centric similarity metrics, and multi-modal analysis combining visual and contextual information.

7 Discussion

Recommendations. We propose several key improvements focused on comprehensive threat detection and resilience against manipulation attacks. First, integrating advanced text recognition capabilities through OCR-aided deep learning models or online search verification would significantly strengthen brand identification accuracy. This enhancement would address current limitations in systems (`Phishpedia`) that struggle with semantic variations in phishing attempts.

Second, detection systems must strengthen their defenses through comprehensive adversarial training that incorporates manipulated logos and visual elements. This can be achieved by systematically exposing machine learning models to real-world manipulation patterns during the training phase. Organizations should implement sophisticated data augmentation techniques that account for common visual modifications, including scaling transformations, color adjustments, and other alterations that may be used to evade detection.

Third, we recommend implementing a holistic, multi-modal detection approach. It should integrate analysis across multiple dimensions: examining logo characteristics, evaluating webpage structural elements, assessing textual content authenticity, and analyzing additional visual indicators.

Finally, we recommend using preprocessing and normalization techniques, including image scaling and denoising, before visual similarity analysis. These methods can reduce the efficacy of adversarial manipulations and provide additional layers of defense against sophisticated phishing tactics.

Limitations. Our work has a few key limitations. First, the lack of a user study limits the assessment of our manipulation methods' effectiveness in real-world scenarios, as users may readily recognize manipulated logo images. To address this, we perform manual verification with 500 randomly generated images to ensure our manipulations are not easily recognizable. However, a comprehensive user study would be valuable for gathering insights into the perceptibility and deceptiveness of adversarial manipulations.

Second, our analysis is limited to logo manipulations and does not consider other webpage elements that could be targeted. Expanding the scope could provide a more comprehensive understanding of potential attack vectors. Third, our evaluation is conducted only on models and datasets with publicly available source code and data. Despite these limitations, we highlight the potential vulnerabilities of visual similarity-based anti-phishing systems and the need for robust defense mechanisms against adversarial visual manipulations.

8 Related Work

Visual Similarity-based Detection. Visual similarity-based phishing detection systems compare suspicious websites against legitimate ones to identify threats. While Panum *et al*. [53] and Abuadbba *et al*. [2] examined detector robustness and evolving phishing trends, their evaluations lacked comprehensive real-world testing. Literature reviews by Zieni *et al*. [77] and Hou *et al*. [28] documented detection techniques but provided no comparative performance analyses. Our work presents the first comprehensive evaluation of visual similarity-based phishing detectors in controlled environments with consistent brand knowledge across models. We expand upon Liu *et al*. [43]'s `PhishIntention` evaluation by testing detector performance on extensive APWG phishing data and incorporating explainable AI techniques based on LIME [56], similar to Charmet *et al*. [13]'s approach, to understand brand impersonation detection.

Evaluation of Robustness against Adversarial Manipulation. Adversarial attacks use carefully crafted perturbations to manipulate machine learning model predictions [20, 47]. While Lee *et al*. [35] demonstrated how perturbation vectors could bypass phishing detectors, their analysis of visual impact was limited (*i.e*., perturbations on the logo components). Similarly, Ying *et al*. [74]'s user studies on webpage modifications overlooked the resilience of logo-based detection systems. Hao *et al*. [22] also worked on logo manipulation, which focused primarily on style and font modifications, our study examines detector robustness against a comprehensive range of techniques, including 14 visible manipulations and 5 adversarial perturbations observed in real phishing attacks.

9 Conclusion

In this comprehensive evaluation of seven visual similarity-based anti-phishing models across 451k real-world phishing websites, we identified substantial performance disparities between controlled testing environments and real-world applications. Our analysis exposed critical weaknesses that could be exploited through adversarial visual manipulations. To strengthen these systems, we recommend integrating text recognition with visual analysis, implementing data augmentation with adversarial examples, adopting a multi-cue ensemble approach, and utilizing preprocessing techniques such as scaling and denoising. These enhancements are essential for developing more robust and reliable phishing detection systems capable of addressing real-world threats.

10 Acknowledgments

We sincerely thank the anonymous shepherd and all the reviewers for their constructive comments and suggestions. This work is supported in part by the NSF (2210137 and 2335798), a seed grant from the AI Tennessee Initiative at the University of Tennessee Knoxville, Science Alliance's StART program, gifts from Google exploreCSR, and IITP grants from South Korean government (RS-2024-00439762 and RS-2024-00419073). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

11 Ethics Consideration

Our research involving the APWG eCX dataset focused solely on reported phishing websites, ensuring that no benign, legitimate sites were affected. Importantly, no personal data from users or phishing websites was collected or used in our study. To maintain ethical standards, we share only selected failed phishing examples, providing HTML and screenshot data without revealing URLs from the APWG dataset.

Our research utilizes exclusively open-source models, and the techniques we examine may be employed by malicious actors. By openly sharing our source code and findings, we aim to strengthen cybersecurity defenses against phishing attacks. We believe the security benefits of transparency – enabling defenders to better understand and counter these threats – outweigh the potential risks, particularly since attackers already know these methods. This open approach aligns with our commitment to advancing collective cybersecurity capabilities.

