IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 18 November 2024, accepted 6 December 2024, date of publication 11 December 2024, date of current version 19 December 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3514972

TOPICAL REVIEW

A State-of-the-Art Review on Phishing Website Detection Techniques

WENHAO LI[1], **(Graduate Student Member, IEEE),**
SELVAKUMAR MANICKAM[1], **(Member, IEEE), YUNG-WEY CHONG**[2],
WEILAN LENG[3], **AND PRIYADARSI NANDA**[4], **(Senior Member, IEEE)**
[1]Cybersecurity Research Centre, Universiti Sains Malaysia, George Town, Penang 11800, Malaysia
[2]School of Computer Sciences, Universiti Sains Malaysia, George Town, Penang 11800, Malaysia
[3]Research Institute of Drilling and Production Engineering Technology, Chuanqing Drilling Engineering Company Ltd., CNPC, Guanghan 618300, China
[4]Faculty of Engineering and IT, University of Technology Sydney, Sydney, NSW 2007, Australia

Corresponding authors: Selvakumar Manickam (selva@usm.my) and Yung-Wey Chong (chong@usm.my)

ABSTRACT Phishing attacks remain a significant cybersecurity threat, with phishing websites serving as a primary tool for attackers to deceive users and steal sensitive information. The rapid evolution of phishing tactics has spurred the development of increasingly sophisticated detection mechanisms. This paper provides a comprehensive review of state-of-the-art techniques for phishing website detection, highlighting recent advancements in the field. In particular, it addresses emerging methods for detection, such as graph-based, large language model (LLM)-based approaches and phishing kit-based detection methods, which have not been extensively covered in previous surveys. By critically reviewing recent works from reliable databases, this study constructs a new taxonomy for phishing detection techniques. This review offers a comparison of these techniques, highlighting their strengths and limitations, and explores the challenges of real-world applications of these detection systems. Furthermore, the role of artificial intelligence (AI) in phishing website detection is discussed, and future research directions to improve detection capabilities are suggested. This work addresses emerging and uncovered phishing website detection methods in previous review papers and provides valuable insights for both researchers and practitioners working to develop more robust phishing website detection systems.

INDEX TERMS Cybersecurity, deep learning, machine learning, phishing website detection.

I. INTRODUCTION

Phishing attacks are one of the most persistent and damaging forms of cybercrime [1], targeting individuals and organizations alike by exploiting human vulnerabilities to deceive users into divulging sensitive information [2], [3]. These attacks come in many forms, including email phishing, spear phishing, smishing (SMS phishing), vishing (voice phishing), and more [4], [5]. Among these, phishing websites have emerged as one of the most significant threats, as attackers use visually convincing replicas of legitimate websites to steal login credentials, financial information, or other personal data. Phishing tactics are constantly evolving, and the detection and prevention of phishing websites have become a central focus of cybersecurity research [6], [7].

Figure 1 based on the data from Anti-Phishing Working Group (APWG) [8] shows a clear upward trend in the number of phishing websites from 2021 Q1 to 2024 Q1. Despite fluctuations from quarter to quarter, the overall increase highlights a growing phishing threat in recent years. Moreover, these phishing attacks can often lead to financial loss, loss of intellectual property, and reputational harm [9], [10].

As phishing attacks grow in complexity, phishing websites present distinct challenges. Attackers create these fraudulent websites, often mimicking trusted brands or institutions, intending to lure unsuspecting users into providing confidential information [11]. Traditional detection mechanisms

The associate editor coordinating the review of this manuscript and approving it for publication was Sedat Akleylek.

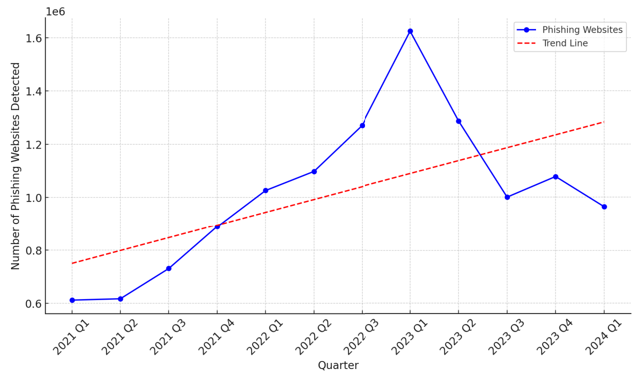


FIGURE 1. Number of unique phishing websites detected.

have focused on rule-based systems, but the increasing sophistication of phishing websites requires more advanced techniques capable of identifying subtle differences between legitimate and malicious sites. These techniques include text-based, image-based, and advanced Machine Learning (ML) and Deep Learning (DL) algorithms that more accurately identify phishing patterns [12].

Despite the extensive research on phishing website detection, a notable gap remains in the current literature. Many reviews have provided insights into traditional detection techniques; however, they fail to cover the latest advancements in the field comprehensively. Specifically, emerging detection mechanisms such as graph-based DL models [13], [14] that examine structural relationships within web pages, large language model (LLM)-based detection approaches [15], [16] that analyze both textual and visual content to identify phishing attempts based on contextual cues, brand verification, and cross-referencing domain names with webpage content, and phishing kit-based approaches [17], [18], which identify phishing attempts by detecting unique signatures left by phishing kits, have not been thoroughly explored. These techniques are cutting-edge methods that offer new approaches to phishing detection by leveraging advanced data structures, contextual analysis, and the exploitation of infrastructure-level weaknesses in phishing campaigns.

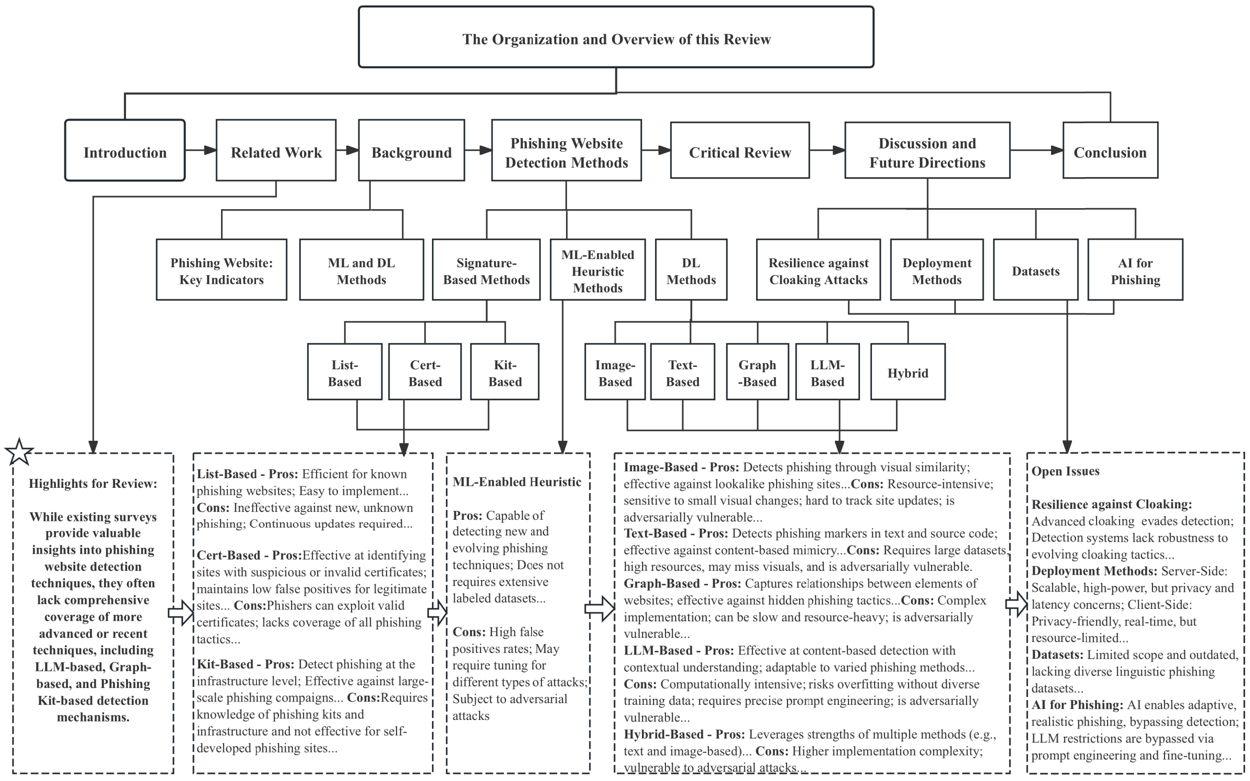
To address aforementioned gaps, this study aims to provide a comprehensive review of the latest phishing website detection techniques. This study's approach is grounded in an extensive search of well-known databases, including IEEE Xplore, ACM Digital Library, Elsevier, Web of Science (WoS), Scopus, and Google Scholar. Relevant literature was collected for review, with a specific focus on phishing websites to ensure the inclusion of all pertinent studies. To ensure a focus on recent advancements, a time filter was applied to prioritize technical papers published after 2020 while also incorporating select foundational works from earlier years to provide essential context.

The significance of this work is in its focus on new and emerging detection mechanisms. This review delves into areas of phishing detection that have not been previously highlighted, including graph-based approaches, LLM-powered models, and phishing kit-based detection. These emerging techniques provide deeper insights into the evolving nature of phishing attacks and offer more robust detection frameworks capable of handling the increasingly dynamic and deceptive nature of phishing websites. Furthermore, this review provides a critical analysis of each detection method's strengths and limitations, facilitating a more nuanced understanding of how these techniques can be applied in various phishing scenarios. In doing so, this review fills an important gap in the current literature, offering researchers and practitioners a detailed, up-to-date overview of the field.

The main contributions of this paper are as follows:

- First, this study presents an in-depth, systematic analysis of recent advancements in phishing website detection, which covers the latest graph-based, LLM-based, and phishing kit-based detection techniques. These areas are underexplored in previous surveys.
- Second, a new taxonomy of phishing website detection methods is constructed, which comprehensively classifies the state-of-the-art techniques into Signature-based, Heuristic, and Deep Learning categories and respective sub-categories.
- Third, this study provides a critical review and comparison of the strengths and limitations of each category of the proposed taxonomy, helping to clarify the applicability of various methods in different phishing scenarios.
- Last, this study uncovers several practical considerations of the current phishing website detection systems, discussing real-world challenges, emerging research areas, and potential future directions.

Figure 2 provides the organization and an overview of this review. The rest of the paper is organized as follows: Section II discusses related works, specifically other surveys, and reviews that examine phishing website detection mechanisms, highlighting their findings and limitations. In Section III, background information is provided to offer insights into key phishing website indicators and how they can be used to train ML and DL models for automated phishing detection systems. Section IV presents a comprehensive review of the literature on phishing website detection methods, detailing the research focus, methodologies, and findings of each approach. Section V provides a critical evaluation, highlighting the strengths and limitations of the various methods. In Section VI, the uncovered challenges of practical aspects of phishing detection systems are addressed, along with the role of Artificial Intelligence (AI) in advancing phishing techniques. Moreover, future directions are proposed to shed light on these issues. Finally, Section VII concludes this review. The list of commonly used acronyms and definitions in this paper is presented in Table 1.



Highlights for Review:

While existing surveys provide valuable insights into phishing website detection techniques, they often lack comprehensive coverage of more advanced or recent techniques, including LLM-based, Graph-based, and Phishing Kit-based detection mechanisms.

List-Based - Pros: Efficient for known phishing websites; Easy to implement... **Cons:** Ineffective against new, unknown phishing; Continuous updates required...

Cert-Based - Pros: Effective at identifying sites with suspicious or invalid certificates; maintains low false positives for legitimate sites... **Cons:**Phishers can exploit valid certificates; lacks coverage of all phishing tactics...

Kit-Based - Pros: Detect phishing at the infrastructure level; Effective against large-scale phishing compaigns... **Cons:**Requires knowledge of phishing kits and infrastructure and not effective for self-developed phishing sites...

ML-Enabled Heuristic

Pros: Capable of detecting new and evolving phishing techniques; Does not requires extensive labeled datasets...

