applied sciences

MDPI

Article

Phishing Website Impersonation: Comparative Analysis of Detection and Target Recognition Methods

Marcin Jarczewski [1], Piotr Białczak [2,*] and Wojciech Mazurczyk [1]

[1] Institute of Computer Science, Warsaw University of Technology, Nowowiejska 15/19, 00-665 Warsaw, Poland; marcin.jarczewski.stud@pw.edu.pl (M.J.); wojciech.mazurczyk@pw.edu.pl (W.M.)
[2] CERT Polska/NASK—National Research Institute, Kolska 12, 01-045 Warsaw, Poland
* Correspondence: piotr.bialczak@cert.pl

Abstract

With the rapid advancements in technology, there has been a noticeable increase in phishing attacks that exploit users by impersonating trusted entities. The primary attack vectors include fraudulent websites and carefully crafted emails. Early detection of such threats enables the more effective blocking of malicious sites and timely user warnings. One of the key elements in phishing detection is identifying the entity being impersonated. In this article, we conduct a comparative analysis of methods for detecting phishing websites that rely on website screenshots and recognizing their impersonation targets. The two main research objectives include binary phishing detection to identify malicious intent and multiclass classification of impersonated targets to enable specific incident response and brand protection. Three approaches are compared: two state-of-the-art methods, Phishpedia and VisualPhishNet, and a third, proposed in this work, which uses perceptual hash similarity as a baseline. To ensure consistent evaluation conditions, a dedicated framework was developed for the study and shared with the community via GitHub. The obtained results indicate that Phishpedia and the Baseline method were the most effective in terms of detection performance, outperforming VisualPhishNet. Specifically, the proposed Baseline method achieved an F1 score of 0.95 on the Phishpedia dataset for binary classification, while Phishpedia maintained a high Identification Rate (>0.9) across all tested datasets. In contrast, VisualPhishNet struggled with dataset variability, achieving an F1 score of only 0.17 on the same benchmark. Moreover, as our proposed Baseline method demonstrated superior stability and binary classification performance, it should be considered as a robust candidate for preliminary filtering in hybrid systems.

Keywords: phishing; phishing detection; machine learning; cybersecurity

Check for updates

Received: 9 December 2025
Revised: 29 December 2025
Accepted: 30 December 2025
Published: 7 January 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

The evolution of the Internet has fundamentally transformed global communication and commerce. As the network expanded and gained in popularity, it became an integral part of the modern economy. As of 2025, estimates indicate that 72% of companies maintained their own websites, with approximately 27% of sales conducted online [1]. However, this rapid digitization has inevitably attracted criminal activity. As the user base has grown, so too has the prevalence of cybercrime. Among the most persistent and damaging categories of cybercrime is phishing [2]. This form of fraud involves attackers impersonating trusted entities, such as financial institutions, delivery companies, or government bodies, through carefully crafted emails or websites designed to deceive users

into revealing sensitive data, including login credentials or financial information. The scale of this threat is escalating rapidly. According to the Anti-Phishing Working Group, the number of observed phishing attacks reached 1,130,393 in the second quarter of 2025, rising from 1,003,924 in the first quarter of 2025 [3]. Phishing attacks are diverse in their delivery, utilizing emails, SMS (smishing), and voice calls (vishing) to reach potential victims [4]. Defending against these attacks requires more than just passive blocking lists [5,6]. It demands proactive, automated detection capabilities [7]. A critical aspect of mitigation is identifying the impersonation target, i.e., the specific brand or organization the attacker is mimicking [8,9]. Correctly identifying the target enables security teams to notify the affected institution and its users, thereby minimizing the impact of the campaign. Traditional detection methods have often relied on analyzing the source code or URL structures of suspicious sites [6]. However, attackers increasingly obfuscate code to evade detection while ensuring the visual rendering of the site remains visually indistinguishable for users [10]. Consequently, this research shifts the focus toward visual analysis, treating the webpage screenshot as the primary data source for classification [11–13]. This approach posits that while the underlying code may change, the visual representation must remain consistent with the impersonated brand to successfully deceive the victim. This paper presents a comparative analysis of visual-based phishing detection and target recognition methods, conducted in collaboration with Computer Emergency Response Team (CERT) Polska, the Polish national Computer Security Incident Response Team (CSIRT). We evaluate three distinct approaches: two state-of-the-art deep learning methods from the recent literature, Phishpedia [13] and VisualPhishNet [11], and a third Baseline approach proposed in this work based on perceptual hashing. To facilitate a fair and reproducible evaluation, we designed and implemented a modular experimental environment using Docker and Python 3.9. The study utilizes datasets provided by the authors of the respective methods as well as a new, real-world dataset collected by CERT Polska analysts. Considering the above, the main contributions of this work are as follows:

- To allow a fair comparison of the selected phishing detection methods, we design and develop a dedicated framework that is easily extendable to incorporate other future solutions and datasets as they emerge. This framework is available to the community at GitHub (https://github.com/Percival33/phish-target-recognition (accessed on 15 December 2025)).
- We perform a thorough comparative analysis of phishing website detection methods that rely on website screenshots and enable the recognition of their impersonation targets. The analysis is conducted on three datasets for two state-of-the-art methods (Phishpedia and VisualPhishNet), and a third solution (a Baseline solution), which uses perceptual hash similarity.
- We demonstrate that the proposed Baseline method achieves superior stability and binary classification performance compared to complex deep learning models, particularly on diverse or synthetically augmented datasets.

The remainder of this paper is organized as follows: Section 2 reviews related work in phishing detection. Section 3 outlines the fundamental concepts of visual analysis and the specific algorithms used. Section 4 details the experimental framework and dataset preparation. Section 5 presents the comparative results of the experiments. Section 6 concludes with a summary and directions for future research.

2. Related Work

Phishing detection has evolved from simple list-based approaches to sophisticated machine learning systems. In this section, we review the existing landscape of phishing detection, with a specific focus on the visual identification of impersonation targets. Fur-

thermore, we address the critical gap in current research regarding the reproducibility of results and the lack of standardized comparison frameworks.

2.1. State-of-the-Art in Phishing Detection and Target Identification

The evolution of phishing detection methodologies can be categorized into three distinct generations: heuristic code-based analysis, visual feature descriptor methods, and contemporary deep learning approaches. This section synthesizes these developments to contextualize the methods evaluated in this study.

2.1.1. Heuristic and Feature-Based Approaches

Early phishing detection primarily relied on the analysis of textual and structural features, such as URL composition, HTML source code, and network characteristics [4,6]. While computationally efficient, these methods proved vulnerable to obfuscation techniques where attackers hide malicious payloads via JavaScript or dynamic content generation [10].

To address the limitations of text-based analysis, research shifted toward visual similarity. Initial attempts in this domain treated websites as images to bypass code obfuscation. These methods utilized classical computer vision techniques, such as the Earth Mover's Distance (EMD) to compare color histograms [14], or local feature descriptors like SIFT [15] and DAISY [16] to construct "bag-of-words" representations. Although these approaches improved robustness against code manipulation, they often struggled with the high computational cost required for real-time processing and lacked the generalization capabilities needed to handle the visual diversity of modern web designs.

2.1.2. Deep Learning-Based Visual Analysis

The advent of deep learning has fundamentally shifted the landscape of visual phishing detection, moving from handcrafted features to automated feature extraction. Recent studies demonstrate that deep neural networks can capture complex visual hierarchies, though they introduce new challenges regarding feature selection and model robustness. As highlighted by Saheed et al. [17] in the context of intrusion detection systems, the selection of robust features is critical for maintaining high detection accuracy while minimizing computational overhead—a principle that applies directly to the visual embeddings used in phishing detection.

Current state-of-the-art visual approaches generally fall into two categories: hybrid systems and pure visual classifiers. Hybrid approaches, such as the work by Chen et al. [18], combine visual analysis (via CNNs) with textual features extracted from URLs and OCR data. Similarly, Bhurtel et al. [19] proposed a system integrating logo recognition via Siamese networks with IP address mapping verification. Other works, such as LogoMotive [20], frame the problem strictly as an object detection task, utilizing YOLO architectures to identify high-risk logos.

In this study, we focus specifically on comparing two prominent, distinct deep learning philosophies: holistic layout analysis versus targeted object detection.