12 Open Science

To facilitate reproducibility and accelerate scientific progress (*i.e.*, strengthening collective efforts in combating phishing attacks), we publicly share: (1) our collected datasets, (2) code, and (3) retrained models. The resources are available on our website (https://moa-lab.net/evaluation-visual-similarity-based-phishing-detection-models/) or Zenodo (https://zenodo.org/records/14668190).

Collected Datasets. We publicly share the 451,514 real-world phishing data that our web crawler collected. This dataset includes both HTML source files and visual screenshots of phishing websites. Due to licensing agreements with the Anti-Phishing Working Group (APWG), the original phishing URLs are withheld from public sharing. Moreover, we share our extended reference list ($\boldsymbol{R}_{ext}$) that is used for our evaluation (see Section 4.1). Furthermore, we share a general benign dataset that covers 100 Tranco domains.

Manipulated Phishing Screenshots. We publicly share 7,223 manipulated screenshots, including 110 original and all manipulated screenshots.

Failed Sampled Screenshots and HTML. We publicly share 6,000 failed, sampled screenshots and HTML with detailed CSV files documenting model-specific failure cases for phishing detection.

Code. We publicly share all our code for collecting datasets and evaluating models under an open-source license: (1) testing code with an open-source license, (2) preprocessing codes for clustering, (3) web crawler source code, and (4) perturbation-based attacking code.

Retrained Models. We have retrained three models (PhishIntension, Phishpedia, and Involution) for evaluation. We publicly share our retrained models as they are clearly under MIT or CC0-1.0 license, which explicitly permits model modification and redistribution.

References

[1] Sahar Abdelnabi, Katharina Krombholz, and Mario Fritz. Visualphishnet: Zero-day phishing website detection by visual similarity. In *Proc. of the ACM SIGSAC Conference on Computer and Communications Security*, 2020.

[2] Alsharif Abuadbba, Shuo Wang, Mahathir Almashor, Muhammed Ejaz Ahmed, Raj Gaire, Seyit Camtepe, and Surya Nepal. Towards web phishing detection limitations and mitigation. *arXiv:2204.00985*, 2022.

[3] Sadia Afroz and Rachel Greenstadt. Phishzoo: Detecting phishing websites by looking at them. In *Proc. of the IEEE International Conference on Semantic Computing*, 2011.

[4] Babak Amin Azad, Oleksii Starov, Pierre Laperdrix, and Nick Nikiforakis. Web runner 2049: Evaluating third-party anti-bot services. In *Proc. of the Detection of Intrusions and Malware, and Vulnerability Assessment*, 2020.

[5] G. Apruzzese, H. S. Anderson, S. Dambra, D. Freeman, F. Pierazzi, and K. Roundy. Real attackers don't compute gradients: Bridging the gap between adversarial ml research and practice. In *Proc. of the IEEE Conference on Secure and Trustworthy Machine Learning*, 2023.

[6] Apwg ecx. https://apwg.org/ecx/. (Accessed on 04/22/2024).

[7] Muhammet Bastan, Hao-Yu Wu, Tian Cao, Bhargava Kota, and Mehmet Tek. Large scale open-set deep logo detection. *arXiv:1911.07440*, 2019.

[8] Marisa Bernabeu, Antonio Javier Gallego, and Antonio Pertusa. Multi-label logo recognition and retrieval based on weighted fusion of neural features. *Expert Systems*, 2022.

[9] Manish Bhurtel, Yuba R Siwakoti, and Danda B Rawat. Phishing attack detection with ml-based siamese empowered orb logo recognition and ip mapper. In *Proc. of the IEEE INFOCOM Conference on Computer Communications Workshops*, 2022.

[10] Ahmet Selman Bozkir and Ebru Akcapinar Sezer. Use of hog descriptors in phishing detection. In *Proc. of the International Symposium on Digital Forensic and Security*, 2016.

[11] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *Proc. of the IEEE Symposium on Security and Privacy*, 2017.

[12] Ee Hung Chang, Kang Leng Chiew, San Nah Sze, and Wei King Tiong. Phishing detection via identification

of website identity. In *Proc. of the International Conference on IT Convergence and Security*, 2013.

[13] Fabien Charmet, Tomohiro Morikawa, Akira Tanaka, and Takeshi Takahashi. Vortex: Visual phishing detections are through explanations. *ACM Trans. Internet Technol.*, 2024.

[14] Kuan-Ta Chen, Jau-Yuan Chen, Chun-Rong Huang, and Chu-Song Chen. Fighting phishing with discriminative keypoint features. *IEEE Internet Computing*, 2009.

[15] Igino Corona, Battista Biggio, Matteo Contini, Luca Piras, Roberto Corda, Mauro Mereu, Guido Mureddu, Davide Ariu, and Fabio Roli. Deltaphish: Detecting phishing webpages in compromised websites. In *Proc. of the European Symposium on Research in Computer Security*, 2017.

[16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proc. of the International Conference on Learning Representations*, 2021.

[17] Matthew Dunlop, Stephen Groat, and David Shelly. Goldphish: Using images for content-based phishing analysis. In *Proc. of the IEEE International Conference on Internet Monitoring and Protection*, 2010.

[18] Fastdup. https://github.com/visual-layer/fastdup. (Accessed on 04/24/2024).

[19] Anthony Y. Fu, Liu Wenyin, and Xiaotie Deng. Detecting phishing web pages with visual similarity assessment based on earth mover's distance. *IEEE Transactions on Dependable and Secure Computing*, 2006.