Cons: High false positives rates; May require tuning for different types of attacks; Subject to adversarial attacks

Image-Based - Pros: Detects phishing through visual similarity; effective against lookalike phishing sites...**Cons:** Resource-intensive; sensitive to small visual changes; hard to track site updates; is adversarially vulnerable...
Text-Based - Pros: Detects phishing markers in text and source code; effective against content-based mimicry...**Cons:** Requires large datasets, high resources, may miss visuals, and is adversarially vulnerable.
Graph-Based - Pros: Captures relationships between elements of websites; effective against hidden phishing tactics...**Cons:** Complex implementation; can be slow and resource-heavy; is adversarially vulnerable...
LLM-Based - Pros: Effective at content-based detection with contextual understanding; adaptable to varied phishing methods...
Cons: Computationally intensive; risks overfitting without diverse training data; requires precise prompt engineering; is adversarially vulnerable...
Hybrid-Based - Pros: Leverages strengths of multiple methods (e.g., text and image-based)... **Cons:** Higher implementation complexity; vulnerable to adversarial attacks...

Open Issues

Resilience against Cloaking: Advanced cloaking evades detection; Detection systems lack robustness to evolving cloaking tactics...
Deployment Methods: Server-Side: Scalable, high-power, but privacy and latency concerns; Client-Side: Privacy-friendly, real-time, but resource-limited...
Datasets: Limited scope and outdated, lacking diverse linguistic phishing datasets...
AI for Phishing: AI enables adaptive, realistic phishing, bypassing detection; LLM restrictions are bypassed via prompt engineering and fine-tuning...

FIGURE 2. Paper organization and overview.

II. RELATED WORKS

Phishing is a broad topic that has been extensively researched over the years from various perspectives, including attack strategies and phishing detection methods. Several surveys and reviews have critically summarized these techniques, with some offering general overviews and others focusing on specific aspects. In this section, these works primarily dedicated to the detection of phishing websites are comparatively examined, highlighting their objectives, key findings, and potential limitations to indicate the motivation for this research.

Dou et at. [19] review various approaches for detecting phishing websites, focusing on software-based solutions to reduce human errors in identifying phishing attacks. The authors discuss the phishing detection solutions lifecycle, which includes feature extraction from sources like Uniform Resource Locators (URLs) and network-based attributes, followed by the application of various detection techniques. The survey explores ML approaches, heuristic-based methods analyzing content, URL patterns, or visual similarity, blocklist-based methods relying on lists of known phishing sites, and hybrid methods that combine heuristic and blocklist approaches for improved defense. In addition, the survey covers the use of plugins and toolbars incorporated into browsers for phishing detection. The survey finds that software-based solutions outperform user education, and combining multiple feature sets improves robustness. However, the discussed methods heavily rely on manually designed features and lack DL methods.

Singh and Meenu [20] discuss various types of information that can be used to detect phishing websites, categorizing approaches based on input data, including heuristic-based, blocklist, fuzzy-based, ML, and image-based methods. The authors emphasize that accurate feature extraction, particularly URL-based, source code-based, and image-based features, is critical for phishing detection. However, the paper provides only a high-level description, discusses a few works, and focuses heavily on ML methods without providing comprehensive details on other approaches like signature-based methods or how features are extracted and applied. This highlights a gap in the survey, as reflected in Table 2, where the discussion is largely limited to ML methods without exploring other significant detection techniques.

Zaimi et al. [21] present phishing detection methods, dividing them into two main categories: content-based and non-content-based approaches. Content-based methods include URL analysis, which examines URL features; image analysis, which compares visual elements like logos and screenshots; and text analysis, which evaluates website text for suspicious keywords, spelling, grammar, and Secure Sockets Layer (SSL) certificate usage. On the other hand, non-content-based methods consist of blocklist/allow list approaches,

TABLE 1. List of commonly used acronyms and definitions.

Acronym	Definition
AI	Artificial Intelligence
APWG	Anti-Phishing Working Group
BERT	Bidirectional Encoder Representations from Transformers
BiLSTM	Bidirectional-LSTM
BP	Belief Propagation
CNN	Convolutional Neural Network
CN	Common Name
CSS	Cascading Style Sheets
DL	Deep Learning
DNS	Domain Name System
DOM	Document Object Model
DT	Decision Tree
DV	Domain Validation
EV	Extended Validation
FL	Federated Learning
GAN	Generative Adversarial Network
GCN	Graph Convolutional Network
HTML	Hypertext Markup Language
HTTPS	Hypertext Transfer Protocol Secure
IP	Internet Protocol
JS	JavaScript
KNN	K-Nearest Neighbors
LLM	Large Language Model
LR	Logistic Regression
LRCN	Long-term Recurrent Convolutional Network
LSTM	Long Short-Term Memory
MHSA	Multi-Head Self-Attention
ML	Machine Learning
MLP	Multilayer Perceptron
MRF	Markov Random Field
NB	Naive Bayes
NLP	Natural Language Processing
OCR	Optical Character Recognition
OV	Organization Validation
OU	Organizational Unit
RNN	Recurrent Neural Network
RF	Random Forest
SVM	Support Vector Machine
URL	Uniform Resource Locator

Domain Name System (DNS) consistency techniques, and user website rating, which involves gathering user feedback to assess website trustworthiness. The authors conclude that content-based phishing detection methods, particularly those leveraging ML, are more effective than non-content-based techniques. However, the survey has several limitations, including the review of very few papers and a high-level description of each method. Besides, it mainly focuses on ML-based methods and lacks a comprehensive review of other methods, such as signature-based methods.

Alkawaz et al. [22] explore and analyze several ML techniques used for detecting phishing websites. The authors describe methods that combine content similarity allow lists, style similarity, and heuristics while also discussing the performance of various ML classifiers. They show that models such as Random Forest (RF) and hybrid models demonstrate high accuracy in phishing detection. However, the paper mainly focuses on ML-based approaches while neglecting the potential of DL models for phishing detection. Also, the survey provides only a high-level overview, with minimal organization or categorization, and does not offer a detailed comparison of the methods.

Tang and Mahmoud [12] provide a comprehensive overview of ML techniques for identifying phishing sites. The survey covers traditional methods like blocklists and rule-based approaches, as well as advanced ML methods. The main works discussed include support vector machines (SVM), decision trees (DT), and DL models like Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). The paper highlights that ML has higher accuracy in detecting new phishing sites compared to traditional methods but with limited coverage of other detection approaches, such as certificate-based and phishing kit-based detection methods.

Rajeswary and Thirumaran [23] analyze existing automated phishing detection techniques for websites, focusing on human behavior and AI-based methods. The authors highlight the effectiveness of both ML and DL techniques, as well as the importance of considering human behavior in phishing detection. However, the paper has several limitations, including a high-level overview, limited review of works, absence of method categorization, and small coverage of the techniques used for phishing detection.

Kumar et al. [24] focus on works aimed at identifying and blocking phishing URLs, particularly discussing DL models such as CNN and Long Short-Term Memory (LSTM), which are used to capture spatial patterns and long-term dependencies in URLs by leveraging Natural Language Processing (NLP) techniques. The survey also covers traditional ML models like RF, DT, Naive Bayes (NB), and K-Nearest Neighbors (KNN) for URL classification based on extracted features. The key findings of the survey emphasize the effectiveness of CNN and LSTM-based models for phishing detection, particularly in classifying phishing URLs and websites. However, the survey has many limitations, including a lack of clear categorization, a narrow focus on URL-based approaches, and an emphasis on complex DL models without detailed descriptions of feature extraction, performance, or datasets.

Safi and Singh [25] provide a systematic literature review (SLR) comparing various phishing detection approaches. The survey categorizes the used methods into five main approaches: heuristic techniques, which analyze specific features like URLs, text content, DNS information, and website traffic to distinguish phishing from legitimate sites; visual similarity-based techniques, which compare visual elements such as Cascading Style Sheets (CSS), text layout, logos, and screenshots to legitimate websites; list-based techniques, which rely on blocklists of known phishing sites and allow lists of trusted sites; ML techniques, which use attributes like URL structure and JavaScript (JS) features to train classifiers; and DL techniques, which employ models such as CNNs and RNNs for phishing detection. The authors found that the RF classifier was used in 31 studies and performed well in many cases. Additionally, the PhishTank [26] was the primary source for phishing data in 53 studies, while the Alexa [27] was used for legitimate data in 29 studies. However, the survey has several limitations, including presenting all methods in table form, which makes the paper difficult to follow, with no detailed explanations of each method.

Asiri et al. [28] discuss the challenges posed by phishing attacks and the limitations of existing detection methods, categorizing various approaches based on data preprocessing, feature extraction, model design, and performance. The study primarily emphasizes URL-based detection while paying less attention to hybrid methods and providing a very detailed description of each method. It emphasizes the importance of data preprocessing techniques such as data cleaning, tokenization, and embedding for DL models. The authors criticize that state-of-the-art methods are highly dependent on the datasets used, making them vulnerable to zero-day phishing methods. However, the survey provides a very detailed description of each method, which limits the coverage of other papers that cover other methods.

Zieni et al. [9] categorize phishing detection approaches into three main categories: list-based, similarity-based, and ML-based. The survey describes the methods proposed in the literature for each category, discusses the datasets used for their evaluation, and identifies research gaps. For instance, list-based approaches rely on blocklists and allow lists of known phishing and legitimate URLs. In addition, similarity-based approaches focus on the textual and visual similarity between phishing websites and their legitimate counterparts. Finally, ML-based approaches involve extracting features from URLs, Hypertext Markup Language (HTML), and other web page elements, followed by the application of ML models to classify websites as phishing or legitimate. The authors find that list-based approaches are effective but reactive, struggling with new phishing mechanisms. Similarity-based approaches are strong at detecting visually and textually similar phishing sites but can be resource-intensive and slow. On the other hand, ML-based approaches show promising results, particularly in identifying unseen phishing sites. Nevertheless, they require careful feature selection and large and diverse datasets. However, it includes only high-level descriptions of the methods, insufficient coverage of advanced techniques, and a predominant focus on traditional methods.

Justindhas et al. [29] explore various ML algorithms, feature engineering techniques, data sources, and models used for phishing website detection. The survey also discusses general phishing types and techniques. Similar to other surveys, the paper confirms the effectiveness of ML for phishing detection and the importance of feature engineering. Even so, the survey includes only high-level descriptions, a limited review of works, a lack of organization or categorization among methods, and minimal coverage of techniques used in phishing website detection.

Kulkarni et al. [30] categorize phishing webpage detection approaches based on different inputs. First, URL-based analysis focuses on URL structure and characteristics, such as the presence of Internet Protocol (IP) addresses, use of Hypertext Transfer Protocol Secure (HTTPS), URL length, and suspicious words or special characters. In addition, web page content analysis examines HTML, CSS, and JS code for hidden content, suspicious form actions, and other phishing indicators. Besides, Visual-based techniques involve analyzing screenshots and logos to detect visual similarities between phishing and legitimate websites. Finally, Third-party features consider factors such as domain registration length, domain age, DNS records, and website traffic. The authors indicate that effective phishing detection heavily relies on selecting the right features. ML and DL approaches show remarkable performance, with ensemble learning techniques like RF and XGBoost showing promise in improving detection rates. Additionally, the survey notes that attackers can exploit LLMs to generate very convincing phishing websites automatically. While the survey presents a significant and up-to-date contribution to the field by covering a wide array of phishing detection approaches, it provides only a cursory discussion of the use of LLMs in phishing website detection. The potential applications of LLMs in phishing detection have not been explored in depth, nor have they been compared in detail with other methods. Additionally, the survey overlooks detection techniques based on phishing kits, which are increasingly used by phishers to construct phishing websites at scale, as well as graph-based methods that leverage graph features at a higher level of abstraction and show promising performance. Though the classification in [30] provides a valuable framework for organizing phishing detection techniques by categorizing them broadly into URL-based, webpage-based, and hybrid approaches, it fails to capture the full range of recent advancements and nuanced distinctions in modern phishing detection methods. The proposed taxonomy in this study offers a more representative and detailed classification by introducing distinct categories, such as signature-based methods (including list-based, certificate-based, and phishing kit-based techniques) and ML-enabled heuristic methods. Additionally, the proposed taxonomy provides a comprehensive breakdown of DL approaches, further divided into image-based, text-based, graph-based, LLM-based, and hybrid techniques. This refined structure not only reflects the diverse and specialized applications of ML and DL in contemporary phishing detection but also enhances clarity by delineating the roles of each technique, presenting a more comprehensive framework for understanding state-of-the-art detection methodologies.