VisualPhishNet: Proposed by Abdelnabi et al. [11], this method represents the holistic approach. It employs a triplet Convolutional Neural Network (CNN) based on a VGG-16 backbone to learn a universal representation of website layouts. The core assumption is that different pages belonging to the same entity (e.g., a login page and a homepage) share a visual "feel" that can be grouped in an embedding space. The network is trained using a triplet loss function, which minimizes the Euclidean distance between an anchor and a positive sample while maximizing the distance to a negative sample. A key aspect of

this method is its reliance on "hard negative mining" during training to refine the decision boundaries between visually similar but distinct brands.

Phishpedia: In contrast to the layout-based approach, Phishpedia [13], developed by Lin et al., frames the problem as a high-precision object detection and recognition task. The system utilizes a two-stage pipeline: first, a Faster R-CNN model extracts potential logo regions from the screenshot; second, a Siamese network identifies the specific brand by comparing the extracted region against a reference database. Unlike VisualPhishNet, Phishpedia utilizes transfer learning on the Logo-2K+ dataset and avoids standard triplet loss in favor of a classification-based fine-tuning approach. This design choice aims to handle visual variations of specific logos (e.g., "Adobe" vs. "Adobe AIR") more effectively than holistic layout matching.

2.2. The Challenge of Reproducibility and Evaluation Frameworks

A significant challenge in the field of phishing detection is the lack of standardized evaluation benchmarks. Most novel methods are evaluated on ad-hoc datasets collected by the authors, which often become unavailable or outdated due to the short lifespan of phishing URLs. This makes direct comparison of different architectures, such as comparing the holistic approach of VisualPhishNet against the logo-based approach of Phishpedia, difficult and potentially biased. The importance of reproducibility in cybersecurity research cannot be overstated. Without a shared framework and unified datasets, it is impossible to verify whether reported performance gains are due to architectural improvements or simply artifacts of a specific data distribution. While frameworks for comparing general malware detection exist, there is a notable scarcity of open-source environments specifically designed for the fair, side-by-side evaluation of visual phishing detection methods that handle the complexities of conflicting software dependencies and diverse input formats. Some works try to address these issues by providing some benchmarking and comparative evaluation elements for phishing detection methods. V. Zeng et al. in [21] introduced PhishBench 2.0, a framework for benchmarking phishing detection methods. It is a second version of the framework that provides means for testing various email and URL datasets. The framework has built-in model feature sets, classifiers, and metrics; however, they can be extended by additional ones defined by the user. This modular architecture is similar to ours; however, the authors focused on a text-based input dataset, while we analyze the visual form of phishing webpages. A. Hannousse et al. in [22] created a benchmark dataset for phishing website detection systems that use machine learning. The dataset comprises a predefined feature set based on URL, the website's content, and data from external services. The authors evaluated features and classifiers based on the dataset they created. This approach focuses on creating a versatile benchmarking dataset based on textual data, whereas our approach evaluates multiple visual-based classifiers across multiple datasets. In [23] T. Dalton et al. presented a high quality phishing website dataset along with benchmark datasets. The datasets consist of URLs and HTML of websites. The benchmark datasets are created based on the main dataset, introducing additional challenges to effectively evaluate phishing detection methods. The authors also analyzed 4 models using their datasets to provide baseline results. While our approach also aims to benchmark phishing detection systems, we focus on the visual aspects of phishing websites, whereas the authors focus on text-based aspects. Additionally, we provide a framework for performing comparisons on any dataset provided by the user.

F. Ji et al. in [24] present a comparison of various visual phishing detection methods. The authors utilize a dataset of 451,000 phishing websites to evaluate the effectiveness of detection and the capability of target identification. Additionally, the authors analyzed the models' robustness against adversarial attacks. This work is similar to ours; however,

our approach focuses on ease of extensibility to additional methods, whereas the authors' system does not provide any means for this. In this work, we address the gaps identified in previous research by introducing a modular framework that integrates these distinct methodologies, allowing for their evaluation on a shared, stratified dataset comprising both public academic data and real-world samples from CERT Polska.

3. Fundamentals

To effectively analyze the proposed phishing detection frameworks, it is necessary to establish the theoretical underpinnings of web identity, visual similarity processing, and the specific neural architectures employed in these frameworks. This section outlines the mechanics of perceptual hashing and the principles of deep learning applied to image recognition. A URL allows for the location of a resource and consists primarily of the protocol, the Fully Qualified Domain Name (FQDN), and the path to the specific resource. The FQDN further decomposes into the Registered Domain Name (RDN) and subdomains. The RDN, which includes the main level domain (MLD) and the public suffix, is a critical component for identifying the entity that owns the website. Phishing attacks frequently manipulate the FreeURL (subdomains and paths) or mimic the MLD to deceive users, making the correct identification of the registered domain essential for determining the target of impersonation.

3.1. Visual Hashing and Descriptors

Visual analysis relies on transforming images into comparable data structures. A core technique employed in our Baseline approach is Locality-Sensitive Hashing (LSH). Unlike cryptographic hash functions (e.g., MD5, SHA-256), which are designed to produce vastly different outputs for minor input changes, LSH aims to maximize the probability that similar items map to the same hash buckets. Specifically, we utilize perceptual hashing (pHash), which is robust against minor modifications such as compression or resizing (as seen in [25]). This technique often employs the Discrete Cosine Transform (DCT) to convert the image into a frequency domain, retaining the low-frequency components that represent the image's structure while discarding high-frequency noise. Additionally, visual descriptors, such as DAISY, enable the dense extraction of gradient features from an image, allowing for a "bag of words" representation where images are compared based on the frequency of specific visual vocabularies. For more complex visual recognition tasks, such as those performed by VisualPhishNet and Phishpedia, Convolutional Neural Networks (CNNs) are the standard architecture. CNNs process data with a grid-like topology using convolutional layers that apply filters to extract spatial hierarchies of features, from simple edges to complex patterns. In the context of phishing detection, full-scale CNN architectures (e.g., VGG-16 or ResNet) are often utilized as feature extractors to generate embeddings—dense vector representations of the website screenshots. To compare these embeddings effectively, we employ Siamese networks. This architecture consists of two or more identical subnetworks sharing the same weights, which process distinct inputs to map them into a common feature space. The objective is to ensure that embeddings of the same class (e.g., two screenshots of the PayPal login page) are geometrically closer to each other than embeddings of different classes. The training of such networks often involves *Triplet Loss*. This function takes three inputs: an anchor (a reference image), a positive (a similar image), and a negative (a dissimilar image). The loss function minimizes the distance between the anchor and the positive while simultaneously maximizing the distance between the anchor and the negative by a defined margin. As noted in the literature, the selection of "semi-hard" negatives—examples that are difficult but not impossible to distinguish—is crucial for stable model convergence. EMD calculates

the minimum cost required to transform one distribution into another. While effective for image retrieval, EMD has a high computational complexity of $\Theta(n^3 \log n)$, which often necessitates the use of approximations or alternative metrics for real-time applications.

3.2. Evaluation Metrics

The assessment of phishing detection systems presents unique challenges due to two primary factors: the significant class imbalance often inherent in cybersecurity datasets and the dual nature of the problem (binary detection vs. multiclass target recognition). Relying solely on simple accuracy can be misleading; a trivial classifier that labels all samples as "benign" could achieve high accuracy in a dataset where phishing attacks are rare, despite failing to detect a single threat. To address these challenges and provide a comprehensive performance assessment, we employed a diverse set of metrics. For binary classification (determining whether a site is phishing), we utilized the F1 Score and the Matthews Correlation Coefficient (MCC) to ensure robustness against class imbalance. The MCC is particularly important in this context as it considers all four quadrants of the confusion matrix, providing a balanced measure even when classes are of very different sizes, unlike the F1 score which ignores true negatives. For the multiclass task of impersonation target recognition, we adopted the Identification Rate to measure the precision of brand attribution.

3.2.1. Binary Classification Metrics

The foundation of our evaluation is the confusion matrix, which categorizes predictions into True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN).