[20] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *Proc. of the International Conference on Learning Representations*, 2015.

[21] Izzeddin Gur, Ofir Nachum, Yingjie Miao, Mustafa Safdari, Austin Huang, Aakanksha Chowdhery, Sharan Narang, Noah Fiedel, and Aleksandra Faust. Understanding HTML with large language models. In *Proc. of the Findings of the Association for Computational Linguistics*, 2023.

[22] Qingying Hao, Nirav Diwan, Ying Yuan, Giovanni Apruzzese, Mauro Conti, and Gang Wang. It doesn't look like anything to me: Using diffusion model to subvert visual phishing detectors. In *Proc. of the USENIX Security Symposium*, 2024.

[23] Shuichiro Haruta, Hiromu Asahina, and Iwao Sasase. Visual similarity-based phishing detection scheme using image and css with target website finder. In *Proc. of the IEEE Global Communications Conference*, 2017.

[24] Frank L Hitchcock. The distribution of a product from several sources to numerous localities. *Journal of mathematics and physics*, 1941.

[25] Grant Ho, Asaf Cidon, Lior Gavish, Marco Schweighauser, Vern Paxson, Stefan Savage, Geoffrey M Voelker, and David Wagner. Detecting and characterizing lateral phishing at scale. In *Proc. of the USENIX Security Symposium*, 2019.

[26] Steven CH Hoi, Xiongwei Wu, Hantang Liu, Yue Wu, Huiqiong Wang, Hui Xue, and Qiang Wu. Logo-net: Large-scale deep logo detection and brand recognition with deep region-based convolutional networks. *arXiv:1511.02462*, 2015.

[27] Alain Horé and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *Proc. of the International Conference on Pattern Recognition*, 2010.

[28] Sujuan Hou, Jiacheng Li, Weiqing Min, Qiang Hou, Yanna Zhao, Yuanjie Zheng, and Shuqiang Jiang. Deep learning for logo detection: A survey. *ACM Trans. Multimedia Comput. Commun. Appl.*, 2023.

[29] Yannis Kalantidis, Lluis Garcia Pueyo, Michele Trevisiol, Roelof Van Zwol, and Yannis Avrithis. Scalable triangulation-based logo recognition. In *Proc. of the ACM international conference on multimedia retrieval*, 2011.

[30] Zhanghan Ke, Yuhao Liu, Lei Zhu, Nanxuan Zhao, and Rynson W. H. Lau. Neural preset for color style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[31] Doowon Kim, Haehyun Cho, Yonghwi Kwon, Adam Doupé, Sooel Son, Gail-Joon Ahn, and Tudor Dumitras. Security analysis on practices of certificate authorities in the https phishing ecosystem. In *Proc. of the ACM Asia Conference on Computer and Communications Security*, 2021.

[32] Taeri Kim, Noseong Park, Jiwon Hong, and Sang-Wook Kim. Phishing url detection: A network-based approach robust to evasion. In *Proc. of the ACM SIGSAC Conference on Computer and Communications Security*, 2022.

[33] Hung Le, Quang Pham, Doyen Sahoo, and Steven C. H. Hoi. Urlnet: Learning a URL representation with deep learning for malicious URL detection. *arXiv:1802.03162*, 2018.

[34] Jehyun Lee, Farren Tang, Pingxiao Ye, Fahim Abbasi, Phil Hay, and Dinil Mon Divakaran. D-fence: A flexible, efficient, and comprehensive phishing email detection system. In *Proc. of the IEEE European Symposium on Security and Privacy*, 2021.

[35] Jehyun Lee, Zhe Xin, Melanie Ng Pei See, Kanav Sabharwal, Giovanni Apruzzese, and Dinil Mon Divakaran. Attacking logo-based phishing website detectors with adversarial perturbations. In *Proc. of European Symposium on Research in Computer Security*, 2023.

[36] Chenge Li, István Fehérvári, Xiaonan Zhao, Ives Macedo, and Srikar Appalaraju. Seetek: Very large-scale open-set logo recognition with text-aware metric learning. In *Proc. of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022.

[37] Duo Li, Jie Hu, Changhu Wang, Xiangtai Li, Qi She, Lei Zhu, Tong Zhang, and Qifeng Chen. Involution: Inverting the inherence of convolution for visual recognition. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[38] Shuaiji Li, Tao Huang, Zhiwei Qin, Fanfang Zhang, and Yinhong Chang. Domain generation algorithms detection through deep neural network and ensemble. In *Proc. of the ACM Web Conference*, 2019.

[39] Xigao Li, Babak Amin Azad, Amir Rahmati, and Nick Nikiforakis. Good bot, bad bot: Characterizing automated browsing activity. In *Proc. of the IEEE Symposium on Security and Privacy*, 2021.

[40] Kyungchan Lim, Jaehwan Park, and Doowon Kim. Phishing vs. legit: Comparative analysis of client-side resources of phishing and target brand websites. In *Proc. of the ACM on Web Conference*, 2024.

[41] Yun Lin, Ruofan Liu, Dinil Mon Divakaran, Jun Yang Ng, Qing Zhou Chan, Yiwen Lu, Yuxuan Si, Fan Zhang, and Jin Song Dong. Phishpedia: A hybrid deep learning based approach to visually identify phishing webpages. In *Proc. of the USENIX Security Symposium*, 2021.