Table 2 provides an overview of the comparison of recent survey and review works based on the content and focus of each work. To summarize, while existing surveys provide valuable insights into phishing website detection techniques, they often lack comprehensive coverage of more advanced or recent techniques, including LLM-based, graph-based, and phishing kit-based detection mechanisms. This motivates our research to fill these gaps, offering a more detailed and

TABLE 2. Comparison of related surveys on phishing website detection.

Reference	Signature-Based			ML-Enabled Heuristic	Deep Learning					Taxonomy
	List-based	Certificate-based	Phishing-kit		Text-based	Image-Based	Graph-Based	LLM-based	Hybrid	
[19]	✓	×	×	✓	✓	✓	×	×	✓	✓
[20]	✓	×	×	✓	✓	✓	×	×	✓	×
[21]	✓	✓	×	✓	✓	✓	×	×	✓	✓
[22]	✓	×	×	✓	×	✓	×	×	×	✓
[12]	✓	×	×	✓	✓	✓	×	×	✓	×
[23]	✓	×	×	✓	✓	✓	×	×	✓	×
[24]	×	×	×	×	✓	✓	×	×	✓	×
[25]	✓	✓	×	✓	✓	✓	×	×	✓	✓
[28]	✓	×	×	✓	✓	×	×	×	✓	×
[9]	✓	✓	×	✓	✓	✓	×	×	✓	×
[29]	×	×	×	✓	✓	✓	×	×	✓	×
[30]	✓	✓	×	✓	✓	✓	×	×	✓	✓
Ours	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

TABLE 3. Key indicators and phishing detection methods.

Category	Indicators / Features	Potential ML Methods	Potential DL Methods
Domain & URL Indicators	Irregular domain names (e.g., typosquatting like 'paypa1.com'), use of subdomains, encoded characters, unusually long URLs, unusual top-level domains (e.g., '.xyz')	DT, SVM, Logistic Regression (LR), RF	CNNs for detecting patterns in URLs; RNNs for handling sequences in URLs
Certificate Indicators	HTTP vs. HTTPS use, presence of self-signed certificates, invalid or suspicious certificate details, short registration duration, lesser-known Certificate Authorities (CAs)	Clustering algorithms (e.g., DBSCAN for certificate features), Bayesian networks	LSTM models for real-time certificate pattern detection
Network Indicators	Abnormal DNS patterns, discrepancies in packet round-trip times, Transport Layer Security (TLS) version, TLS libraries	Naive Bayes for detecting statistical anomalies, KNN for comparing network traffic patterns	Hybrid models integrating CNNs with traditional ML for comprehensive network feature analysis
Content Indicators	Urgent language ('Immediate Action Required'), generic greetings, poor grammar, low-resolution logos, fake trust seals	RF and Gradient Boosting for identifying feature patterns; LightGBM for overall content pattern recognition	CNNs for image analysis (logos, brand consistency); Transformer models for text-based language patterns
Source Code Indicators	Hidden fields, misleading redirects, JS code for capturing user data, and traces left by phishing kits in default filenames	LR for consistent feature identification, RF for pattern aggregation	RNNs and CNNs for analyzing HTML and JS code structure
Behavioral Indicators	Forced logins, unusual interactions (e.g., forced redirects), malicious JS for disabling browser functions (e.g., right-click disabling)	Heuristic-based RF approaches for real-time behavior checks	LSTM networks trained on JS and page behavior patterns
Visual Indicators	Visual resemblance to legitimate sites, low-quality images, logo misalignment, use of fake security icons or badges	—	Triplet CNN for comparing legitimate and phishing page visuals; Siamese networks for matching logos and other visual elements
Multimodal Indicators	Combination of text, images, and relational data for holistic detection; contextual analysis of language and visuals on the webpage	—	LLMs for integrating text and image analysis; Transformers for analyzing multimodal data and contextual relationships

updated perspective on phishing website detection strategies and uncover the open issues in this field.

III. BACKGROUND

This section provides an overview of the key indicators used to identify phishing websites, such as suspicious URL patterns, hidden forms, abnormal link behaviors, and obfuscated scripts. These indicators form the basis for training ML and DL models, which have emerged as effective tools for detecting phishing attacks. In addition, it introduces the foundational principles behind ML and DL, outlining their role in leveraging these indicators to improve automated phishing detection systems. Indicators span multiple data types, including domain and URL characteristics, certificate details, content patterns, source code structures, user behavior, and traffic metrics. Advanced detection methods, particularly in DL, leverage visual similarity, graph structures, and multimodal data integration to enhance phishing detection. Table 3 offers a quick reference on various phishing indicators, and potential ML and DL models can be used.

A. PHISHING WEBSITES: KEY INDICATORS

Phishing websites are deceptive platforms designed to mimic legitimate websites, tricking users into providing sensitive information such as passwords, credit card details, or personal identification. These fraudulent websites often appear convincing, making it difficult for users to identify them by themselves. Detecting phishing websites requires recognizing specific indicators that differentiate them from authentic sites. These indicators can range from variations in domain names to hidden irregularities in website content and behavior.

These websites often exhibit noticeable irregularities in their **domain names and URLs** that can serve as red flags [31] by manipulating one of the components of the URL as shown in Figure 3.

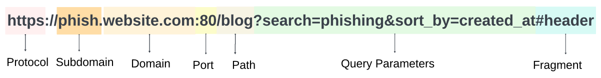

https://phish.website.com:80/blog?search=phishing&sort_by=created_at#header
Protocol
Subdomain
Domain
Port
Path
Query Parameters
Fragment

FIGURE 3. URL anatomy.

Attackers commonly use *typosquatting*, where legitimate brand names are deliberately altered by misspelling or swapping characters (e.g., "paypa1.com" instead of "paypal.com") [32], [33]. They may also insert additional words or numbers into the URL to confuse users, such as "login-paypal-security.com" rather than "paypal.com". In addition, special characters and creating excessively long URLs are generally considered hard evidence for phishing. For instance, special characters such as '-', '..','/','@' or numbers are commonly inserted into URLs to obscure the true domain and mislead users into believing the site is legitimate. A phishing URL may look like "www.bank-login.com/user-security-check," where the hyphen and extra path are used to deceive the user. In some cases, phishing URLs use *encoded characters*, such as "%20" (which represents a space in URL encoding), to hide important delimiter characters [34]. Also, *the length of the URL* can be an indicator for a phishing attempt, as phishing URLs tend to be unusually long and complicated, often filled with unnecessary parameters, subdirectories, or query strings that conceal the actual destination [35]. Moreover, URLs that include multiple subdomains, such as "www.login-update.paypal-security.com/verify," are designed to look like legitimate paths but may direct users to malicious pages [36]. On the other hand, attackers might also leverage less common or free top-level domains (TLDs) such as ".xyz," ".info," or ".click" to make their sites look unfamiliar but seemingly legitimate [37].

While many legitimate sites use HTTPS for secure connections, phishing websites may still rely on HTTP or, worse, use deceptive HTTPS certificates. They can use self-signed or suspicious certificates [38]. In practice, *CAs* issue certificates that contain fields such as *Common Name (CN)*, *Organization (O)*, and *Organizational Unit (OU)*. These details are displayed in two sections: *"Issued To"* (typically the domain owner) and *"Issued By"* (the issuer), along with information about the certificate's validity period and fingerprints. Websites are generally categorized into three types of SSL certificates: *Domain Validated (DV)*, *Organization Validated (OV)*, and *Extended Validation (EV)* [39]. The most basic level of validation is DV, where the domain owner is confirmed via an email from the CA, granting them control over the domain. OV certificates involve a more thorough process, requiring CAs to verify the domain owner's business name, status, type, and physical address. The highest level, EV, is considered the most strict validation. It includes verifying the business's public phone number, the duration of its existence, registration number, and other specific details about the organization [30].

The content on phishing websites also represents hard evidence for phishing attempts; it can be either **visual evidence** or **internal evidence**. Generally, the content plays a crucial role in executing social engineering attacks, aiming to manipulate and convince users to take harmful actions. Attackers tend to craft text that mimics legitimate websites, but minor inconsistencies can serve as key indicators of fraud. One common technique is the use of *urgent language*, similar to many other types of phishing, with phrases like "Immediate Action Required," which are designed to cause panic and prompt users to respond quickly without verifying the site's legitimacy [40]. Also, these websites often contain *generic greetings* or personalized messages for a specific target. In addition, *poor grammar*, *spelling mistakes*, and *strange words* are common indicators, as phishing websites are often created hastily or by individuals with limited proficiency in the target language [41]. Alternatively, visual deception can be in the form of media such as *images* or *webpage layout* [42]. Attackers may display a static screenshot of a login page or other interactive components that do not respond to user interaction as expected on a legitimate site. Also, phishing sites may use low-resolution logos, outdated branding, or incorrectly formatted media, making the overall design feel inconsistent with the legitimate one. Moreover, they may employ fake security badges or trust seals, often placed in specific areas of the page so that the user can notice them. On the other hand, beyond visual indicators, the *source code* of phishing websites often reveals additional internal indicators of malicious intent [43]. One common sign is the use of *suspicious or poorly structured HTML*, *JS*, and *CSS*, where attackers may include elements such as hidden fields, misleading redirects, or scripts designed to capture user inputs. They may also force users to log in or submit information before accessing any content, even in situations where a legitimate site would not require such actions. Another significant indicator is the usage of *phishing kits* [44], [45], which are pre-built templates for creating phishing websites. These kits often leave recognizable footprints in the code, such as default filenames, identical structures across multiple phishing sites, or even commented-out sections that reveal the kit's origin. Furthermore, these websites often include malicious JS that performs actions like disabling right-click, preventing content inspection, or redirecting users to fraudulent pages upon login. These internal code-level indicators, although less visible to the average user, are very important key indicators for phishing attempts.

B. ML AND DL METHODS

Building on these indicators, various ML and DL methods have been employed in phishing website detection. While certain models may be suited to specific data characteristics, in practice, the selection of a model typically requires testing multiple algorithms and fine-tuning their parameters to determine optimal performance. For instance, while CNNs are often associated with image data due to their capacity to capture spatial relationships, they have also demonstrated success in handling structured and even text data when spatial correlations are relevant. Eventually, website classification ("phishing" or "legitimate") is the common task of any model in this case.

ML models operate by finding patterns or relationships in data and using these learned patterns to make predictions [46]. The process begins with structured data, where each input is represented by features, which could be numerical or categorical. For instance, in the context of phishing detection, structured inputs might include URL characteristics (such as length or domain type), metadata, or other relevant features extracted from websites. A popular class of models in ML includes *DT*, where the model recursively splits the data based on feature values, forming a tree structure to classify new data points. Another example is the *SVM*, which works by finding a hyperplane that best separates the data into different classes. In *LR*, the model estimates probabilities for binary outcomes by fitting the input features to a sigmoid function. Other models, like RF and *Gradient Boosting*, combine multiple decision trees to improve accuracy and robustness by aggregating predictions across several trees. Generally, ML methods are well-suited for datasets where features are well-defined, and the relationships between features and labels can be learned effectively [47]. However, many models, such as SVM and DT, are very sensible to overfitting, particularly with noisy data. Figure 4 illustrates the ML pipeline, which begins with data collection and cleaning to ensure quality. Next, data preprocessing transforms the raw data into a suitable format for model training. Before the prediction phase, the model must be evaluated on a set of test samples to assess its performance.

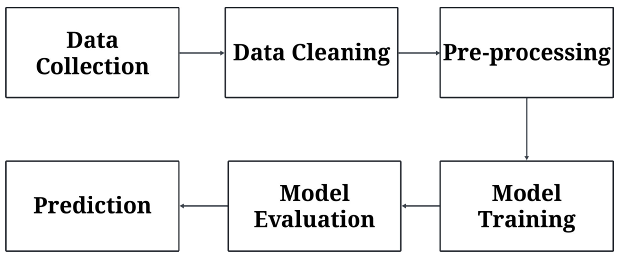


FIGURE 4. ML pipeline.