F1 Score: The F1 score is the harmonic mean of Precision and Recall. It is particularly useful when the cost of False Negatives (missing a phishing attack) and False Positives (blocking a legitimate user) requires a balanced view. It is defined as:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2TP}{2TP + FP + FN} \tag{1}$$

Matthews Correlation Coefficient (MCC): The MCC is widely regarded as one of the most robust metrics for binary classification on imbalanced datasets. Unlike the F1 score, which ignores True Negatives, MCC incorporates all four quadrants of the confusion matrix. It returns a value between -1 and +1, where +1 represents a perfect prediction, 0 indicates no better than random guessing, and -1 indicates total disagreement between prediction and observation. It is defined as:

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{2}$$

ROC AUC (Receiver Operating Characteristic Area Under the Curve): The methods evaluated in this study, particularly VisualPhishNet and the Baseline, rely on calculating a distance metric that is compared against a threshold to determine the final classification. To evaluate the performance of these classifiers across all possible classification thresholds, rather than a single, fixed cut-off point, we employ the ROC AUC metric. The Receiver Operating Characteristic (ROC) curve plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings:

$$TPR = \frac{TP}{TP + FN} \tag{3}$$

$$FPR = \frac{FP}{FP + TN} \tag{4}$$

On the other hand, the Area Under the Curve (AUC) summarizes the ROC curve into a single scalar value representing the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one. An AUC of 0.5 suggests no discriminatory power (random guessing), while an AUC of 1.0 indicates a perfect classifier. This metric is crucial for determining how well the model separates the distributions of benign and phishing sites before a specific decision threshold is applied.

3.2.2. Target Recognition Metrics

For the multiclass problem of identifying the specific brand being impersonated (e.g., distinguishing a fake PayPal site from a fake Microsoft site), it is necessary to aggregate the scores across all potential target classes (N classes). We employed several distinct averaging strategies to provide a holistic view of model performance: Micro/Macro-averaged F1 and Identification Rate.

Macro-averaged F1: We utilize Macro-averaged F1, which calculates the F1 score independently for each class and then takes the unweighted mean. This metric treats all classes equally, regardless of their frequency. In the context of phishing, this is crucial for ensuring that the system protects smaller, less frequently attacked brands just as effectively as major global targets.

$$F1_{macro} = \frac{1}{N}\sum_{i=1}^{N} F1_i = \frac{1}{N}\sum_{i=1}^{N} \frac{2TP_i}{2TP_i + FP_i + FN_i} \tag{5}$$

Micro-averaged F1: Complementing the macro approach, we employ the Micro-averaged F1. Unlike macro-averaging, this metric aggregates the contributions of all classes by summing the total True Positives, False Positives, and False Negatives across all N classes before calculating the score.

$$F1_{micro} = \frac{2\sum_{i=1}^{N} TP_i}{2\sum_{i=1}^{N} TP_i + \sum_{i=1}^{N} FP_i + \sum_{i=1}^{N} FN_i} \tag{6}$$

The rationale behind using two F1 metrics is as follows. Micro-averaged F1 effectively weights the result by class frequency. It provides a measure of the system's global effectiveness across the entire volume of traffic. While Macro F1 highlights performance on minority classes, Micro F1 reveals how well the system performs on the majority of incoming samples. In multiclass classification, where each sample is assigned exactly one exclusive label, the Micro F1 score is mathematically equivalent to overall Accuracy. Including both metrics allows us to detect if a model is biased towards high-volume targets at the expense of niche brands.

Identification Rate: To specifically evaluate the utility of the system in a real-world triage scenario, we utilized the Identification Rate. Proposed in Phishpedia [13], this metric measures the system's ability not only to detect a phishing page, but also to correctly identify the target. It is calculated as the ratio of correctly recognized targets (Id) to the total number of correctly detected phishing pages (Rep^{p}_{TP}), effectively acting as a recall metric for the specific sub-task of brand recognition. It is important to note that this is a conditional metric, i.e., it evaluates classification performance only on the subset of samples already correctly identified as phishing.

$$\text{Identification Rate} = \frac{Id}{Rep^{p}_{TP}} \tag{7}$$

4. Framework for Fair Comparison of Phishing Detection Methods

To address the fragmentation in current phishing detection research, where methods often rely on conflicting environments and disparate data formats, we designed and implemented a comprehensive evaluation framework. This system was engineered not only to host the detection algorithms but also to standardize the input data pipeline, ensuring that the comparative analysis described in Section 5 remains fair, reproducible, and extensible.

4.1. The Framework's Architecture and Design

The architecture of the framework is presented in Figure 1. Screenshots from datasets are mapped and split accordingly to evaluation protocol. Then they are forwarded to the API Gateway that acts as an orchestrator, distributing prediction requests with appropriate screenshots to isolated method containers. Prediction results are sent back by methods to the API Gateway, from where they are forwarded to the Persistance Layer for saving to database.

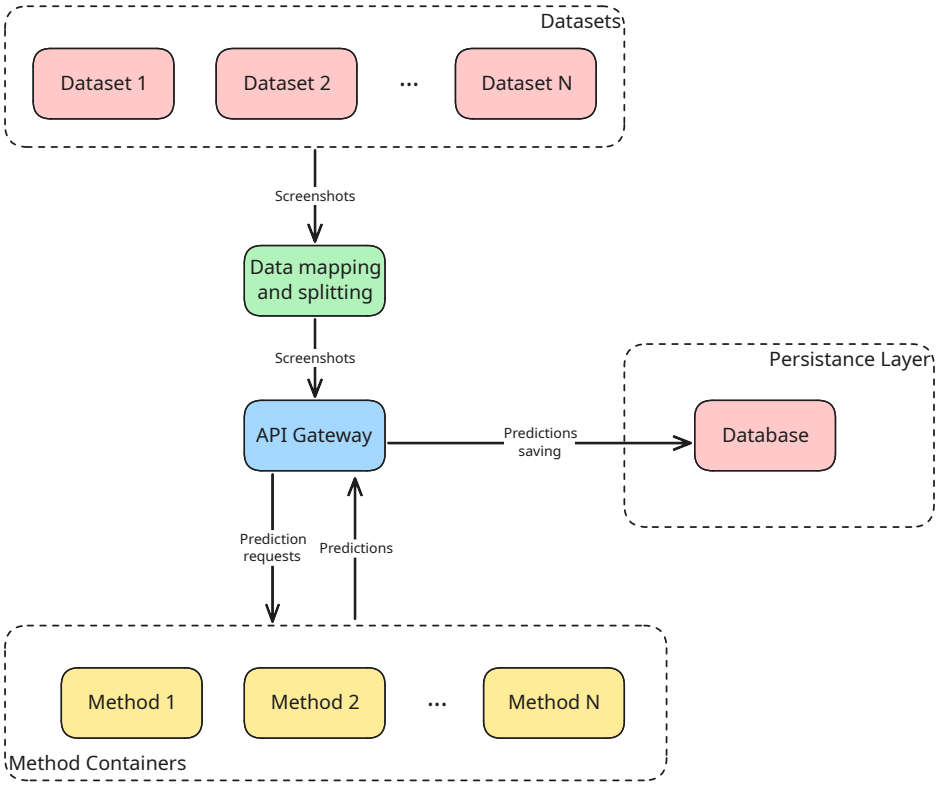


Figure 1. The framework's architecture diagram. The system follows a microservice pattern where the API Gateway acts as an orchestrator, distributing requests to isolated method containers (VisualPhishNet, Phishpedia, Baseline) and storing results in a unified persistence layer.

The framework adopts a microservices architecture based on containerization to resolve the "dependency hell" often encountered when comparing deep learning models built on different library versions. The system is orchestrated using Docker and Docker Compose, ensuring strict isolation between the detection methods. The architecture comprises three distinct layers: the API Gateway, the independent Model Containers, and the Persistence Layer. The central component is the API Gateway, implemented in Python using the FastAPI framework [26], which was chosen for its built-in data validation and automatic documentation generation capabilities. This gateway acts as the orchestrator; it accepts incoming requests containing Base64-encoded screenshots and target URLs, validates the payload using *pydantic* [27], and dispatches the data to the active model containers.

To ensure extensibility, we defined a rigid contract for all classifiers via an abstract base class. Every integrated method must implement specific lifecycle hooks, i.e., *on_startup* for loading heavy weights or indices, and predict for the inference logic, allowing the system to treat complex underlying models as interchangeable "black boxes". Results from the inference process are stored in an SQLite database, which is managed by the SQLAlchemy Object-Relational Mapper (ORM) [28]. This persistence layer captures not only the binary classification (phishing vs. benign) but also the multiclass target prediction and confidence scores, enabling detailed post-hoc analysis of model performance. The framework's three layers are extended with data mapping and splitting scripts. They help in standardizing the input datasets to be adapted to the analysis methods. They also provide means for data splitting, allowing for the creation of train/validation/test dataset splits.