[42] Phishingbaseline. https://github.com/lindsey98/PhishingBaseline. (Accessed on 02/07/2024).

[43] Ruofan Liu, Yun Lin, Xianglin Yang, Siang Hwee Ng, Dinil Mon Divakaran, and Jin Song Dong. Inferring phishing intention via webpage appearance and dynamics: A deep vision based approach. In *Proc. of the USENIX Security Symposium*, 2022.

[44] Ruofan Liu, Yun Lin, Yifan Zhang, Penn Han Lee, and Jin Song Dong. Knowledge expansion and counterfactual interaction for Reference-Based phishing detection. In *Proc. of the USENIX Security Symposium*, 2023.

[45] Wenyin Liu, Xiaotie Deng, Guanglin Huang, and Anthony Y Fu. An antiphishing strategy based on visual similarity assessment. *IEEE Internet Computing*, 2006.

[46] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proc. of the IEEE/CVF international conference on computer vision*, 2021.

[47] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *Proc. of the International Conference on Learning Representations*, 2018.

[48] Eric Medvet, Engin Kirda, and Christopher Kruegel. Visual-similarity-based phishing detection. In *Proc. of the ACM International Conference on Security and Privacy in Communication Netowrks*, 2008.

[49] Biagio Montaruli, Luca Demetrio, Maura Pintor, Luca Compagna, Davide Balzarotti, and Battista Biggio. Raze to the ground: Query-efficient adversarial html attacks on machine-learning phishing webpage detectors. In *Proc. of the ACM Workshop on Artificial Intelligence and Security*, 2023.

[50] Adam Oest, Yeganeh Safaei, Penghui Zhang, Brad Wardman, Kevin Tyers, Yan Shoshitaishvili, and Adam Doupé. Phishtime: Continuous longitudinal measurement of the effectiveness of anti-phishing blacklists. In *Proc. of the USENIX Security Symposium*, 2020.

[51] Adam Oest, Penghui Zhang, Brad Wardman, Eric Nunes, Jakub Burgis, Ali Zand, Kurt Thomas, Adam Doupé, and Gail-Joon Ahn. Sunrise to sunset: Analyzing the end-to-end life cycle and effectiveness of phishing attacks at scale. In *Proc. of the USENIX Security Symposium*, 2020.

[52] Chidimma Opara, Bo Wei, and Yingke Chen. Htmlphish: Enabling phishing web page detection by applying deep learning techniques on html analysis. In *Proc. of the International Joint Conference on Neural Networks*, 2020.

[53] Thomas Kobber Panum, Kaspar Hageman, René Rydhof Hansen, and Jens Myrup Pedersen. Towards adversarial phishing detection. In *Proc. of the USENIX Workshop on Cyber Security Experimentation and Test*, 2020.

[54] Routhu Srinivasa Rao and Syed Taqi Ali. A computer vision technique to detect phishing attacks. In *Proc. of the International Conference on Communication Systems and Network Technologies*, 2015.

[55] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *Proc. of the Advances in Neural Information Processing Systems*, 2015.

[56] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[57] Stefan Romberg and Rainer Lienhart. Bundle min-hashing for logo recognition. In *Proc. of the ACM Conference on International Conference on Multimedia Retrieval*, 2013.

[58] Google Safe Browsing. https://safebrowsing.google.com/. (Accessed on 10/30/2023).

[59] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proc. of the IEEE Conference on Computer Vision and Pattern Pecognition*, 2015.

[60] Selenium. https://www.selenium.dev/documentation/. (Accessed on 04/12/2024).

[61] Rishab Sharma and Anirudha Vishvakarma. Retrieving similar e-commerce images using deep learning. *arXiv:1901.03546*, 2019.

[62] Karthika Subramani, William Melicher, Oleksii Starov, Phani Vadrevu, and Roberto Perdisci. Phishinpatterns: Measuring elicited user interactions at scale on phishing websites. In *Proc. of the ACM Internet Measurement Conference*, 2022.

[63] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv:1312.6199*, 2013.

[64] Saravanan Thirumuruganathan, Mohamed Nabeel, Eujin Choo, Issa Khalil, and Ting Yu. Siraj: A unified framework for aggregation of malicious entity detectors. In *Proc. of the IEEE Symposium on Security and Privacy*, 2022.

[65] Bram van Dooremaal, Pavlo Burda, Luca Allodi, and Nicola Zannone. Combining text and visual features to improve the identification of cloned webpages for early phishing detection. In *Proc. of the International Conference on Availability, Reliability and Security*, 2021.

[66] Ostap Viniavskyi, Mariia Dobko, Dmytro Mishkin, and Oles Dobosevych. Openglue: Open source graph neural net based pipeline for image matching. *arXiv:2204.08870*, 2022.

[67] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 2004.

[68] Liu Wenyin, Guanglin Huang, Liu Xiaoyue, Zhang Min, and Xiaotie Deng. Detection of phishing webpages based on visual similarity. In *Proc. of the Special Interest Tracks and Posters of the International Conference on World Wide Web*, 2005.

[69] Suzanne Widup, Alex Pinto, David Hylender, Gabriel Bassett, and Philippe langlois. Verizon Data Breach Investigations Report, 2021.

[70] Jonathan Woodbridge, Hyrum S. Anderson, Anjum Ahuja, and Daniel Grant. Detecting homoglyph attacks with a siamese neural network. In *Proc. of the IEEE Security and Privacy Workshops*, 2018.