Alternatively, DL models have shown significant promise in processing and understanding **text data**, making them highly suitable for detecting phishing websites based on text content such as URLs or website content [48]. One commonly used model is the *RNN* (Recurrent Neural Network), particularly the *LSTM* variant, which is effective for sequential data like text. LSTMs are designed to capture the dependencies between words or characters over long sequences [49]. For example, an LSTM model can be trained to detect patterns in how phishing attempts structure sentences, such as the use of urgent language or repetitive phrases, or it can be used to detect the use of special indicators in the URLs, as discussed before. Figure 5 provides an overview of an RNN; it shows how RNNs process sequential data by taking an input sequence x and producing an output sequence y. At each time step t, the network takes the current input x_t and the previous hidden state h_{t-1} to compute the current hidden state h_t, which influences the output and the next time step. This mechanism allows RNNs to capture dependencies and patterns in sequential data. In addition, with the emergence of *Transformers*, such as Bidirectional Encoder Representations from Transformers (BERT), text processing tasks have become more accurate. BERT excels at understanding the context of words in a sentence by analyzing text bi-directionally [50], [51]. These models are generally used to learn a vector representation of a given string or long text, which will then be used for classification. To do so, these models have a certain memory limit, which is determined by the token size. It breaks unstructured data chunks of information, which will be used as a sequence for training.

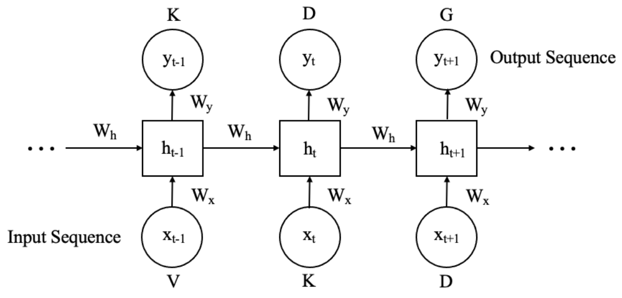


FIGURE 5. RNN overview.

In addition, **image analysis** plays a key role in identifying suspicious visual elements like logos, layout discrepancies, or embedded text in screenshots of phishing websites [52]. One of the most widely used models for image processing is the *CNN*, which is highly effective at detecting patterns in image data. CNNs work by applying a series of convolutional filters to the image, extracting hierarchical features like edges, shapes, and more complex structures. CNNs begin with feature extraction through convolution and pooling layers, where the network detects patterns and reduces dimensionality, which is then passed to fully connected layers for classification [53] as demonstrated in Figure 6. For phishing detection, CNNs can identify forged logos or visual inconsistencies, such as poorly rendered images or low-quality graphics. CNNs are designed to capture the spatial features of a given input; they can be further used in detecting the location of certain characters in text data. In cases where phishing sites attempt to mimic the design of legitimate websites, *object detection* models like *YOLO* (You Only Look Once) or *Faster R-CNN* can be employed. Also, image data can be used to extract text embedded within the image (such as screenshots) through *Optical Character Recognition* (OCR) [54], [55].

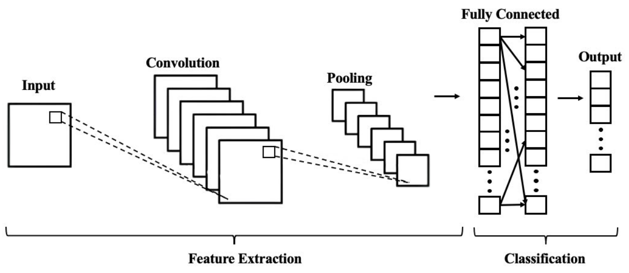


FIGURE 6. CNN overview.

Moreover, **graph data** is a structure composed of nodes (or vertices) and edges, mathematically represented as $G = (V, E)$, where V denotes the set of nodes and E represents the edges [13]. These edges can be either directed or undirected, depending on whether directional dependencies exist between nodes. In contrast to Euclidean spaces, the distance between nodes in a graph is not necessarily defined by their physical coordinates, making it laborious to apply traditional neural networks designed for Euclidean data. *Graph Neural Networks* (GNNs) are a type of DL models specialized and built in to learn from graph-structured data [56]. Using a message-passing mechanism, convolution mechanism, or attentional mechanism to aggregate information from neighboring nodes, enabling them to capture the relationships within the graph and then update the features as shown in Figure 8. GNNs are widely deployed across diverse applications such as social network analysis, biochemistry, fraud detection, and network security, demonstrating remarkable success in these areas [57], [58]. This makes GNNs particularly effective for classification, prediction, detection, and clustering at the node level, edge level, graph level, and subgraph level. For instance, interaction between users and websites can be modeled as graph data.

The figure 7 illustrates the process of using a graph-based approach for phishing website detection through GNNs. Given a website, two primary pathways are commonly employed in the literature to extract graph representations—either through visual elements (e.g., screenshots) or the raw source code of the website.

Using the source code, a redirection map of interconnected domains can be created by tracking various redirections originating from a single domain. Each domain or website in this redirection chain is represented as a node, forming the basis for a node classification task, where a node embedding algorithm encodes these nodes. The resulting node embeddings enable the identification of potentially malicious nodes within the network of domains.

Additionally, the source code can be parsed into a Document Object Model (DOM) tree, capturing the hierarchical structure of webpage elements, such as HTML tags. This typically results in a large-scale graph for each website. Another approach involves converting a screenshot of the website into a graphical structure, where visual features are represented as nodes and edges. Both the DOM tree and the image-based graph representation support graph-level classification tasks, generating graph embeddings that encapsulate the characteristics of entire websites.

Finally, **LLMs**, such as Generative Pre-trained Transformer (GPT), is built on the Transformer architecture, which uses self-attention mechanisms to capture the relationships between words and their context across long sequences of text [60]. LLMs are pre-trained on a large corpus of data and then fine-tuned for specific tasks, which have garnered widespread adoption and demonstrated exceptional effectiveness across various industries. For instance, in the banking sector, financial institutions are increasingly deploying LLM-powered solutions to enhance digital banking services [61]. Additionally, LLM-driven recommendation systems are supporting relationship managers [62], underscoring the trustworthiness and reliability of LLMs in critical sectors.

In phishing detection, LLMs can be applied in multiple ways. They can analyze large volumes of text data, such as phishing emails, website content, and URLs, to detect suspicious patterns or phishing language [63]. Generally, LLMs are integrated with a knowledge database. A query initiates the process, and an embedding model retrieves relevant context from the database based on the query, which is then passed to the LLM, which generates a response informed by both the input query and the database information as illustrated in Figure 9. LLMs are particularly effective at understanding contextual nuances in text, such as detecting the subtle difference between a legitimate request for information and a phishing attempt. Moreover, LLMs can be multi-modal, meaning they can process and integrate information from different data types, such as text and images. However, LLMs often require *prompt engineering* to fine-tune their outputs for specific tasks, which is the process of crafting precise input prompts to guide the LLM in generating accurate and relevant responses.

Figure 10 illustrates the general process of leveraging LLMs for phishing website detection. The process begins with the initialization of a web crawler that systematically collects the features, such as source code and screenshots of potential phishing websites. These raw data are then subjected to pre-processing techniques, where critical components such as the URL, source code, and visual elements (screenshots) are extracted for more effective analysis. Once the relevant data is extracted, a prompt is crafted using this information and fed into the LLM. The LLM is then tasked with making a binary classification decision, determining whether the website is legitimate or phishing.

IV. PHISHING DETECTION METHODS

Phishing website detection hinges on identifying specific features that reveal malicious intent. These features can be structured, like domain names or URLs, or unstructured,

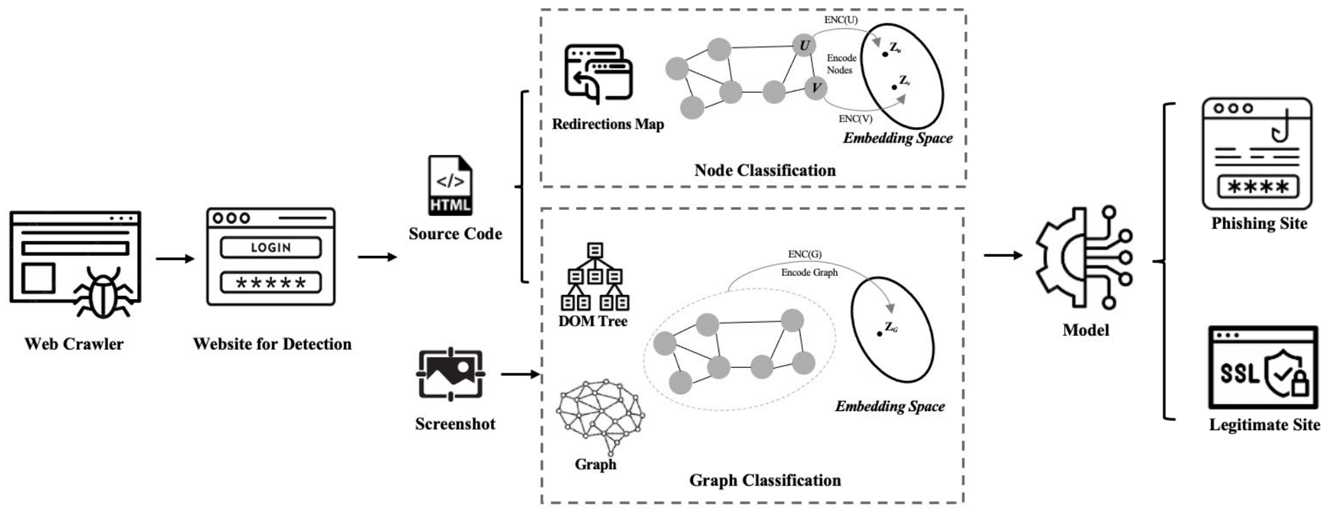


FIGURE 7. Illustration of graph-based phishing detection.

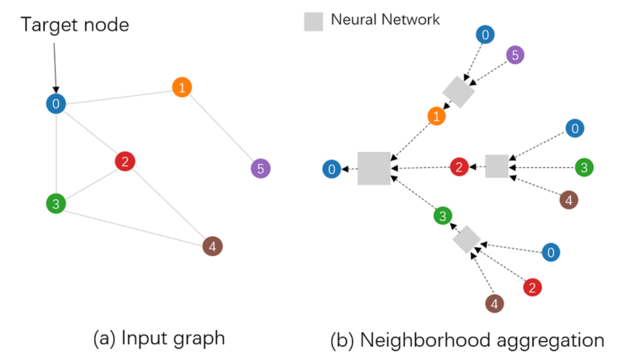


FIGURE 8. GNN overview [59].

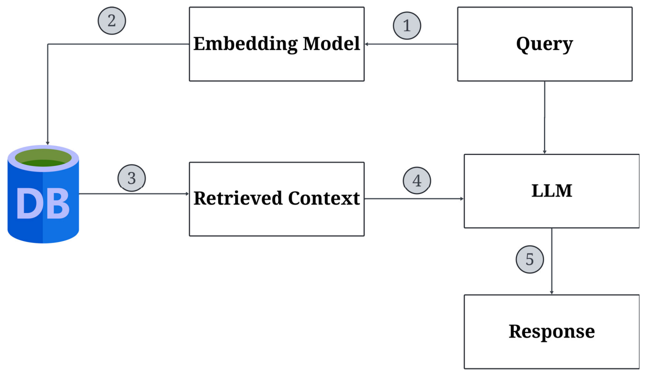


FIGURE 9. LLM overview.

like visual content or dynamic web elements. The nature of these features informs the selection of detection methods: structured data often lends itself to traditional ML techniques, while unstructured data requires more advanced approaches.

To address this diversity, phishing detection methods are categorized into three main approaches: Signature-based, Heuristic, and DL, as shown in Figure 11. While ML methods can be employed when structured data is available, these methods are discussed alongside more complex data formats and advanced approaches. Signature-based methods focus on identifying known patterns; Heuristic methods detect suspicious behaviors and novel threats. At the same time, DL approaches excel in learning from both structured and unstructured data, handling more complex phishing techniques.