4.2. Data Management and Standardization

A critical component of the framework is the data processing pipeline, which aggregates and standardizes datasets from different sources to support cross-validation. The study utilizes three primary datasets, processed via a dedicated *data_splitter* module to ensure consistency.

Given the different folder structures and labeling conventions of the datasets, we implemented mapping scripts to unify them into a single format. A major challenge in phishing detection is class imbalance. To address this, our framework employs stratified sampling based on the target brand classes to divide data into training (60%), validation (20%), and test (20%) sets, ensuring the class distribution remains consistent across splits. To provide a sufficient number of samples for validation and test dataset parts, we limited the datasets to classes that contain at least 25 samples. Furthermore, we integrated the *torchvision* library [29] to perform synthetic data augmentation in cases where the sample limitation would reject the majority of the dataset. We applied brightness, contrast, blur, and noise adjustments to increase the representation of minority classes in the datasets. Specifically, for the VP dataset, 78 benign images and 2537 phishing images were generated. For the PP dataset, 1231 benign images and 317 phishing images were added. These synthetic samples were necessary to meet the minimum class size requirements for valid training and testing. The three datasets and their adaptations are described below. Table 1 presents the number of samples in each dataset divided into training, validation, and test sets.

Table 1. Datasets used in this study and their division into training, validation, and test sets, and the corresponding number of targets.

Dataset	Training		Validation		Test		∑		No. of Targets
	Phish	Benign	Phish	Benign	Phish	Benign	Phish	Benign	
CERT	4326	4703	1442	1568	1442	1568	7210	7839	36
PP	8700	924	2900	309	2900	309	14,500	1542	56
VP	2786	5332	929	1751	929	1752	4644	8835	144

The VisualPhishNet (VP) dataset: This dataset originates from VisualPhishNet (VP) paper. The original VP dataset comprises 9363 benign images and 1195 phishing samples covering 155 brands, sourced from Alexa and PhishTank. To adapt the dataset to other methods than those presented in the original paper, a set product was calculated of its trusted list with the target list of the Phishpedia dataset. 144 classes remained, comprising 8835 benign screenshots and 2107 phishing samples. Note that the benign count includes 78 images added via augmentation to ensure sufficient training samples per class. To address the class imbalance, 2537 synthetic screenshots were added to the phishing subset.

The Phishpedia (PP) dataset: This dataset originates from the Phishpedia (PP) paper. The original dataset consisted of 29,496 phishing samples from OpenPhish and 29,951 benign websites from Alexa. To adapt the dataset to our needs, we created a set of products with targets that include logos and screenshots in the *benign_sample_30k* directory. We obtained 56 targets, with 14,183 phishing screenshots and 311 benign samples. To fulfill the minimum limit of 25 samples per class, we augmented the dataset with 317 phishing and 1231 benign samples.

The CERT Polska dataset: To evaluate performance on real-world, region-specific threats, we collaborated with CERT Polska to curate a dataset of 15,049 unique URLs and screenshots collected between June 2023 and June 2024. This dataset is of particularly high quality; the phishing examples were verified by analysts, and we filtered out low-quality screenshots (e.g., loading screens or error pages). Unlike the other two, this dataset was not augmented, but includes only classes with at least 25 samples of benign and 25 samples of phishing screenshots. Overall, it covers 36 distinct impersonation targets.

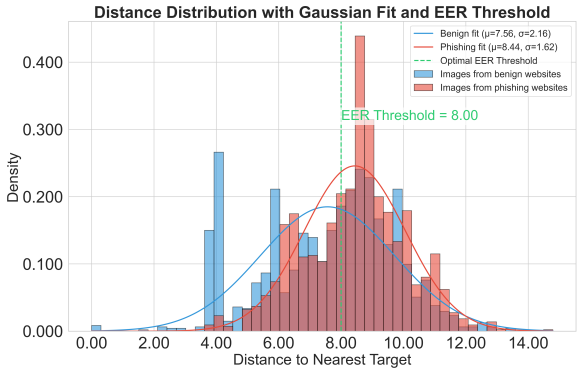
4.3. Adaptation and Implementation of Detection Methods

The framework integrates three distinct detection strategies. To ensure a fair evaluation, we refactored existing state-of-the-art implementations to fit our standardized interface and developed a Baseline method for benchmarking.

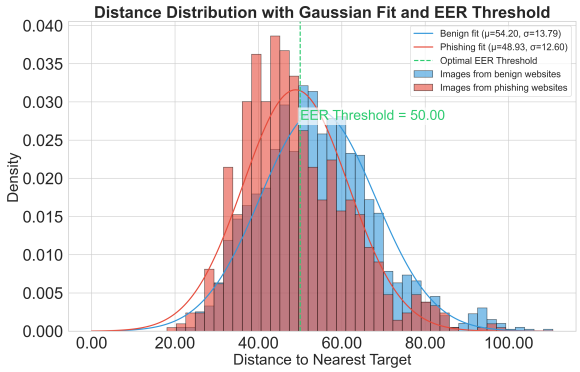
VisualPhishNet adaptation: The original implementation of VisualPhishNet was provided as a series of Jupyter Notebooks, which are unsuitable for automated pipelines. We refactored this code into executable Python scripts, adding command-line argument support and integrating the Weights & Biases platform for experiment tracking. This enables real-time monitoring of loss convergence and GPU resource utilization during the training of the triplet loss network. Training progress was monitored using Weights & Biases to ensure all models reached convergence and to trigger early stopping based on validation loss, preventing underfitting. The training phase was performed using implementation details provided by the VisualPhishNet's authors, including training protocol and sizes of mini-batches.

Threshold determination for VisualPhishNet: For the VisualPhishNet model, the classification decision relies on a distance-based metric: a suspicious website is classified as a phishing attempt if the distance between its embedding and the embedding of a protected target falls below a specific cutoff threshold. To ensure a faithful reproduction of the method, we adopted the threshold selection strategy described in the original VisualPhishNet paper [11], optimizing for the Equal Error Rate (EER) point where the difference between the False Positive Rate (FPR) and the False Negative Rate (FNR) is minimized. To locate this optimal point efficiently, the search was conducted in two stages: *(i) coarse search*—we analyzed potential threshold values ranging from 0 to the maximum distance observed in the validation set, incrementing with a step size of 10; *(ii) fine-grained search*—we refined the search within the range of the mean distance $\pm$ one standard deviation, utilizing a step size of 1. This process ensured that the binary classification boundary was tuned specifically to the data distribution of each dataset prior to final testing. The optimal threshold was computed independently for each of the three datasets using their respective validation partitions. The procedure involved calculating the distances between the embeddings of the validation samples and the training examples. The objective was to identify the Equal Error Rate (EER) point, defined as the threshold value where the difference between the False Positive Rate (FPR) and the False Negative Rate (FNR) is minimized. The optimal threshold for the CERT dataset was identified at 8.00, while for the VP dataset, it was significantly higher, determined at 50.00. Finally, for the PP dataset, the threshold was determined to be 3.00. We further analyzed the efficacy of these thresholds by examining the distribution of

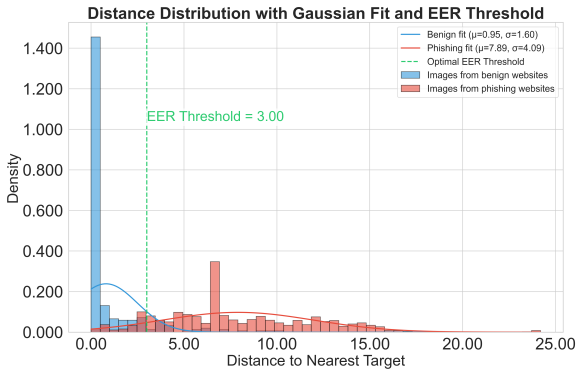
embedding distances between the phishing and legitimate validation sets during training, using the fitted Gaussian Probability Density Functions. The distributions are presented in the Figure 2.



(**a**)



(**b**)



(**c**)

Figure 2. Distance distribution of the minimum distances between the phishing and legitimate validation sets to the training. EER thresholds are annotated. (**a**) Distribution for the CERT Polska dataset. (**b**) Distribution for the VP dataset. (**c**) Distribution for the PP dataset.

A critical observation from this analysis is that the VP dataset exhibited a unique characteristic: it was the only case where the mean distance for benign images was larger than for phishing images. This inversion suggests that for the VP dataset, the model struggled to cluster visual similarity in a way that consistently distinguished the two classes, leading to the high threshold of 50 relative to the others.