[71] Liang Wu, Chengquan Zhang, Jiaming Liu, Junyu Han, Jingtuo Liu, Errui Ding, and Xiang Bai. Editing text in the wild. In *Proc. of the ACM International Conference on Multimedia*, 2019.

[72] Qiangpeng Yang, Jun Huang, and Wei Lin. Swaptext: Image based texts transfer in scenes. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[73] Li Yuan, Tao Wang, Xiaopeng Zhang, Francis EH Tay, Zequn Jie, Wei Liu, and Jiashi Feng. Central similarity quantization for efficient image and video retrieval. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[74] Ying Yuan, Qingying Hao, Giovanni Apruzzese, Mauro Conti, and Gang Wang. Are adversarial phishing webpages a threat in reality? understanding the users' perception of adversarial webpages. In *Proc. of the ACM on Web Conference*, 2024.

[75] Haijun Zhang, Gang Liu, Tommy W. S. Chow, and Wenyin Liu. Textual and visual content-based anti-phishing: A bayesian approach. *IEEE Transactions on Neural Networks*, 2011.

[76] Penghui Zhang, Adam Oest, Haehyun Cho, Zhibo Sun, RC Johnson, Brad Wardman, Shaown Sarker, Alexandros Kapravelos, Tiffany Bao, Ruoyu Wang, et al. Crawlphish: Large-scale analysis of client-side cloaking techniques in phishing. In *Proc. of the IEEE Symposium on Security and Privacy*, 2021.

[77] Rasha Zieni, Luisa Massari, and Maria Carla Calzarossa. Phishing or not phishing? a survey on the detection of phishing websites. *IEEE Access*, 2023.

A Appendix

Specific references to the corresponding contents are specified in the following.

A.1 Model Summary

We conduct a comprehensive literature review of top conferences and highly cited papers from 2005 to 2023 to identify popular visual similarity-based models for phishing detection, as summarized in Table 10. The gray sharing in the table indicates the seven models selected for re-training and evaluation. Liu *et al*. [68] compare features like layout, colors, fonts, and image placement, while EMD [19] uses Earth Mover's Distance to measure visual similarity. Medvet *et al*. [48], CCH [14], and Goldphish [17] analyze discriminative key points, employ image hashing techniques, and leverage classifiers for phishing detection, respectively. More recent approaches like Phishpedia [41] combine text and visual content analysis, while OpenGlue [66], Bernabeu *et al*. [8], OSLD [7], Bhurtel *et al*. [9], and SeeTek [36] explore advanced deep learning techniques for image retrieval, logo recognition, and visual similarity assessment. In particular, the use of deep learning (DL) techniques since 2019 shows a growing trend for visual similarity-based phishing detection.

A.2 Selected Models

Based on the candidate papers, we carefully selected seven models in Table 13 with diverse architectures, input types, and detection methods to compare visual similarity-based phishing detection approaches comprehensively. Particularly, four of them take screenshots, URLs, and HTML as input, while three of them take screenshots as input.

A.3 FLOPs and Parameters Performance

We compare the FLOPs and parameters for key model components in Table 11. `DynaPhish`, `PhishIntention`, and `Phishpedia` have similar model structures, resulting in close parameters and FLOPs for detecting logos and siamese modules. We exclude the CRP locator and web interaction parts of `PhishIntention` and `DynaPhish`, as our URLs may not be alive now. Furthermore, the online search function of `DynaPhish` is not included in the calculation. `Involution` employs the same module with `Phishpedia` for logo cropping and thus shares the parameter size. `EMD` and `PhishZoo` are not taken into consideration because they do not use neural networks.

A.4 Failure Examples Categorization

To better understand the limitations of visual similarity-based phishing detection models, we analyzed the failure cases observed during our real-world evaluations in Section 5. We categorized these failure examples into four main categories:

(a) FGSM (b) PGD (c) CW

(d) ViT (e) Swin (f) SRNet

Figure 4: Perturbated Logos Cropped by Faster-RCNN.

(a) Rackspace brand target list example (b) Recent searched Rackspace logo (c) Misidentified target logo

Figure 5: Text Logo Case (Original logo and benign URL).

logo, popup, login form, and other related issues, as summarized in Table 12. Logo-related issues (L1-L22) encompass various manipulations and alterations to the logo, such as elimination and color replacement. These issues highlight models' challenges in accurately identifying and comparing logos under diverse visual variations. Popup-related issues (P1-P4) pose a significant threat. They involve the presence of pop-ups, advertisements, cookies, alerts, and other overlays on the screenshot. These elements can obstruct or confuse the visual analysis of the web pages, potentially leading to misclassifications by the phishing detection models and, consequently, to successful phishing attacks. Login form-related issues (F1-F5) include changes to the login form's text, color, language, font, and other website login forms as a phishing tactic. These variations in the login form's appearance and design can make it difficult for models to identify phishing attempts based on visual similarity alone accurately. Other manipulations (O1-O3) include adding extra text on the screenshot and some pages that are blocked.

A.5 Perturbated and SRNet

The perturbed and SRNet logo samples cropped from screenshots by Faster-RCNN are shown in Figure 4. Three white-box attacks, two black-box attacks, and SRNet methods are summarized as follows:

Fast Gradient Sign Method (FGSM). [20]: A white-box attack that perturbs the data in a single step by using an imperceptibly small vector, elements are equal to the sign of the gradient of the cost function with respect to the input.