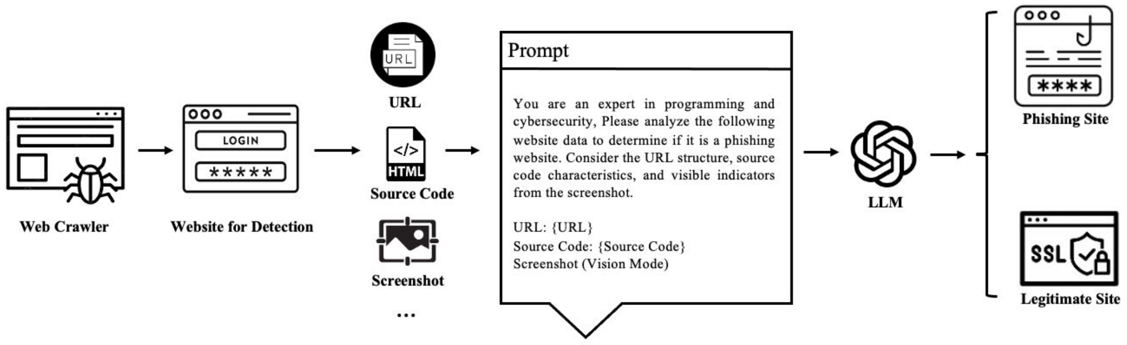
The taxonomy provides a comprehensive framework that aligns detection methods with the varying types of phishing data and evolving attack strategies, ensuring a more effective approach to combating phishing threats.

A. SIGNATURE-BASED

Signature-based methods for website phishing detection rely on identifying known patterns or signatures associated with phishing attacks. These signatures can include blocklists and allow lists of URLs, investigating certificates used by websites, detection of phishing kits, and the libraries leveraged by phishing sites. By matching observed behaviors or elements against these predefined signatures, signature-based approaches can quickly flag suspicious or malicious websites as described in Figure 12. However, these methods struggle to detect new or evolving phishing techniques and require sophisticated implementation to expand their ability against these new attacks. The main characteristics of each method are summarized in Table 4. It is important to note that some of the following methods are "Analysis-based," where the authors inspect the legitimacy of the website without incorporating an ML or DL classifier, and "N/A" means that the element has not been mentioned in the paper.

1) LIST-BASED

A common and straightforward defense against malicious websites involves the use of blocklists of known phishing URLs and allow lists of trusted ones. Many browsers, such



Prompt

You are an expert in programming and cybersecurity, Please analyze the following website data to determine if it is a phishing website. Consider the URL structure, source code characteristics, and visible indicators from the screenshot.

URL: {URL}
Source Code: {Source Code}
Screenshot (Vision Mode)

FIGURE 10. Illustration of phishing detection using a LLM.

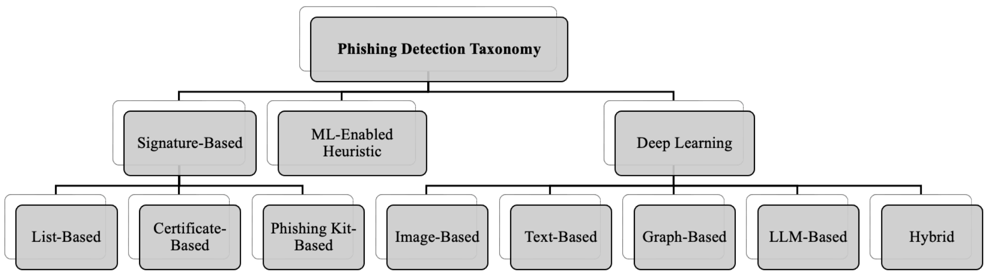


FIGURE 11. Taxonomy of phishing detection techniques.

as Google Chrome and Mozilla Firefox, integrate regularly updated blocklists, like the Google Safe Browsing service, to protect users [64]. Although effective against known threats, these list-based approaches struggle to detect zero-day phishing attacks—newly hosted phishing sites that have yet to be added to the blocklists.

Blocklisting stores known phishing URLs and blocks user access when a match is found, offering fast detection of previously identified threats. On the other hand, allow listing involves maintaining a database of legitimate URLs, only allowing pre-approved sites to be accessed, which ensures stronger security but may block legitimate sites that are not in the database. Despite its strengths, allow listing requires constant updates and can be resource-intensive to maintain [65]. To enhance the efficacy of list-based detection, hybrid methods have emerged, combining blocklists and allow lists with heuristic techniques, such as reputation scoring, URL analysis, and visual content comparison. Moreover, some research explored the client-side anti-phishing solutions leveraging these hybrid techniques to offer real-time detection and protection against both known and emerging threats. These approaches, integrating static lists with dynamic features, ensure more comprehensive phishing protection [66].

2) CERTIFICATE-BASED

As aforementioned, there has been an increasing trend of using certificates in phishing websites in recent years [67], as users often associate the HTTPS protocol with secure, encrypted connections and assume these indicators signal trustworthy sites. Therefore, attackers have adapted by creating self-signed certificates, enabling phishing sites to display HTTPS and a padlock icon, which deceives users into believing the sites are secure. Consequently, phishing detection methods have incorporated checks for web page certificates to identify such deceptive practices.

Drury and Meyer [38] investigate the public key certificates of phishing websites compared to those of legitimate websites to determine whether certificate information can serve as a reliable indicator for phishing detection. The approach involves several key steps to identify phishing websites using their public key certificates. It begins by collecting certificates from phishing and benign websites, using sources from Phishtank [26] and Alexa [27]. Once the certificates are gathered, they undergo feature extraction from validated and distinct certificates, focusing on key fields such as the subject's CN, Organization, Issuer, and validity information. These extracted features are then analyzed to identify patterns that distinguish phishing websites from benign ones. Special attention is given to comparing certificates from phishing sites to those of their intended targets, looking for differences in fields like organization names and issuers. With the analysis, the authors highlight that although differentiating the certificates of benign and phishing websites is hard, certificates can potentially be used as indicators of phishing websites.

To further investigate and leverage certificates for phishing detection, Sakurai et al. [68] focus on detecting phishing websites that use HTTPS encryption by analyzing the characteristics of their TLS certificates. The study first emphasizes more phishing websites leverage publicly free CAs, such as Let's Encrypt [69], and cPanel [70], to obtain TLS certificates, which lend them a legitimate appearance. The workflow of this method begins with the collection of phishing websites collected from Openphish [71], and corresponding TLS certificates stored in certificate transparency (CT) logs using Censys [72]. The clustering analysis using DBSCAN [73] is conducted to identify certificate groups of similar characteristics. Then, the intrinsic templates from grouped certificates are extracted and applied to the collected free CAs from Let's Encrypt [69] and cPanel [70]. With that, certificates potentially related to phishing attacks can be identified, and the authors further evaluate the findings with VirusTotal [74] and highlight that 90.8% of detected phishing websites had triggered alarms with at least one anti-virus provider on VirusTotal [74].

3) PHISHING KIT-BASED

Phishing kits, pre-packaged tools used by cybercriminals to create fraudulent websites, play a crucial role in website phishing schemes. Fortunately, each phishing kit typically leaves behind identifiable traces or "signatures" in the form of unique files, directories, or code structures, which helped researchers improve the phishing website detection tools.

To this matter, Kondracki et al. [17] focus on the comprehensive analysis and detection of Man-in-the-Middle (MiTM) phishing toolkits, which operate as reverse proxy servers between end users and targeted legitimate websites. These toolkits intercept sensitive user credentials and session cookies while maintaining seamless communication between users and legitimate web services. Consequently, this technique not only allows attackers to circumvent traditional security measures such as two-factor authentication (2FA) but also enhances the believability of the phishing pages. The authors first collected the MiTM phishing toolkits from the hacking forums and code repositories. Then, the exploratory data analysis was conducted on the collected toolkits to extract network-level features, which were then used to train a supervised ML classifier capable of detecting MiTM phishing toolkits with 99.9% accuracy. The authors conducted a longitudinal study using a proposed framework called *PHOCA*, which automatically collects phishing URLs from public data feeds, and then a queue-based system feeds these URLs into a crawler, which then collects various types of data about each website. This includes HTML content and screenshots, TLS certificates, IP addresses, domains, and classifications of original and redirected web pages. The collected data is then sent to an analysis module, which clusters web pages based on their content, helping to verify the classifications made by the detection model. The findings reveal that only a fraction of these toolkit-related domains and IPs appear on standard phishing blocklists, exposing many users to undetected phishing attacks. The authors suggest that online services could adopt fingerprinting techniques to identify and disrupt requests from these toolkits, enhancing real-time phishing prevention.

Bijmans et al. [18] present a detailed analysis of the Dutch phishing landscape, focusing on campaigns specifically targeting the Dutch financial sector. The study investigates the lifecycle of phishing campaigns, from the domain registration and deployment of phishing kits to the eventual takedown of phishing domains. The methodology begins by acquiring phishing kits aimed at Dutch banks, with 46 phishing kits collected through 50 public Telegram channels using a snowball sampling technique, as well as 24 phishing kits downloaded from open directories on suspected phishing domains. The researchers manually inspected the phishing kits to create unique fingerprints, identifying them based on file names, paths, and HTML strings. This fingerprinting enabled the efficient detection of phishing kits. For domain identification, the study monitored TLS CT Logs to track newly issued certificates, flagging potential phishing domains by detecting suspicious features such as Punycode use and the inclusion of brand names. Then, the scoring system was implemented to prioritize domains for further investigation based on the presence or usage of suspicious elements. Lately, flagged domains were then crawled using Selenium to collect data such as IP addresses, HTML source code, screenshots, and WHOIS information. Phishing kits were detected by matching the earlier-generated fingerprints, and additional resources on the domains were identified using HTTP GET requests. By using this crawler and matching unique fingerprints of known phishing kits, the system identifies whether a domain is associated with a phishing kit. The study identified 70 unique phishing kits, which were grouped into ten distinct families. The most prevalent kit, "uAdmin," was responsible for nearly 89% of the identified phishing websites. In total, 1,363 phishing domains deploying these kits were detected, primarily targeting Dutch banks. This study demonstrates the effectiveness of phishing website detection using signatures from phishing kits.

Zhang et al. [75] introduce a proactive defense mechanism called *Spartacus*, designed to protect users from phishing attacks that utilize server-side cloaking techniques. These techniques allow phishing websites to evade detection by presenting benign content to anti-phishing crawlers while displaying malicious content to actual users. The study begins by manually inspecting phishing kits from a dataset by *phishunt.io* [76] and extracting the features of the most common fingerprinting-based cloaking techniques. Then, it used these observed patterns to automatically find the phishing kits using these cloaking techniques. Spartacus was built based on these collected and observed features. Spartacus mainly operates by disguising user traffic to mimic that of anti-phishing crawlers. This is achieved through the manipulation of user-agent strings, HTTP headers, and the obfuscation of IP addresses, effectively tricking phishing websites into showing benign content instead of harmful pages. The framework consists of two main components: the front end and the back end. The front end handles the mutation of HTTP profiles, blocking blocklisted URLs and checking previously visited URLs from a local fingerprinting database. If the URL is not found or blocklisted, Spartacus mutates the HTTP request using predefined profiles that contain trigger words commonly blocked by phishing kits, such as "bot" or "crawler." The back end stores information related to fingerprinted phishing kits and manages the classification of web content. If suspicious content is detected, Spartacus updates the fingerprinting database and records the result of the mutation to prevent future phishing attacks from the same site. The system's effectiveness was rigorously tested by interacting with both known and unknown phishing sites. To evaluate the effectiveness of Spartacus, the study leveraged phishing feeds from APWG [8] and benign websites from Alexa's top website list [27]. The findings revealed that Spartacus successfully provided protection against phishing websites using cloaking techniques over 82.3% in the wild, effectively preventing users from encountering malicious content and with little impact on benign websites.

Lee et al. [77] present an in-depth analysis of phishing kits at the script level to better understand their server-side behaviors, classify UI patterns, and examine the evasive techniques employed in real-world phishing attacks. The methodology involves gathering a comprehensive dataset of phishing kits from compromised websites, performing a detailed examination of their server-side scripts, and collecting phishing websites from APWG [8]. The collected phishing kits are filtered, verified, and analyzed for their evasive behaviors, such as redirection features and dynamically generated URLs. In parallel, phishing HTML scripts are analyzed for user interaction patterns and information exfiltration methods. The user interaction patterns are classified into different categories, and their characteristics, including information leakage, are examined. In addition, the phishing kits' web appearance components are compared to the landing page scripts of phishing sites, allowing the identification of distinct landing page scripts. The findings reveal that phishing kits have become increasingly sophisticated in their evasion and credential theft techniques. Notably, over 60% of the analyzed phishing kits, as well as 1.5M phishing URLs, employed redirection as a way to evade detection. Moreover, this study revealed that a single kit was deployed on an average of 46.92 domains with an increasing deployment trend and highlighted the promising implications of building phishing detection methods based on the findings of this work.