Phishpedia integration: Phishpedia combines object detection (Faster R-CNN) with a Siamese network for logo matching. The original codebase lacked training scripts and

suffered from data inconsistencies, including missing mappings between protected brands and their corresponding domains. We extended the implementation to include data normalization scripts that repair these mappings and handle text encoding issues, ensuring the model could be evaluated on external datasets without runtime errors.

Baseline method (perceptual hashing): To establish a performance baseline, we implemented a lightweight visual similarity method. This approach utilizes the Discrete Cosine Transform (DCT) to generate perceptual hashes (pHash) and real-valued vector hashes (pHashF) of the website screenshots. These vectors are indexed using the FAISS library [30] for efficient similarity search. During inference, the system calculates the Euclidean or Hamming distance to the nearest neighbor in the training set; if the distance falls below a learned threshold, the site is classified as phishing, inheriting the target label of its neighbor. The optimal threshold for this method was selected to maximize the F1 Macro score, prioritizing balanced performance across all classes.

5. Analysis Results

This section presents the results of the analysis for each method, followed by a comparison and evaluation of the methods.

5.1. Performance of the Methods

As stated before, we have analyzed three phishing detection methods: VisualPhishNet, Phishpedia, and Baseline. In this section, we present results for each of them.

5.1.1. VisualPhishNet

The VisualPhishNet method binary classification results are presented in Table 2.

Table 2. Results for binary classification metrics for VisualPhishNet.

Dataset	F1	ROC AUC	MCC
CERT	0.3577	0.4101	−0.1812
VP	0.4954	0.5887	0.1693
PP	0.1673	0.1018	−0.6160

The best results for binary classification were achieved for the VP dataset (F1 = 0.4954, ROC AUC = 0.5887, MCC = 0.1693), indicating a moderate ability to differentiate between the classes. The results for the CERT Polska dataset were lower (F1 = 0.3577, ROC AUC = 0.4101, MCC = −0.1812), and for the PP dataset, the results were the lowest (F1 = 0.1673, ROC AUC = 0.1018, MCC = −0.6160). Table 3 presents results for multiclass classification for the VisualPhishNet method.

Table 3. Results for multiclass classification metrics for VisualPhishNet.

Dataset	F1 Micro	F1 Macro	MCC	Identification Rate
CERT	0.3654	0.1481	0.0733	0.7093
VP	0.3924	0.0047	0.0694	0.0037
PP	0.1003	0.0334	0.0111	1.0000

The method achieved the best results of F1 micro = 0.3924 for the VP dataset, and a perfect Identification rate (1.0000) for the PP dataset. On the other hand, on the VP dataset, the method achieved a very low F1 macro score of 0.0047. On the CERT Polska dataset, the method gave F1 micro = 0.3654, F1 macro = 0.1481, and the second-highest Identification rate of 0.7093. Overall, the results show that VisualPhishNet gives good results for phishing target identification, but only when it correctly classifies the sample as

phishing (high levels of Identification rate). However, it fails to properly classify which samples belong to phishing classes (low levels of both F1 measures and MCC). In the binary classification, negative values of MCC indicate problems with classification and the reliability of predictions.

An extended evaluation of detection results for the VP dataset revealed that among 1442 samples classified as phishing, the model assigned these detections to only 14 unique source screenshots from the training set, representing three companies: Adobe, Absa, and Paschoalotto. The three most matched images are presented in Figure 3. They correspond to 85% of samples treated by the model as phishing. The first two do not show any significant features, but the third one was generated synthetically and is exceptionally dark. This means that the majority of samples detected as phishing are assigned to one of the three mentioned targets, which results in low detection metrics. This phenomenon indicates a "feature collapse" where the model overfitted to artifacts in the synthetic data (Figure 3c), causing it to map diverse inputs to a single, uninformative cluster in the embedding space.

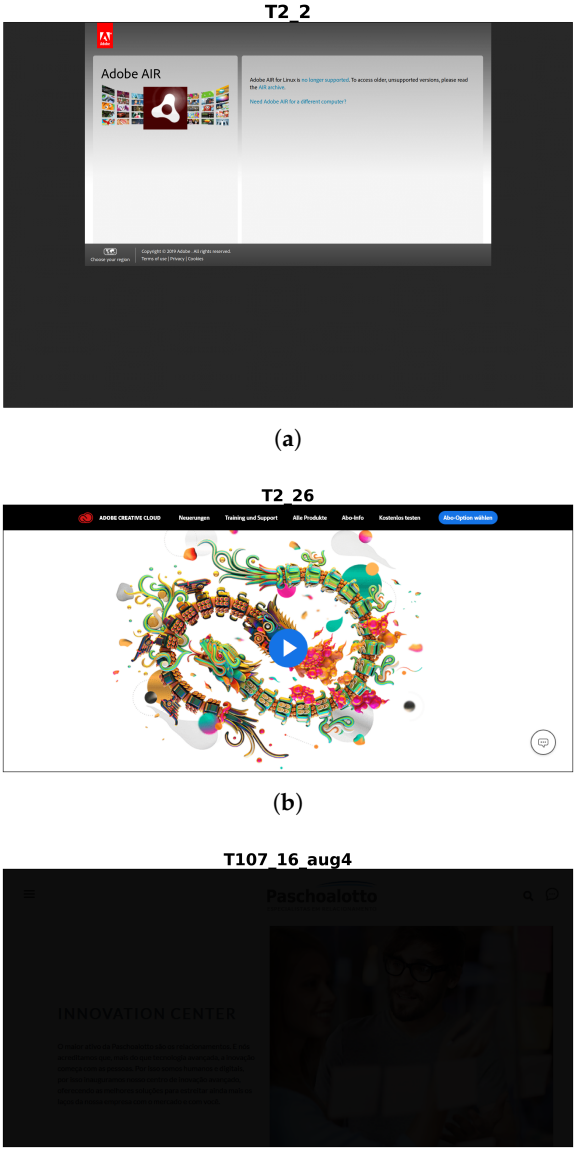


Figure 3. The three most matched images from the VP dataset for VisualPhishNet method: (**a**) Adobe Air, (**b**) Adobe Creative Cloud, (**c**) Synthetic augmented sample. The images correspond to 85% of samples treated by the model as phishing, failing to properly identify phishing targets.

5.1.2. Phishpedia

The results for binary classification for the Phishpedia model are presented in Table 4.

Table 4. Results for binary classification metrics for the Phishpedia.

Dataset	F1	ROC AUC	MCC
CERT	0.1598	0.5304	0.1301
VP	0.4263	0.4544	−0.0955
PP	0.9062	0.6679	0.2729

The method achieves good F1 levels for the PP dataset (F1 = 0.9062). The lowest F1 levels were obtained for the CERT Polska dataset (F1 = 0.1598), for which ROC AUC and MCC metrics were also low (0.5304 and 0.1301, respectively). The ROC AUC value was only slightly higher than for a random classifier, which achieves a ROC AUC of 0.5. For the VP dataset, F1 levels were lower than for the PP dataset but higher than for CERT Polska dataset (F1 = 0.4263), however ROC AUC and MCC levels were the worst from the three datasets. Results for multiclass classification for Phishpedia are in Table 5.

Table 5. Results for multiclass classification metrics for Phishpedia.

Dataset	F1 Micro	F1 Macro	MCC	Identification Rate
CERT	0.5482	0.2621	0.2013	0.9845
VP	0.3782	0.3073	0.2569	0.9270
PP	0.7691	0.2894	0.7384	0.9154

The results show that the method achieves high identification rates (from 0.9154 to 0.9845) for all datasets. For the PP dataset the method obtained good results regarding F1 micro and MCC (0.7691 and 0.7384, respectively), however low level for F1 macro (0.2894), hinting that only some frequent targets are classified properly. F1 macro classification levels for the other two datasets are similar to those of the PP dataset; however, their MCC levels and F1 micro are significantly lower. To summarize, Phishpedia gives good results regarding target Identification rate, and relatively good multiclass classification levels for the PP dataset. Nevertheless, binary classification results can be considered satisfactory only for the PP dataset, unlike the other two datasets.