Projected Gradient Descent (PGD). [47]: An iterative version of the FGM attack with a random start, which applies the perturbations multiple times to create a more effective adversarial example.

Carlini & Wagner (CW). [11]: A strong white-box attack that optimizes the perturbations to minimize the detection confidence of the target model. Constructing three new at-

Table 10: List of Visual Similarity-based Models (Y/N: open source code or not, Y*: reproduced by non-original authors).

Year	Model	Description	DL	Code	Data Source
2005	Liu *et al.* [68]	Compares features like layout, colors, fonts, and image placement of webpages for phishing detection	N	N	N
2006	EMD [19]	Uses Earth Mover's Distance to assess visual similarity of webpages for phishing detection	N	Y* [42]	N
2008	Medvet *et al.* [48]	Relies on visual similarity, comparing features like layout, colors, and overall webpage appearance, to detect phishing	N	N	PhishTank, Alexa
2009	CCH [14]	Employs discriminative keypoint features to distinguish phishing websites based on visual cues	N	N	N
2010	Goldphish [17]	Analyzes images for phishing detection, possibly using techniques like image recognition or text extraction from images	N	N	PhishTank
2011	Zhang *et al.* [75]	Combines textual and visual content analysis with a Bayesian approach for phishing detection	Y	N	N
2011	msDT [29]	Introduces a method for logo recognition (not phishing specific) based on triangulation	N	N	Flickr
2011	PhishZoo [3]	Analyzes visual appearance of webpages, likely using techniques to compare layout, colors, fonts, and images	N	Y* [42]	PhishTank, Alexa
2013	Chang *et al.* [12]	Focuses on website identity recognition, using techniques like domain name analysis or website structure comparison	N	N	PhishTank, Alexa
2013	Romberg *et al.* [57]	Proposes bundle min-hashing for logo recognition	N	Y	Flickr
2015	FaceNet [59]	Deep learning architecture for face recognition (not directly related to phishing)	Y	Y	LFW, YoutubeDB
2015	Rao *et al.* [54]	Presents a computer vision technique for phishing detection using visual similarity	N	N	PhishTank
2015	LOGO-Net [26]	Leverages deep learning for logo detection (not directly related to phishing)	Y	N	N
2016	Bozkir *et al.* [10]	Uses HOG descriptors for feature extraction to potentially compare webpages for phishing detection	N	N	N
2017	DeltaPhish [15]	Compares the static features of HTML and visual appearance of the potential phishing pages against compromised websites	N	N	PhishTank
2017	Haruta *et al.* [23]	Combines image and CSS analysis with a target website finder for phishing detection	N	N	Alexa
2019	Sharma *et al.* [61]	Deep learning approach for image retrieval (adaptable to phishing)	Y	Y	N
2020	CSQ [73]	Deep learning method for image/video retrieval (adaptable to phishing)	Y	Y	ImageNet
2020	VisualPhishNet [1]	Proposes zero-day phishing website detection based on visual similarity	Y	Y	N
2021	Involution [37]	Inverting the inherence of convolution for visual recognition	Y	Y	Cityscapes
2021	Dooremaal *et al.* [65]	Combining text and visual features to improve the identification of cloned webpages for early phishing detection	Y	N	Y
2021	Phishpedia [41]	Employs a hybrid deep learning approach for visual phishing detection	Y	Y	Phishpedia
2022	PhishIntention [43]	Uses deep learning to analyze webpage appearance and dynamics for inferring phishing intention	Y	Y	PhishIntention
2022	OpenGlue [66]	Open-source deep learning pipeline for image matching (not directly related to phishing)	Y	Y	MegaDepth
2022	Bernabeu *et al.* [8]	Leverages deep learning for multi-label logo recognition (not directly related to phishing)	Y	N	METU
2022	OSLD [7]	Deep learning approach for large-scale logo detection (not directly related to phishing)	Y	N	OSLD
2022	Bhurtel *et al.* [9]	Relies on machine learning with a Siamese network for logo recognition for phishing detection	Y	N	LogoSENSE
2022	SeeTek [36]	Deep learning for large-scale logo recognition with text integration (not directly related to phishing)	Y	N	PL8K
2023	DynaPhish [44]	Using deep learning approaches to analyze webpages and Google search to identify brand intention	Y	Y	DynaPhish

Where deeper shades of indicate the seven models that we select for retraining and evaluation.

Table 11: Components' FLOP and Parameters Performance.

Model	Parameters/FLOPs				
	Detect Logo	Siamese	CRP Classifier	Others	Total
DynaPhish	41.32M/203G	24.10M/1.35G	23.50M/11.31G	—	88.92M/215.66G
PhishIntention	41.32M/203G	24.10M/1.35G	23.50M/11.31G	—	88.92M/215.66G
Phishpedia	41.30M/211G	24.10M/1.35G	—	—	65.40M/212.35G
Involution	41.30M/211G	—	—	12.01M/1.67G	53.04M/212.67G
VisualPhishNet	—	—	—	21.27M/92.49G	21.27M/92.49G

tacks for the three distance metrics.

[35]-**ViT** and [35]-**Swin.** Black-box attacks that utilize generative adversarial perturbations to develop adversarial logos. Taking the trained Vision Transformer (ViT) [16] and Swin Transformer [46] models as Discriminators and a Deep Residual Network with six residual blocks (ResNet-6) as the foundational architecture of the Generator.