B. ML-ENABLED HEURISTIC

ML-Enabled Heuristic-based methods determine the legitimacy of a website by extracting and analyzing specific features of a webpage rather than relying on block/allow lists. These features are typically derived from the URL and the HTML DOM of the page. Once extracted, they are compared against known characteristics of both phishing and legitimate websites to assess the site's authenticity. Previously, some approaches use heuristics to calculate a spoof score for each webpage, providing a measure of its legitimacy. In many recent cases, ML models are employed to enhance the accuracy and efficiency of these methods, as shown in Figure 13. Table 5 compares the ML-Enabled heuristic methods used for phishing website detection.

For instance, Sameen et al. [78] introduce *PhishHaven*, which represents an advanced AI-driven approach to phishing URL detection, leveraging ensemble machine learning techniques and a detailed analysis of lexical URL features. Key innovations include HTML URL encoding and the URL Hit method for detecting tiny URLs, allowing real-time identification of phishing threats. PhishHaven's system architecture is structured around four main components: URL Hit, Feature Extractor, Modeling, and Decision Maker. Initially, the URL Hit component determines if a URL is shortened by redirecting it through a browser plugin that captures the URL response. In the Feature Extractor phase, lexical characteristics are gathered using regular expressions, focusing on specific characters in the URL structure. These extracted features are then processed by ten distinct classifiers in the Modeling component, each generating independent predictions. The Decision Maker, utilizing a 67% threshold voting mechanism, assigns the final classification. PhishHaven incorporates parallel execution of ensemble machine learning models through multi-threading, enhancing both the efficiency and speed of phishing detection. Unique to PhishHaven is its independence from external databases, performing all URL analysis and classification internally, thus enabling detection without reliance on third-party services or language constraints. This system also identifies zero-day attacks by examining URL lexical patterns alone. Evaluated on a comprehensive dataset that includes 50,000 AI-generated phishing URLs from DeepPhish [79], 50,000 conventional phishing URLs from PhishTank [26], and 50,000 legitimate URLs from Alexa [27], PhishHaven achieves impressive accuracy, with an overall performance up to 98%, especially excelling in precision for AI-generated and human-crafted phishing URLs alike.

Sonowal and Kuppusamy [80] present the *PhiDMA* model, a multi-layered approach to phishing website detection that integrates allow listing, URL analysis, lexical signatures, string matching, and accessibility score comparison. The detection process in *PhiDMA* follows a sequential methodology, beginning with a URL allow list check to verify if the site is legitimate. If the URL is absent from the allow list, the model advances to a URL feature filter, examining specific attributes such as domain age, URL length, auspicious symbols, and dot count. If these characteristics raise red flags, the model then creates a lexical signature from the page content, which is matched against search engine results

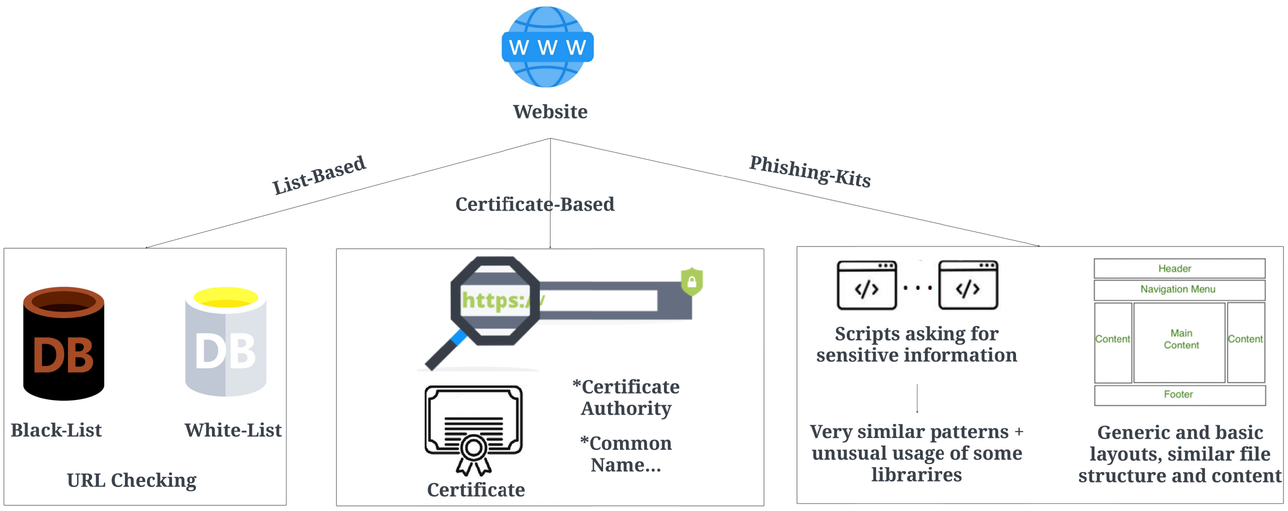


FIGURE 12. Signature based methods overview.

TABLE 4. Signature-based methods main characteristics.

Reference	Used Technique	Detection Mechanism	Datasets	Model	Accuracy
Google Safe Browsing [64]	Blacklist-Based	Uses a database of known phishing URLs	Google Safe Browsing's dataset	N/A	N/A
Drury et al. (2019) [38]	Certificate Comparison	Analyzes and Compares certificates of phishing and legitimate sites (highlights the potential of certificate information as a source for detection)	PhishTank + Alexa	N/A	N/A
Sakurai et al. (2020) [68]	Certificate Analysis	Clusters collected TLS certificates and find groups with similar characteristics	OpenPhish + Censys	DBSCAN	90.8%
Kondracki et al. (2021) [17]	Phishing-Kit Detection	Identifies network-level behaviors of phishing kits and develops a ML classifier	PhishTank + Openphish + CT Logs	RF	99.9%
Bijmans et al. (2021) [18]	Creates unique signatures for phishing-kits	Fingerprintings parts of the source code and structure of gathered phishing-kits and measures their prevalence on live phishing domains	Telegram + CT Logs	N/A	N/A
Zhang et al. (2022) [75]	Phishing Kit Analysis and Traffic based Server-Side Cloaking Detection	Disguises user traffic to mimic anti-phishing crawlers and trigger cloaking behavior	APWG + Alexa	N/A	82.3% (Protection Rate)
Lee et al. (2024) [77]	Script-Level Analysis of Phishing Kits	Measures the user interaction patterns and evasion techniques used in phishing kits and the trends of phishing kits are used and deployed	APWG	N/A	N/A

to assess legitimacy. If valid results are not returned, the URL is flagged as suspicious. Conversely, if relevant URLs are found, they undergo further verification using two string matching algorithms, Longest Common Subsequence, and Damerau-Levenshtein Edit Distance, to confirm legitimacy. Lastly, the model uses an accessibility analysis tool to compare the accessibility scores of the current page with those of the returned matching pages. High similarity between these scores indicates a phishing attempt, revealing the legitimate site targeted for impersonation. To evaluate the performance of the proposed mechanism, a dataset was built with legitimate data from Phishload [81] and phishing data from PhishTank [26].

Sánchez-Paniagu et al. [82] focus on investigating the authenticity of websites containing login forms. To address limitations in previous methods, the authors introduce new feature sets that consist of four groups, i.e., URL, HTML, Hybrid, and Web Technology. For instance, URL features consist of subdomain level features such as the number of subdomains on the URL, a subdomain named 'com' by including a binary feature set to 1 in case there is a subdomain called 'com,' IP address that corresponds to a binary value whether the address is represented as an IP address, Common Top-Level Domain by considering generic TLDs such as 'com,''org,' 'net,' 'edu', and 'gob' and all the Country Code TLDs (ccTLD) as ordinary, URL length, number of digits, and special characters such as '@,' '–,' '/,''?','='…The HTML features consist of considering Link features such as counting the number of external links, internal links, same page links, empty links, null links, Body length and tags, and Base64 resources by assigning a binary value if the HTML has content with base64 encoding, i.e., media files such as images. Furthermore, the Hybrid features consist of checking whether a website has the copyright disclaimer © or not. Also, instead of using the raw HTML, they remove the comments in case attackers placed their domain name into the comments as a detection bypass, verifying subdomain-copyright, path-copyright, title-domain-copyright, and general information about the domain extracted from the title and body. Finally, the technology-based features consist of counting the number of technologies used to develop the website (libraries, for example) and detecting specific technologies that are more frequent in developing legitimate websites. Lately, the detection model used was LightGBM, which was trained on these extracted features. The findings demonstrate the effectiveness of this approach, with the LightGBM classifier achieving a 97.95% accuracy on the PILWD-134K dataset [83].

Chinnasamy et al. [84] propose a comprehensive heuristic-based detection mechanism, which is applied to identify phishing websites by analyzing distinctive URL and webpage characteristics through machine learning algorithms. Leveraging a dataset of 1,871 labeled phishing and non-phishing websites, with 20% designated for testing, this approach uses selected features known to correlate with phishing behavior. These features range from URL properties like IP addresses, shortening services, length, symbols (e.g., '@'), and redirects to security attributes such as SSL state, HTTPS tokens, and WHOIS-based domain age. The detection method also incorporates behavioral indicators, such as web traffic metrics, page rank, and Google indexing status, to gauge site legitimacy. Machine learning models, including Random Forest, SVM, and Genetic algorithms, classify URLs based on patterns in these characteristics. The results show that the Genetic algorithm significantly outperformed the other two models, achieving the highest accuracy of 94.73%.

Roy et al. [85] analyze phishing attacks that exploit Free Website Building Services (FWBs) and introduce *FreePhish*, a scalable detection framework that identifies FWB-based phishing websites shared on social media platforms like Twitter and Facebook. The framework utilizes a layered approach with five key modules: streaming, preprocessing, classification, reporting, and analysis, enabling the identification, reporting, and longitudinal monitoring of FWB phishing threats. Through continuous monitoring, FreePhish captured and analyzed over 31.4K phishing URLs from 17 distinct FWB services over a six-month period. The study reveals how FWBs empower attackers by offering features that aid in the evasion of traditional anti-phishing mechanisms, such as obfuscating banner codes or employing "noindex" meta-tags to avoid search engine detection. Additionally, the authors compared popular anti-phishing tools and discovered significant limitations in coverage and response time when addressing FWB phishing attacks, underscoring gaps in current defenses. FreePhish, enhanced by a Chromium web extension, empowers end-users by blocking access to FWB-based phishing sites in real-time. The framework's effectiveness was demonstrated through high accuracy (97%) in detecting FWB phishing, contributing valuable insights to the anti-phishing ecosystem, and supporting the prompt removal of FWB-based phishing threats. The study provides a foundational tool for improving real-time phishing detection and prevention in the evolving landscape of FWB-based phishing attacks.

Bahaghighat et al. [86] leverage a robust machine learning-based approach, six algorithms—LR, KNN, NB, RF, SVM, and XGBoost—were implemented on a public dataset [87] comprising 58,000 legitimate and 30,647 phishing websites with 112 attributes each. To enhance model performance, constant features were removed, and the Synthetic Minority Over-sampling Technique and Edited Nearest Neighbors (SMOTEENN) method was applied to balance the dataset, addressing its inherent class imbalance. In the feature selection and evaluation process, eight unique scenarios were tested. Experimental results demonstrate impressive accuracies exceeding 93% across all algorithms, with XGBoost achieving outstanding results: 99.2% overall accuracy, 99.1% precision, 99.4% recall, and 99.1% specificity. Feature scaling, via standardization, was utilized to ensure consistent data ranges, while the model's runtime was optimized—achieving approximately 1500 ms without dimensionality reduction and 869 ms with principal component analysis. These steps enabled the model to operate effectively in both real-time and offline environments. This extensive evaluation shows that the XGBoost classifier, particularly in balanced scenarios, excels in phishing detection, presenting a practical solution with high precision, rapid processing, and adaptability for dynamic cybersecurity applications.