5.1.3. Baseline

This method is based on the similarity of vectors representing visual hashes. In a training process equivalent to this, hashes for all screenshots in the training dataset are calculated. Next, using the validation dataset, hashes' distance is calculated between validation and training samples. The classification decision threshold is based on the calculated distance. We analyzed threshold values from 0 to the maximum distance in the validation dataset with a step value of 10, and then in the range of mean $\pm$ standard deviation with a step value of 1, following the same approach as the VisualPhishNet method. The optimal threshold was defined as the one maximizing the F1 macro measure. The decision to classify the screenshot as phishing was based on the distance between the analyzed sample and the training set samples. If the distance is higher than the decision threshold, then the image is labeled as benign. If it is classified as phishing, the target is assigned based on the target of the nearest sample from the training set. Results of binary classification of the Baseline method are presented in Table 6.

Table 6. Results for binary classification metrics for Baseline.

Dataset	F1	ROC AUC	MCC
CERT	0.6229	0.5682	0.1449
VP	0.7673	0.8201	0.6294
PP	0.9539	0.7759	0.5391

The method provides good results for the VP and PP datasets—F1 levels are high (0.9539 and 0.7673, respectively), with good levels of ROC AUC (0.8201 and 0.7759, respectively), and moderate levels of MCC. The results for the CERT Polska dataset are lower than those for the other two datasets, with a lower MCC value of 0.1449. The multiclass classification results for the Baseline method are presented in Table 7.

Table 7. Results for the multiclass classification metrics for Baseline.

Dataset	F1 Micro	F1 Macro	MCC	Identification Rate
CERT	0.5823	0.1670	0.3668	0.2554
VP	0.7009	0.4111	0.5089	0.5679
PP	0.5467	0.3591	0.4939	0.5687

The method achieves the highest F1 values from all the datasets for the VP dataset (F1 micro = 0.7009, F1 macro = 0.4111). The Identification rate of 0.5679 is comparable to the result for the PP dataset (0.5687). The same applies to the MCC values, which are 0.5089 and 0.4939, respectively. For the CERT Polska dataset, the F1 micro value is 0.0356 higher than for the PP dataset, but the other metrics are the lowest among all datasets. The Baseline method achieves relatively good results for binary classification across all the datasets. However, multiclass classification does not give such good results.

5.2. Comparison of Methods

To compare the methods, we have aggregated the evaluation results based on classification metrics. The binary classification results for the three analyzed methods are presented in Figure 4, where the results for each method are grouped by dataset for ease of presentation and discussion.

The Baseline method achieves the best results across all three metrics, yielding the highest F1 value of 0.9539 for the PP dataset, 0.8201 for the ROC AUC of the VP dataset, and 0.6294 for the MCC of the VP dataset. Depending on the metric and the dataset, the other two methods achieved varied results. Regarding the F1, VisualPhishNet achieved higher results than Phishpedia for the CERT Polska and the VP datasets, but significantly lower results for the PP dataset. The ROC AUC metric indicates that Phishpedia achieved higher values than VisualPhishNet for the CERT Polska and PP datasets; however, for the VP dataset, the relationship is inverted. When considering the MCC, VisualPhishNet achieved lower levels than Phishpedia for the CERT Polska and the PP datasets, but higher levels for the VP dataset. Figure 5 presents the results of multiclass classification grouped by datasets. Tables 8–10 provide comparison of multiclass classification results for CERT Polska, VP, and PP datasets, accordingly.

Regarding F1 micro and F1 macro, Phishpedia and Baseline methods achieve the best results. The Baseline method achieves better F1 micro results than Phishpedia for the CERT Polska and VP datasets, and the same holds for F1 macro results for the VP and PP datasets. For the F1 micro metric on the VP dataset, the VisualPhishNet method achieves better results than Phishpedia. For other datasets, their F1 micro and F1 macro levels are lower than those of the other two methods. Considering the MCC metric, VisualPhishNet achieves significantly lower results than the other two methods. Furthermore, Phishpedia is better

than Baseline only for the PP dataset. The Identification rate levels show that Phishpedia provides the best results, providing values higher than 0.9 for all three datasets. A peculiar situation arises with VisualPhishNet. For the PP dataset, it achieves a 1.0000 Identification rate, which is higher even for Phishpedia, whereas for the VP dataset, the value drops to 0.0037. Such a variance of results is significant. The Baseline method's results are inferior to those of the other two methods, except for the VP dataset. On the basis of the presented results, below conclusions can be made:

- The Baseline method provides the lowest variation in multiclass classification results between datasets compared to the other two methods. It achieves the best results for binary classification and relatively good results in multiclass classification.
- The Phishpedia method provides the highest levels of Identification rate, which do not change significantly between datasets, unlike VisualPhishNet. The method also achieves moderate results in multiclass classification.
- The VisualPhishNet method exhibited significant variations in detection results across datasets for both binary and multiclass scenarios. Also, in many cases, the method achieves the worst results among all the methods.

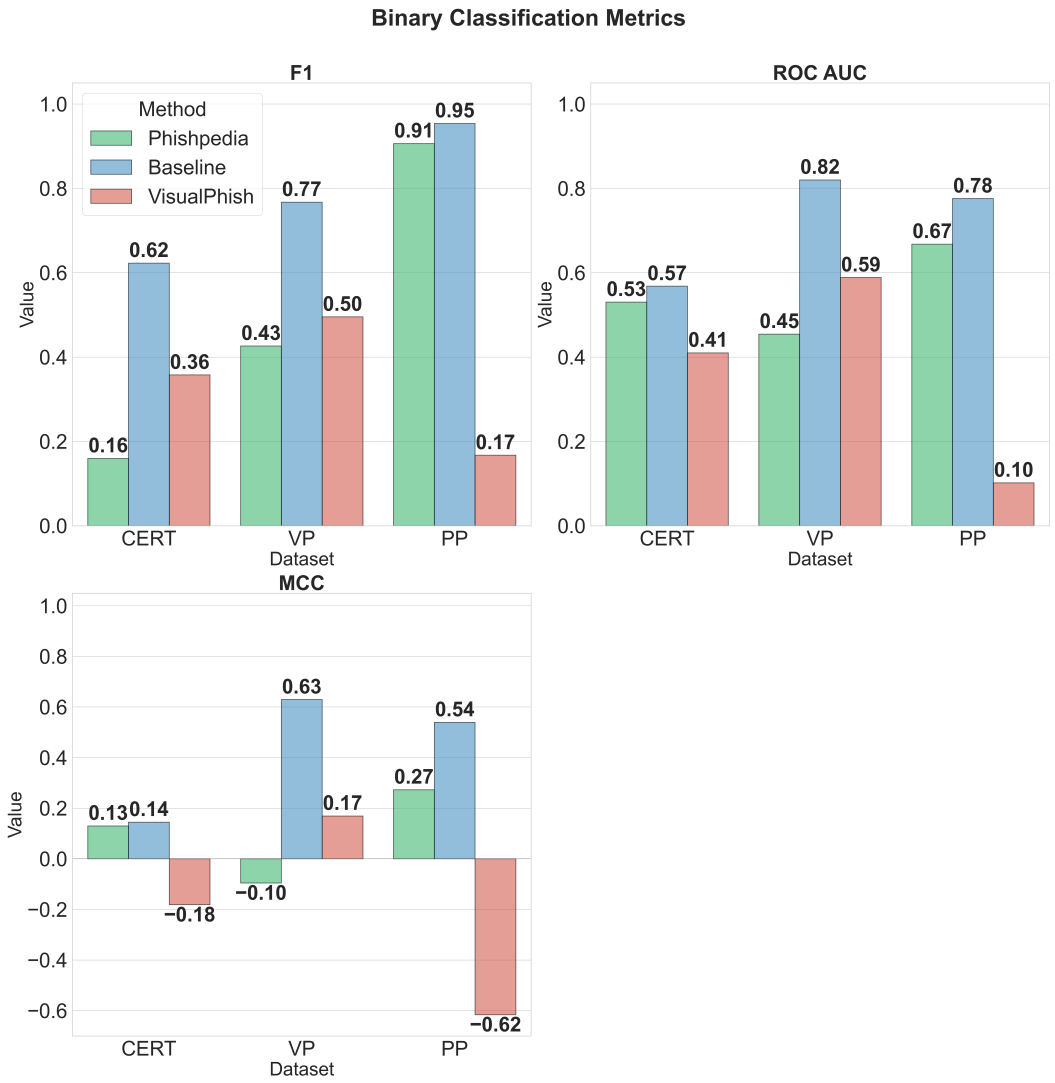


Figure 4. Binary Classification metrics. Each bar represents a different method, and a group of bars represents a dataset.