Style Retention Network (SRNet). [71]: A generative adversarial network (GAN) that aims to transfer the style of images to another while preserving the content.

A.6 Examples of Visible Manipulation

Focusing on the logo component, we randomly select the samples that can make models fail. We also selected the samples based on the manipulations used by the adversaries. Detailed examples can refer to Figure 6. Additionally, Figure 5 shows an example that `PhishIntention` correctly identified but `Phishpedia` failed to recognize.

Table 12: Failure Categorization in Our Dataset.

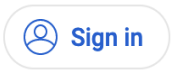
	ID	Name	Description
Logo	L1	Similar	Similar to the reference list
	L2	Elimination	Screenshots delete logos
	L3	BrokenImage	Logo images are damaged
	L4	ColorReplace	Different colors of logos
	L5	LogoBackground	Different backgrounds of logos
	L6	Integration	Logos are combined with other logos
	L7	Re-position	Logos appear in different locations on the screenshot
	L8	Outdated	Logos are not in the reference list
	L9	CaseConversion	Changing the case of textual logos
	L10	TextAsLogo	Type text as the logo
	L11	Scaling	Enlarge or shrink logos
	L12	Resizing	Logos' height-to-width are changed
	L13	FontReplace	Changing the textual font of logos
	L14	Omission	Only partial logos are used
	L15	Shape	Logo with different shapes, like square, rectangular
	L16	LogoAddText	Add text close to the logos
	L17	Replacement	Screenshots replace logos with other logos
	L18	Rotation	Logos are rotated in some angles
	L19	Flipping	Flipping logo by vertical or horizontal
	L20	Blurring	The logo or screenshot is blurred
	L21	CraftLogo	Craft logos based on different information
	L22	Language	Change the textual logos language
Popup	P1	LoginPopup	Login forms pop-up on the screenshot
	P2	AdPopup	Advertisements pop-up on the screenshot
	P3	CookiePopup	The cookie pop up on the blurred screenshot
	P4	OtherPopup	Alert, remind, location, etc.
Login	F1	LoginForm	Change login form text (text, color, language, fonts)
	F2	Button	Change button color, shape, location, text, etc.
	F3	NewForm	Design a new form
	F4	ThirdParty	Use other websites as login methods
	F5	QR	Login by scanning the QR code
Others	O1	ImageAddText	Add text on the screenshot not close to logo areas
	O2	Blocked	The image is blocked, only left text
	O3	ImageBackground	Different backgrounds of screenshots

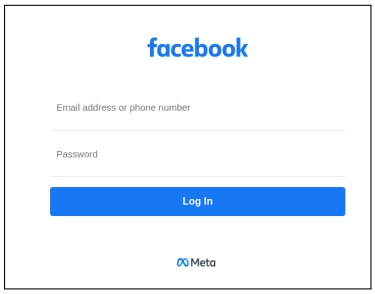
Table 13: Description of Used Seven Model Information.

Model Name	Training Dataset	Input	Description
EMD [19]	—	S	Calculate distance by EMD through color and coordinate feature
PhishZoo [3]	—	S, U, H	Use TF-IDF on URL and HTML for profile matching and use the SIFT feature for image matching
VisualPhishNet [1]	$\boldsymbol{R}_{ext}$	S	Use Triplet CNN to learn similarities of the same websites' screenshots and dissimilarities between different websites' screenshots.
Involution [37]	Logo2K+, $\boldsymbol{R}_{base}$ or $\boldsymbol{R}_{ext}$	S	Use Faster-RCNN to find the logo region, learn logo representations through Involution, and then compare cosine similarity
Phishpedia [41]	Logo2K+, Benign30K, $\boldsymbol{R}_{base}$ or $\boldsymbol{R}_{ext}$	S, U, H	Contains a layout classifier designed to detect and locate the logo region within images, and a Siamese neural network model that analyzes the identified logo to recognize and classify the brand it represents
PhishIntention [43]	Logo2K+, Sampled Benign30K, $\boldsymbol{R}_{base}$ or $\boldsymbol{R}_{ext}$	S, U, H	Contains a layout classier part to find the different components' regions, a CRP classifier to check if the screenshot has CRP, an HTML static classifier to check whether have CRP, a CRP locator to find additional links' CRP, and a Siamese model to recognize the logo's brand
DynaPhish [44]	—	S, U, H	Based on [43] and [41], it contains a Google search part to check targeted brands and dynamically expand reference lists

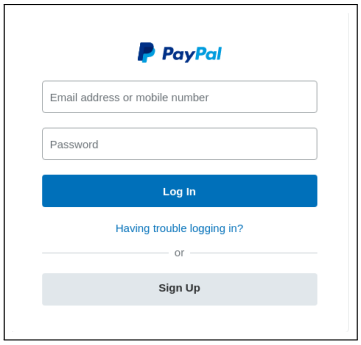
*__Testing Dataset__ = APWG Dataset, Manipulating Dataset; **__Brand Reference List__ = Baseline Ref. $\boldsymbol{D}_{base}$, Extended Ref. $\boldsymbol{D}_{ext}$;
S = Screenshot; **U** = URL; **H** = HTML.

Table 14: Example and Description of Visible Manipulation Methods.