Adap et al. [88] proposes a URL-based phishing detection model using XGBoost and Random Forest classifiers, with XGBoost achieving a high accuracy of 96.51% and a kappa value of 0.930. The dataset for this model includes a balanced set of phishing and legitimate URLs gathered from trusted sources such as PhishTank [26], OpenPhish [71], and a public dataset on Mendeley [89], ensuring diversity and reliability. Preprocessing steps included filtering duplicates and minimizing lexical similarity using a Token Set Ratio, resulting in a balanced dataset of 50,934 URLs. Feature extraction involved 48 URL features—45 lexical and three external—enhanced by using OpenPageRank and Tranco as replacements for deprecated services like AlexaRank. The study highlights the dynamic nature of phishing URL features, as only 38 features retained relevance over time, emphasizing the importance of updating data to reflect current phishing tactics. Additionally, external features showed significant potential for robust detection, though their use can be limited by rate restrictions. Finally, The authors highlight that future work should consider adding HTML and JavaScript features, incorporating paid third-party data sources, and retraining the model on newer datasets to adapt to evolving phishing techniques.

Talukder et al. [90] employ NB classifiers—specifically Bernoulli, Multinomial, and Complement NB models—to enhance detection accuracy. Using a Kaggle dataset [91] with over 549,346 URL entries labeled as either "Good" (non-phishing) or "Bad" (phishing), the study applies rigorous data preprocessing. URLs are vectorized with CountVectorizer, and text tokens are refined with RegexpTokenizer for precise feature extraction, preparing data for NB classification. The implementation phase splits data into training and testing subsets, leveraging the independence assumptions of NB models to optimize classification. Evaluation reveals that Multinomial and Complement NB classifiers outperform Bernoulli, each achieving a 96% accuracy rate in identifying phishing sites.

C. DEEP LEARNING

ML classifiers require feature engineering, where relevant features must be manually designed from the training dataset to accurately detect phishing webpages. In contrast, DL algorithms eliminate the need for manual feature engineering, as they automatically extract relevant features from the training data. In addition, DL models are capable of processing unstructured data directly, such as media content and strings, making them more suitable for an efficient detection mechanism against phishing websites. Figure 14 highlights the main components of DL phishing detection tools.

1) IMAGE-BASED

The primary goal of a phishing website is to deceive users by mimicking the appearance of legitimate websites. Some of the early works propose reference-based visual similarity, comparing screenshots of a given webpage to the screenshots in the reference database. However, legitimate websites are continuously updated, making it both challenging and resource-intensive to track every change in the reference database. Still, various approaches have been proposed that leverage visual elements such as screenshots, logos, and brand indicators to assess the legitimacy of a website more effectively. Table 6 presents the comparison of different Image-based methods.

For instance, Abdelnabi et al. [92] developed *VisualPhishNet*, which employs a triplet CNN to learn the similarity between website screenshots. The triplet CNN takes three inputs: an anchor image (a legitimate website), a positive image (another legitimate instance), and a negative image (a phishing website). When a webpage is suspected of being phishing, the model compares its screenshot against those in the reference list, flagging it as phishing if a strong resemblance exists, regardless of domain differences. This approach provides enhanced resistance to phishing attacks by detecting visually similar fraudulent pages, even those previously unseen. To evaluate their approach, the authors created the VisualPhish dataset, which includes over 155 websites for both phishing and legitimate sourced from PhishTank [26] and the top 500 ranked websites from Alexa [27], the top 100 websites from SimilarWeb [93]. The experimental results showed that by using VisualPhishNet, 81% of the phishing test pages were matched to their correct website using the top-1 closest screenshot, while the top-5 match is 88.6%, outperforming previous works at a large margin.

Lin et al. [94] present *Phishpedia* that identifies phishing websites by visually comparing their identity logos with legitimate brand logos. Phishpedia integrates two core components: an object detection model that identifies logos on web pages and a Siamese neural network that matches the detected logos with those of known legitimate brands. By comparing logos from phishing sites to a reference set of legitimate logos, Phishpedia can further identify the targeted official webpage. Moreover, they synthesize a visual phishing explanation by detecting identity logos and input boxes in the screenshot with an object detection model. A key strength of this system is that it operates without relying on a phishing-specific training dataset, mitigating bias and improving generalization. Phishpedia was highly effective, achieving a precision overall identification rate of 99.2% on a dataset of 29,496 phishing webpages from OpenPhish Premium Service [71] and 29,951 benign sites from Alexa's top ranks. In addition, the authors highlight the importance of the object detection model's ability to accurately detect relevant objects, as this plays a crucial role in the subsequent classification of the webpage.

Trinh et al. [95] analyze the visual similarities between phishing and legitimate websites, leveraging pre-trained DL models, including VGG16, VGG19, ResNet50, InceptionV3, and Xception, to extract high-level visual features from website screenshots resized to 224×224 RGB images. By fine-tuning these models, they transform the visual features into feature vectors, which are then classified using ML algorithms such as SVM, RF, and k-NN for the classification task. The approach is evaluated using the VisualPhish dataset, which consists of 9363 legitimate and 1195 phishing site screenshots. The evaluation was made through the combination of different feature extractors and ML models. The results show that VGG16-LR achieved the best performance with 88.68% accuracy and 0.834 F1 score.

Liu et al. [96] introduce *PhishIntention* that combines visual appearance analysis and dynamic interaction monitoring to identify phishing websites. PhishIntention uses DL models to infer both brand intention and credential-taking behavior by examining webpage layouts and interacting with sites to confirm phishing activity. The system first extracts an Abstract Webpage Layout (AWL) from a screenshot to identify key User Interface (UI) elements like logos and input fields. It then performs brand intention recognition

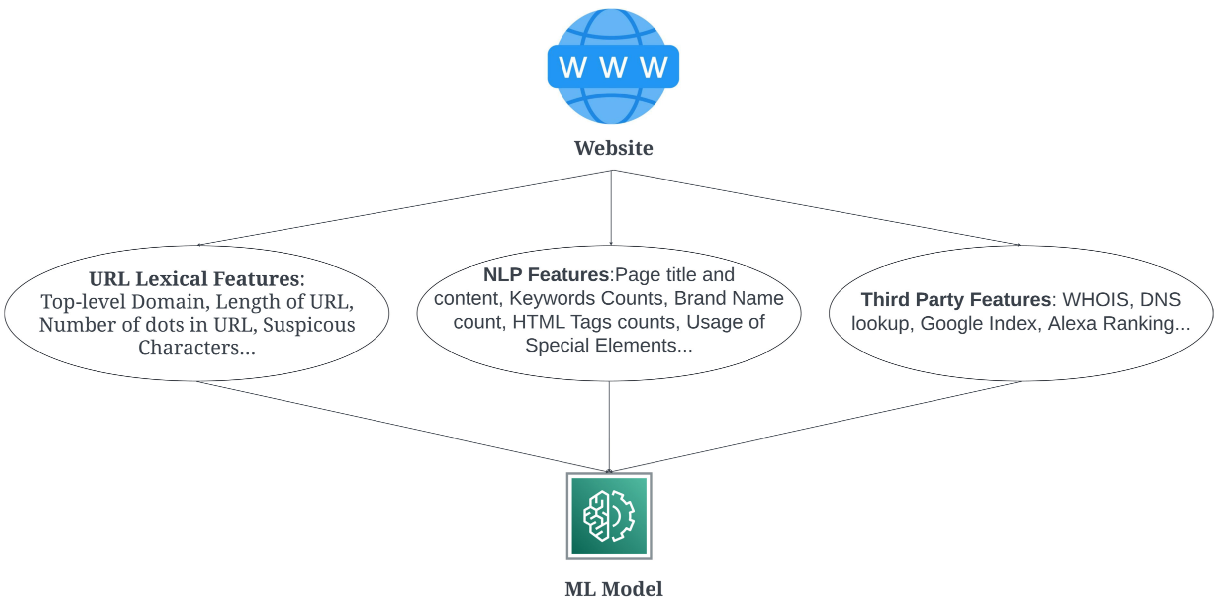

Website
URL Lexical Features: Top-level Domain, Length of URL, Number of dots in URL, Suspicous Characters...
NLP Features:Page title and content, Keywords Counts, Brand Name count, HTML Tags counts, Usage of Special Elements...
Third Party Features: WHOIS, DNS lookup, Google Index, Alexa Ranking...
ML Model

FIGURE 13. ML-enabled heuristic methods overview.

TABLE 5. ML-enabled heuristic methods main characteristics.

Reference	Used Technique	Detection Mechanism	Datasets	Model	Accuracy
Sameen et al. (2020) [78]	URL Analysis	Employs an ensemble of machine learning classifiers on lexical URL features and a URL Hit method for tiny URLs, enabling efficient real-time detection	Alexa + Phishtank + AI-Generated URLs using DeepPhish	Voting from different ML models	98%
Sonowal et al. (2020) [80]	Multilayer model to detect phishing	Combines whitelist layer, feature layer, lexical signature, string matching, and accessibility score assessments	PhishTank + Phishload	Multi-Filter Model	92.72%
Sanchez-Paniagu et al. (2022) [82]	URL, HTML, Hybrid and Web Technology features analysis	An independent phishing detection mechanism using URL, HTML, and web technology features, leveraging a LightGBM classifier and 54 unique features.	PILWD-134K	LightGBM	97.95%
Chinnasamy et al. (2022) [84]	URL and Traffic Analysis	URL and webpage attributes analyzed through machine learning for robust phishing identification	Custom dataset	RF, SVM, Genetic	94.73% (Genetic)
Saha et al. (2023) [85]	URL and Website Content Pattern Identification	An advanced two-layer StackModel synergizes multiple ML algorithms with unique URL and HTML-based features of phishing with FWBs	URLs from Twitter and Facebook	GBDT, XGBoost, and LightGBM	97%
Bahaghighat et al. (2023) [86]	URL Feature Extraction	Utilizes a machine learning ensemble of six algorithms, optimized with feature selection, dataset balancing, and feature scaling	A public phishing dataset using SMOTEENN to balance the dataset	LR, KNN, NB, RF, SVM, and XGBoost	99.2% (XGBoost)
Adap et al. (2023) [88]	URL Feature Extraction and Analysis	Employs a URL-based detection mechanism utilizing XGBoost and Random Forest classifiers to identify phishing websites	PhishTank, OpenPhish and a public dataset on Mendeley	XGBoost, RF	96.51% (XGBoost)
Talukder et al. (2024) [90]	NB URL Vectorization	Employs NB classifiers (Bernoulli, Multinomial, and Complement) on vectorized URL data to identify phishing sites	A phishing site prediction dataset on Kaggle	Bernoulli NB, Multinomial NB, and Complement NB	96% (Multinomial NB, and Complement NB)

by matching detected logos with legitimate brands in the reference list and determines whether the page is a Credential-Requiring Page (CRP) if there is a match in the brand. If no credentials are identified, PhishIntention further examines the potential region leading to CRP with object detection. To detect phishing in the wild, PhishIntention is designed as a distributed system that consists of a crawler that keeps crawling the URLs fed from CertStream [97]. Moreover, multiple virtual private networks (VPNs) are used to access the same URL to mitigate the effect of cloaking techniques. Finally, The URL, along with the HTML code and its screenshot, are stored in a database node.