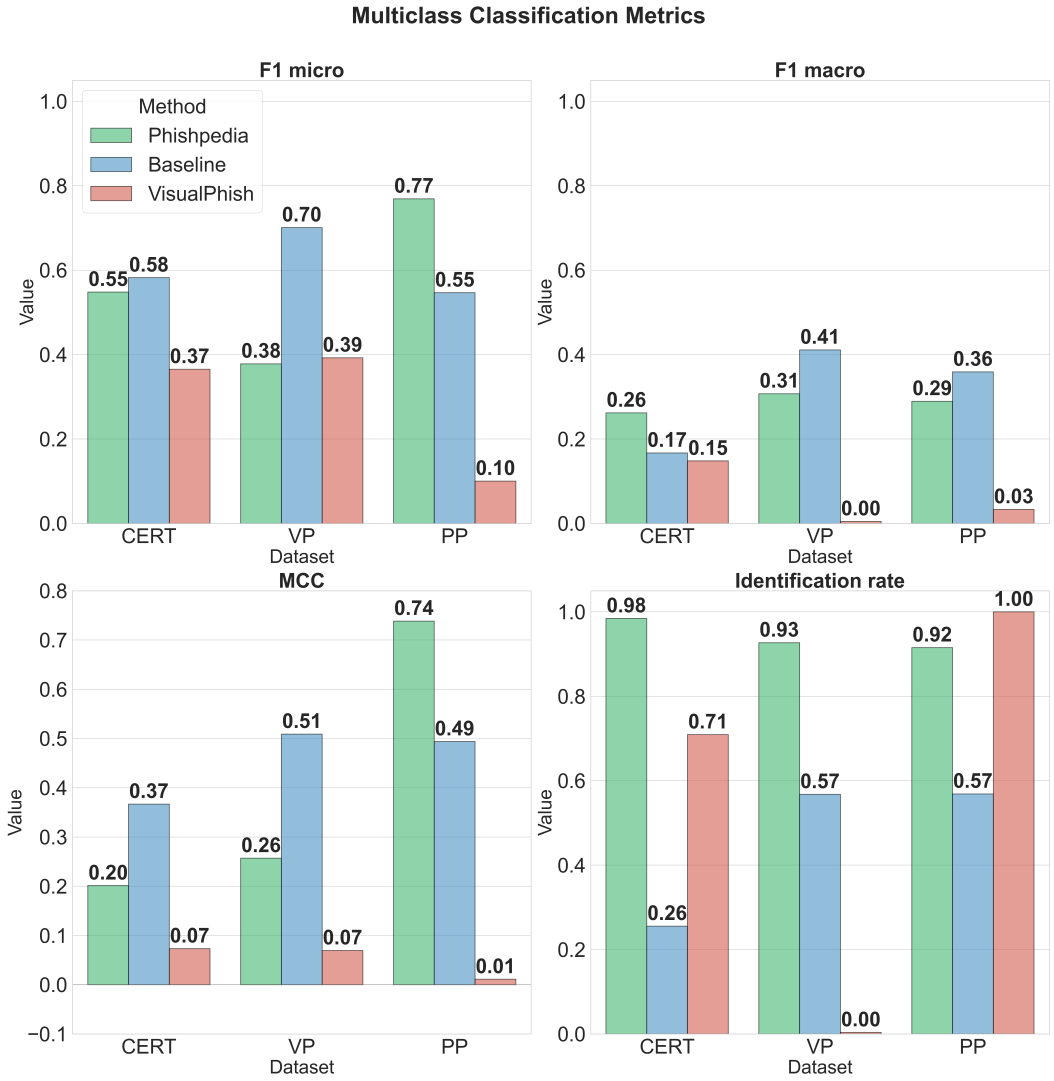


Figure 5. Multiclass Classification metrics. Each bar represents a different method, and a group of bars represents a dataset.

An additional factor to consider is that VisualPhishNet requires between 8 and 10 h to train the model, whereas Phishpedia or Baseline require approximately 20 min on the same hardware configuration. To summarize, the Baseline method is the best choice among the three analyzed for preliminary classification, while Phishpedia effectively identifies phishing targets. In a practical scenario, it would be beneficial to combine the output from two methods: the Baseline approach used for phishing classification and Phishpedia to identify the target. Based on our results, VisualPhishNet usability is limited.

Table 8. Comparison of methods on the CERT dataset—multiclass classification. The highest value in each column is in **bold**.

Method	F1 Micro	F1 Macro	MCC	Identification Rate
VisualPhishNet	0.3654	0.1481	0.0733	0.7093
Phishpedia	0.5482	**0.2621**	0.2013	**0.9845**
Baseline	**0.5823**	0.1670	**0.3668**	0.2554

Table 9. Comparison of methods on the VP dataset—multiclass classification. The highest value in each column is in **bold**.

Method	F1 Micro	F1 Macro	MCC	Identification Rate
VisualPhishNet	0.3924	0.0047	0.0694	0.0037
Phishpedia	0.3782	0.3073	0.2569	**0.9270**
Baseline	**0.7009**	**0.4111**	**0.5089**	0.5679

Table 10. Comparison of methods on the PP dataset—multiclass classification. The highest value in each column is in **bold**.

Method	F1 Micro	F1 Macro	MCC	Identification Rate
VisualPhishNet	0.1003	0.0334	0.0111	**1.0000**
Phishpedia	**0.7691**	0.2894	**0.7384**	0.9154
Baseline	0.5467	**0.3591**	0.4939	0.5687

6. Conclusions and Future Work

In this study, we addressed the challenge of detecting phishing websites and identifying their impersonation targets by developing a modular comparative framework. The experimental results demonstrate that the effectiveness of visual detection methods varies significantly depending on the specific task. Contrary to the assumption that complex deep learning architectures always yield superior results, our proposed Baseline method (utilizing perceptual hashing) proved to be the most robust for binary classification. It achieved the lowest variation in results across datasets and the best results in binary metrics (F1, ROC AUC, and MCC) for all three evaluated datasets. Conversely, VisualPhishNet delivered the weakest performance, suffering from "feature collapse" where synthetic training data artifacts caused the model to misclassify diverse inputs into a small cluster of incorrect targets. Phishpedia, however, demonstrated consistently high performance in the specific task of target recognition (Identification Rate), confirming its utility for attributing attacks to brands once they are detected.

Based on these observations, we recommend a hybrid approach for operational use. A combined system that utilizes the Baseline method for rapid initial classification and Phishpedia for precise target identification offers the optimal balance of performance and accuracy. This setup leverages the speed and robustness of perceptual hashing to filter traffic, while reserving the computationally expensive deep learning inference for confirming the target identity of suspicious pages.

The framework developed in this research is fully functional and designed to support security operators. However, future work should focus on transitioning from an experimental environment to a high-performance production system. A key limitation of this study was the reliance on a single train–test split due to computational constraints. Future iterations of this framework should incorporate k-fold cross-validation to provide rigorous uncertainty quantification and confidence intervals. Furthermore, accessing more extensive computational resources would enable the evaluation of larger, more diverse datasets, addressing the generalization challenges observed with synthetic data augmentation. Additional work can be put into providing more operational analysis scenarios, for example, analysis grouped by time periods to model phishing campaign time relationships.