Example	Method Description	Example	Method Description
	**Original:** This is the original logo cropped from the "YouTube" original website.		**Flipping:** We flip the logo vertically or horizontally. It differs from "Rotation," where we control the rotation to a small degree.
	**Color Replacement:** We identify the logo in the screenshot and then replace the color. In this example, we change the original red to blue, but the attacker could use any other predefined color.		**Resizing:** We randomly modify the height-to-width ratio of the logo. Note that logo resizing does not necessarily maintain the proportion.
	**Rotation:** We rotate the logo in small increments clockwise or counterclockwise, and fill the empty area created by the rotation with the color of the surrounding background. In this example, it is rotated clockwise by one degree.		**Integration:** We randomly select a second logo from a set of 110 different target brands and place it either above, below, or to the left of the original logo in the screenshot. For example, the "YouTube" is combined with "Spark NZ."
	**Replacement:** We replace the original logo with a logo randomly selected from 110 brands. For example, the login form is still "YouTube," but the logo is replaced with "Raiffeisen Bank."		**Scaling:** We scale up the logo, increasing both the length and width to 1.1 times the original size. Then, we place the resized logo in the screenshot of "Elimination."
	**Elimination:** We remove the logo from the screenshot and fill the area with the surrounding background color. The region detector may identify other components ("sign in") as the logo.		**Blurring:** We add Gaussian blurring with kernel size 9 to the entire screenshot image, including the logo and the background, by the "OpenCV" Python package.
	**Re-position:** We move the position of the logo horizontally within the screenshot and fill it with the surrounding background color. The example is cropped from the screenshot when the logo is moved from the top left to the bottom left.		**Omission:** We use only one of the elements of the logo (either icon or text) and fill the rest with the surrounding color. For example, we keep the icon and remove the text "YouTube."
	**Font Replacement:** We use a font identification tool (https://www.myfonts.com/pages/whatthefont) to find similar fonts. Then, we generate text in those fonts and replace the original logo. We also use the SRNet [71] to generate text logos while keeping the background context, font style, and color.		**Case Conversion:** We find a font that looks similar to the text logo and then change the capitalization of the text to make all letters capitalized, all letters lowercase, or just the first letter capitalized. For example, "YouTube" is transformed into "YOUTUBE."



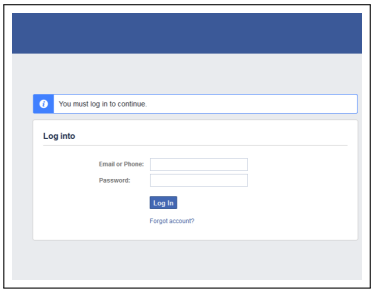
(a) EMD Failed Example



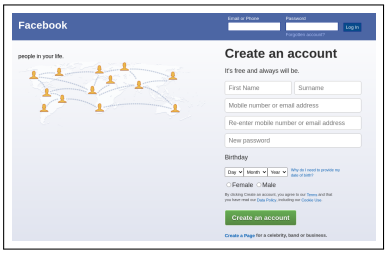
(b) VisualPhishNet Failed Example

(c) Wrong Logo Area and QR Code

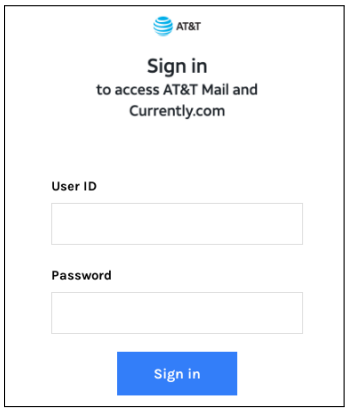
(d) Similar (<Threshold)



(e) Elimination of Logo

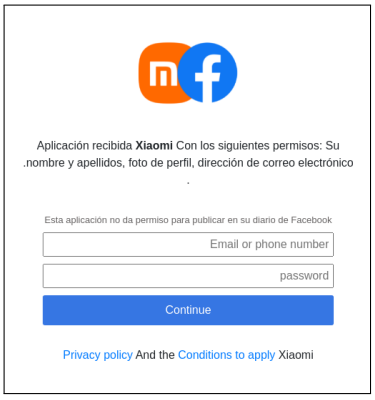


(f) First Letter Upper Case Conversion

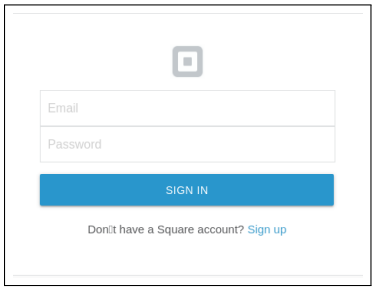


(g) PhishZoo Failed Example

(h) Adding Text

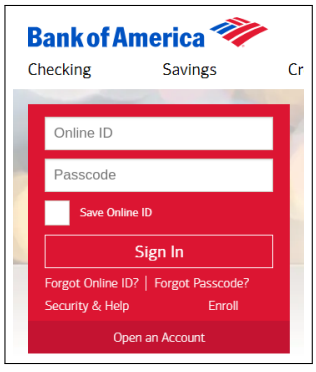


(i) Integration of Logos



(j) Omission and Color Replacement

(k) Font Replacement



(l) Case Conversion and Outdated

Figure 6: Examples of Manipulated Samples Found in Our Real-world Phishing Dataset.