Author Contributions: Conceptualization, M.J., P.B. and W.M.; methodology, M.J., P.B. and W.M.; software, M.J.; validation, M.J., P.B. and W.M.; formal analysis, M.J., P.B. and W.M.; investigation, M.J., P.B. and W.M.; resources, M.J. and P.B.; data curation, M.J. and P.B.; writing—original draft preparation, M.J., P.B. and W.M.; writing—review and editing, P.B. and W.M.; visualization, M.J. and

P.B.; supervision, W.M.; project administration, W.M.; funding acquisition, M.J., P.B. and W.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The VisualPhishNet dataset presented in the study is available on request for research purposes at https://github.com/S-Abdelnabi/VisualPhishNet (accessed on 15 December 2025). The Phishpedia dataset is openly available at: https://github.com/lindsey98/Phishpedia (accessed on 15 December 2025). The CERT Polska dataset is not available for legal reasons. Code to reproduce the experiments on the public datasets is available at https://github.com/Percival33/phish-target-recognition (accessed on 15 December 2025).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Haan, K. Top Website Statistics Today. 2024. Available online: https://www.forbes.com/advisor/business/software/website-statistics/ (accessed on 5 December 2025).
2. ENISA—European Union Agency for Cybersecurity. *ENISA Threat Landscape 2025*; ENISA: Athens, Greece, 2025. Available online: https://www.enisa.europa.eu/sites/default/files/2025-12/ENISA%20Threat%20Landscape%202025_v1.1.pdf (accessed on 29 December 2025).
3. Anti-Phishing Working Group (APWG). Phishing Activity Trends Report. 2nd Quarter 2025. 2025. Available online: https://docs.apwg.org/reports/apwg_trends_report_q2_2025.pdf (accessed on 20 November 2025).
4. Khonji, M.; Iraqi, Y.; Jones, A. Phishing detection: A literature survey. *IEEE Commun. Surv. Tutor.* **2013**, *15*, 2091–2121. [CrossRef]
5. Zieni, R.; Massari, L.; Calzarossa, M.C. Phishing or not phishing? A survey on the detection of phishing websites. *IEEE Access* **2023**, *11*, 18499–18519. [CrossRef]
6. Almomani, A.; Gupta, B.B.; Atawneh, S.; Meulenberg, A.; Almomani, E. A survey of phishing email filtering techniques. *IEEE Commun. Surv. Tutor.* **2013**, *15*, 2070–2090. [CrossRef]
7. Graziano, G.; Ucci, D.; Bisio, F.; Oneto, L. PhishVision: A Deep Learning Based Visual Brand Impersonation Detector for Identifying Phishing Attacks. In *Optimization, Learning Algorithms and Applications, Proceedings of the Third International Conference, Ponta Delgada, Portugal, 27–29 September 2023*; Pereira, A.I., Mendes, A., Fernandes, F.P., Pacheco, M.F., Coelho, J.P., Lima, J., Eds.; Springer: Cham, Switzerland, 2024; pp. 123–134.
8. Mishra, R.; Varshney, G. A Study of Effectiveness of Brand Domain Identification Features for Phishing Detection in 2025. In *Applied Cryptography and Network Security Workshops, Proceedings of the ACNS 2025 Satellite Workshops: AIHWS, AIoTS, QSHC, SCI, PrivCrypt, SPIQE, SiMLA, and CIMSS 2025, Munich, Germany, 23–26 June 2025*; Manulis, M., Ed.; Springer: Cham, Switzerland, 2026; pp. 89–108.
9. Bozkir, A.S.; Aydos, M. LogoSENSE: A companion HOG based logo detection scheme for phishing web page and E-mail brand recognition. *Comput. Secur.* **2020**, *95*, 101855. [CrossRef]
10. Ren, K.; Qiang, W.; Wu, Y.; Zhou, Y.; Zou, D.; Jin, H. An Empirical Study on the Effects of Obfuscation on Static Machine Learning-Based Malicious JavaScript Detectors. In Proceedings of the ISSTA 2023: 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis, Seattle WA USA, 17–21 July 2023; Association for Computing Machinery: New York, NY, USA, 2023; pp. 1420–1432. [CrossRef]
11. Abdelnabi, S.; Krombholz, K.; Fritz, M. VisualPhishNet: Zero-Day Phishing Website Detection by Visual Similarity. In Proceedings of the ACM Conference on Computer and Communications Security (CCS), Virtual, 9–13 November 2020.
12. Liu, R.; Lin, Y.; Yang, X.; Ng, S.H.; Divakaran, D.M.; Dong, J.S. Inferring Phishing Intention via Webpage Appearance and Dynamics: A Deep Vision Based Approach. In Proceedings of the 31st USENIX Security Symposium (USENIX Security 22), Boston, MA, USA, 10–12 August 2022; pp. 1633–1650.
13. Lin, Y.; Liu, R.; Divakaran, D.M.; Ng, J.Y.; Chan, Q.Z.; Lu, Y.; Si, Y.; Zhang, F.; Dong, J.S. Phishpedia: A Hybrid Deep Learning Based Approach to Visually Identify Phishing Webpages. In Proceedings of the 30th USENIX Security Symposium (USENIX Security 21), Virtual, 11–13 August 2021; USENIX Association: Berkeley, CA, USA, 2021; pp. 3793–3810.
14. Fu, A.Y.; Wenyin, L.; Deng, X. Detecting Phishing Web Pages with Visual Similarity Assessment Based on Earth Mover's Distance (EMD). *IEEE Trans. Dependable Secur. Comput.* **2006**, *3*, 301–311. [CrossRef]
15. Afroz, S.; Greenstadt, R. PhishZoo: Detecting Phishing Websites by Looking at Them. In Proceedings of the 2011 IEEE Fifth International Conference on Semantic Computing, Palo Alto, CA, USA, 18–21 September 2011; pp. 368–375. [CrossRef]

16. Seifert, C.; Stokes, J.W.; Colcernian, C.; Platt, J.C.; Lu, L. Robust scareware image detection. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 2920–2924. [CrossRef]
17. Saheed, Y.K.; Kehinde, T.O.; Ayobami Raji, M.; Baba, U.A. Feature selection in intrusion detection systems: A new hybrid fusion of Bat algorithm and Residue Number System. *J. Inf. Telecommun.* **2024**, *8*, 189–207. [CrossRef]
18. Chen, S.; Lu, Y.; Liu, D.J. Phishing Target Identification Based on Neural Networks Using Category Features and Images. *Secur. Commun. Netw.* **2022**, *2022*, 5653270. [CrossRef]
19. Bhurtel, M.; Siwakoti, Y.R.; Rawat, D.B. Phishing Attack Detection with ML-Based Siamese Empowered ORB Logo Recognition and IP Mapper. In Proceedings of the IEEE INFOCOM 2022—IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), New York, NY, USA, 2–5 May 2022; pp. 1–6. [CrossRef]
20. van den Hout, T.; Wabeke, T.; Moura, G.C.M.; Hesselman, C. LogoMotive: Detecting Logos on Websites to Identify Online Scams—A TLD Case Study. In *Passive and Active Measurement*; Springer International Publishing: Berlin/Heidelberg, Germany, 2022; pp. 3–29. [CrossRef]
21. Zeng, V.; Zhou, X.; Baki, S.; Verma, R.M. PhishBench 2.0: A Versatile and Extendable Benchmarking Framework for Phishing. In Proceedings of the CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual, 9–13 November 2020; ACM: New York, NY, USA, 2020; pp. 2077–2079. [CrossRef]
22. Hannousse, A.; Yahiouche, S. Towards benchmark datasets for machine learning based website phishing detection: An experimental study. *Eng. Appl. Artif. Intell.* **2021**, *104*, 104347. [CrossRef]
23. Dalton, T.; Gowda, H.; Rao, G.; Pargi, S.; Khodabakhshi, A.H.; Rombs, J.; Jou, S.; Marwah, M. PhreshPhish: A Real-World, High-Quality, Large-Scale Phishing Website Dataset and Benchmark. *arXiv* **2025**, arXiv:2507.10854. [CrossRef]
24. Ji, F.; Lee, K.; Koo, H.; You, W.; Choo, E.; Kim, H.; Kim, D. Evaluating the effectiveness and robustness of visual similarity-based phishing detection models. In Proceedings of the SEC '25: 34th USENIX Conference on Security Symposium, Seattle, WA, USA, 13–15 August 2025.
25. Zauner, C. Implementation and Benchmarking of Perceptual Image Hash Functions. 2010. Available online: http://phash.org/docs/pubs/thesis_zauner.pdf (accessed on 15 December 2025).
26. Ramírez, S. FastAPI. Available online: https://github.com/fastapi/fastapi (accessed on 15 December 2025).
27. Colvin, S. Pydantic. Data Validation Using Python Type Hints, Version v2.11.7. 2025. Available online: https://docs.pydantic.dev/latest/ (accessed on 14 August 2025).
28. Bayer, M. SQLAlchemy. In *The Architecture of Open Source Applications Volume II: Structure, Scale, and a Few More Fearless Hacks*; Brown, A., Wilson, G., Eds.; Lulu.com: Morrisville, NC, USA, 2012.
29. TorchVision Maintainers and Contributors. TorchVision: PyTorch's Computer Vision Library. 2016. Available online: https://github.com/pytorch/vision (accessed on 15 December 2025).
30. Douze, M.; Guzhva, A.; Deng, C.; Johnson, J.; Szilvasy, G.; Mazaré, P.E.; Lomeli, M.; Hosseini, L.; Jégou, H. The Faiss Library. *arXiv* **2025**, arXiv:2401.08281. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.