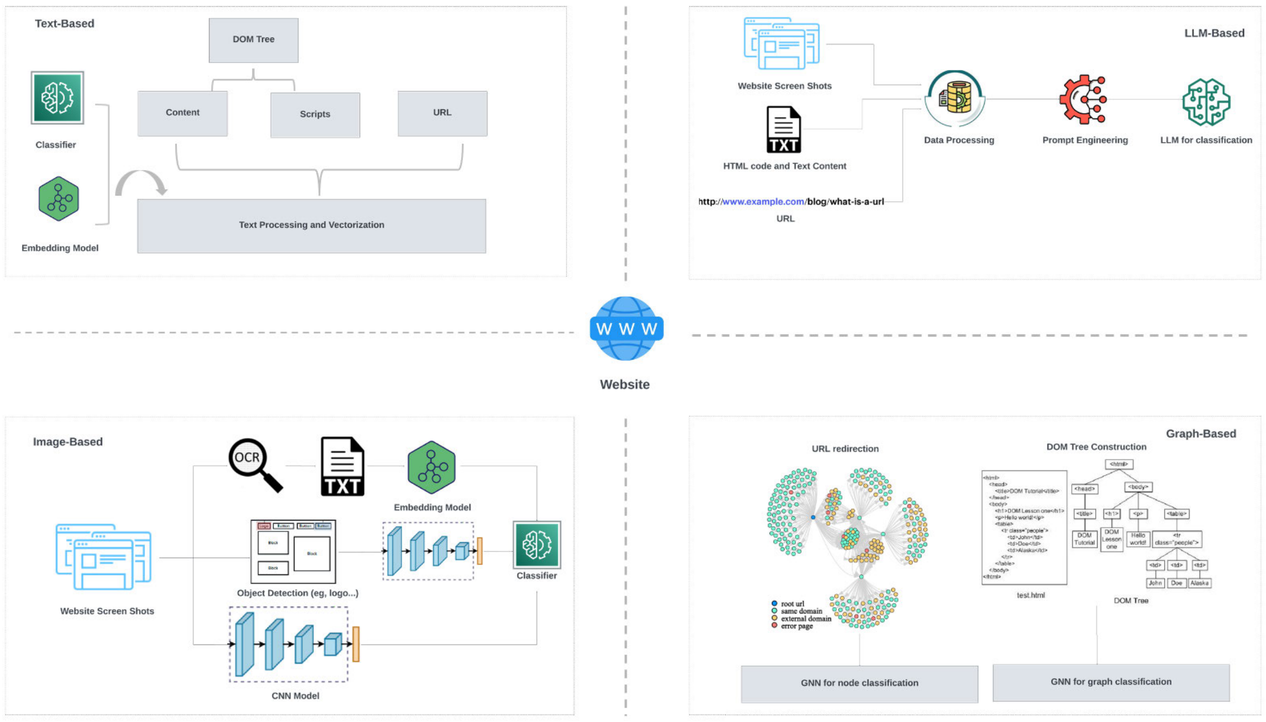


FIGURE 14. An overview of deep learning methods in phishing website detection.

As a result, the system significantly reduced false alerts by 86.5% compared to Phishpedia and identified 1,942 new phishing sites, including 1,368 not previously reported by VirusTotal [74].

Liu et al. [98] introduce *DynaPhish*, a novel framework designed to enhance reference-based phishing detection systems by addressing two major limitations: the inability to detect phishing attempts targeting emerging or unknown brands and the failure to recognize phishing on webpages without explicit brand representations (i.e., brandless phishing pages). DynaPhish dynamically expands its reference list by learning new brand representations, such as logos, domains, and keywords, from real-world phishing data. This dynamic reference expansion allows the system to detect phishing websites that target previously unseen brands, significantly enhancing its adaptability and coverage. For phishing pages lacking explicit brand information, the system employs a counterfactual interaction framework, which simulates user interactions to infer phishing intent based on behavior. The detection pipeline of DynaPhish integrates both visual and textual feature extraction. Visual features, such as logos, are identified using image recognition models, while textual features are extracted from the web page's HTML content. For known brands, these features are compared against a dynamically updated knowledge base. In cases of brandless phishing pages, the counterfactual interaction framework evaluates phishing intent through simulated user actions. The methodology is evaluated with the creation of the DynaPD dataset, which contains 6,344 interactable phishing web pages, including those targeting well-known, lesser-known, and brandless phishing pages. The results demonstrate that DynaPhish's reference expansion technique significantly improves phishing detection, particularly for emerging and lesser-known brands, achieving a recall and precision rate of 100% on the DynaPD dataset. Moreover, the Brand Knowledge Expansion module significantly enhances both DynaPhish and the baselines, which highlights its effectiveness.

Niu and Wu [99] presents a computer vision-based phishing detection framework designed to identify phishing websites by analyzing their visual components, such as logos and input boxes, and comparing them with those of legitimate brand websites. The authors focus on extracting both semantic and spatial features from website screenshots. They start by extracting key visual components such as logos and input boxes from website screenshots using a dual-path feature aggregation model. Leveraging a reference of legitimate brand logos, they feed the suspicious logo and the reference logos into a Logo recognition module to assess their similarity using the cosine similarity of their representations. A webpage is flagged as phishing if the visual elements, particularly the logos, do not correspond to those of the legitimate brand, i.e., the similarity between the logos exceeds the threshold. The findings demonstrate that by leveraging dense feature aggregation and self-attention, visual-based phishing detection frameworks can be further improved. The authors evaluated their work on a dataset of 29,496 phishing URLs and 29,951 benign URLs as well as 30,000 Alexa benign webpages from Phishpedia [94], achieving an accuracy of 97.7% and a recall of 95.0%.

Wang et al. [100] identify phishing websites by comparing both global and local visual features of phishing pages to legitimate websites. The work consists of combining global visual features, such as the overall layout and structure of the entire webpage, with local visual features, particularly logos and input boxes, which are often mimicked in phishing attacks. Given a URL, the corresponding webpage screenshot is captured within a sandbox environment. The authors divided the phishing detection task into two components: object detection and image classification. First, the logo region in the screenshot is localized using an object detection module (LogoLoc). This region is then cropped from the webpage and resized to the predefined size. Then, a feature extractor is employed to compute the region's feature, which is concatenated with the feature of the full webpage. Next, the concatenated feature is used to predict webpage identity. Finally, a prediction result is obtained by comparing the predicted URL with the input URL.

2) TEXT-BASED

Text-based methods for phishing detection can be viewed from two perspectives. The first is URL-based methods, which examine various fields of a URL, such as the domain name, protocol, subdomains, and paths, by treating the URL as a string. The second focuses on webpage source code and text content, where attackers manipulate resources within the webpage by tampering with HTML code by using advanced techniques to target specific elements of a legitimate webpage and replace them with malicious content. The following methods are compared in Table 7.

In this matter, Xiao et al. [101] address the challenge of detecting phishing websites in imbalanced datasets by leveraging CNN and Multi-Head Self-Attention (MHSA) mechanisms. To tackle the issue of dataset imbalance, they employ a Generative Adversarial Network (GAN) to generate synthetic phishing URLs to balance the dataset, making the number of phishing websites equal to legitimate websites. The authors combine CNN to extract local features from character-level URL sequences and MHSA to capture global dependencies by assigning attention weights across the entire URL. The classifier consists of the input block, the attention block, the feature block, and the output block. The findings indicate that this approach performs effectively on imbalanced datasets. The model achieved high performance on various combinations of the dataset, which consists of 64,030 URLs collected from 5000 Best Websites homepage [102] and PhishTank [26]. The authors experimented with different training set combinations, achieving the best results when the ratio between real-world legitimate URLs and real-world phishing URLs was close to 7, resulting in an accuracy of 97.20%.

Aljofey et al. [103] introduce a hybrid feature set that incorporates URL character sequences, hyperlink information, and plaintext, as well as noisy HTML content from webpages to enhance detection accuracy. The methodology involves extracting three key types of features from phishing websites sourced from PhishTank [26] and legitimate websites from Stuff Gate. These include URL character sequence features, which analyze the structure of the URL; textual content from the HTML body; and hyperlink-related features, focusing on internal and external links, along with suspicious login forms. These extracted features are then concatenated into a feature vector, which is used to train an ML classifier to distinguish between phishing and legitimate websites. The experiments showed that XGBoost outperforms different ML classifiers, such as LR and RF, with an accuracy of 96.76% and an F1-score of 96.38%. These results highlight the strength of combining URL and HTML features to enhance phishing website detection performance.

Bozkir et al. [104] present *GramBeddings*, a DL model designed to detect phishing URLs by utilizing n-gram embeddings. The paper proposes a method that eliminates the need for pre-trained models, handcrafted features, or external services, instead relying entirely on the n-gram structure of URLs to detect phishing patterns. The work involves tokenizing URLs into n-grams, which are then transformed into dense vector representations through an embedding layer. The authors apply a CNN layer to capture the local syntactic patterns, followed by an LSTM layer to detect long-range dependencies in the URL sequences. In addition, An attention mechanism is incorporated to assign greater weight to critical n-grams, enhancing the model's ability to focus on the most relevant parts of the URL. The final output is processed through a fully connected layer and classified using a softmax classifier. After trying multiple configurations for the n-gram corpus size, sequence length, embedding dimension, attention size, and LSTM cell size, the model achieved an accuracy of 98.27% on their proposed dataset GramBeddings-DS where 400K phishing and 400K legitimate samples are involved.

Gopali et al. [105] explore the effectiveness of various sequential DL models in identifying phishing websites by treating URLs as sequences. The study evaluates models such as Multi-Head Attention, Temporal Convolutional Networks (TCN), LSTM, and Bidirectional-LSTM (BiLSTM), focusing on how these models leverage contextual features within URLs to detect phishing attempts. The methodology involves tokenizing URLs from a dataset of public GitHub repository [106] of 73,575 URLs and splitting the data into training and testing sets. Each URL is converted into numerical vectors via an embedding layer and then passed through model-specific layers to capture sequential dependencies. The models include Multi-Head Attention for focusing on different parts of the sequence, TCN for temporal relationships, and LSTM/BiLSTM for learning long-range dependencies in the URL structure. The findings show that

TABLE 6. **Image-based methods main characteristics.**

Reference	Used Technique	Detection Mechanism	Datasets	Model	Accuracy (*Precision*)
Abdelnabi et al. (2020) [92]	Visual Similarity Analysis	Triplet CNN compares screenshots of websites	VisualPhish dataset	triplet CNN	88.6% (Top-5 match rate)
Lin et al. (2021) [94]	UI Detection and Logo Matching	Object detection and Siamese network for logo matching	OpenPhish Premium Service, Alexa's top-rank list	Faster RCNN based on Detectron2 + Resnetv2	99.2% (Identification Rate)
Trinh et al. (2022) [95]	Visual Feature Extraction and Mapping	Uses VGG16, ResNet50, and others for high-level visual feature extraction	VisualPhish Dataset	VGG16-LR	88.68%
Liu et al. (2022) [96]	Interaction and Visual Analysis	Combines visual layout extraction with dynamic interaction simulation	Phishpedia dataset	Faster RCNN + ResNetV2-50 + OCR Encoder	95.0% (CRP Classifier)
Liu et al. (2023) [98]	Dynamic Reference-Based Analysis	Learns brand representations and infers phishing intent for brandless pages	DynaPD dataset	Faster RCNN + ResNetV2-50 + OCR Encoder	*100%* (Precision for PhishIntention + DynaPhish)
Niu et al. (2024) [99]	Dual-Path Feature Aggregation	Analyzes logos and input boxes for visual similarity	Phishpedia dataset	Faster RCNN + Feature Pyramid Networks (FPN) + ResNetV2	*97.7%* (Precision)
Wang et al. (2024) [100]	Global and Local Visual Analysis	Combines global layout and logo-specific analysis	Phishpedia + VisualPhish dataset	ResNet50	95% (Matching Accuracy on PhishPedia dataset)

treating URLs as sequences significantly enhances phishing detection accuracy, with the BiLSTM model outperforming the other approaches in terms of phishing detection rate.

Do et al. [107] combines DL classifiers with transformer-based models for URL phishing detection. Their methodology involves preprocessing and tokenizing URLs at both the character and word levels to capture key structural and semantic patterns. The model incorporates RasNet to extract low-level features, TCN to analyze sequential relationships in the URL, and MHSA to focus on various parts of the URL simultaneously. These features are combined with outputs from MPNet, a pre-trained transformer fine-tuned on phishing datasets, to enhance the contextual understanding of URLs. The final concatenated features are passed through a fully connected layer and classified using a softmax classifier. The hybrid model was tested on four datasets: Ebbu2017 [108], PhishCrawl, 420K-PD [109] and 1M-PD [109], achieving an accuracy rates of 99.23%, 98.74%,, 99.11% and 99.71%, respectively. The paper highlights that the integration of RasNet, TCN, and MHSA for feature extraction, along with MPNet for contextual comprehension, enables the model to effectively capture both structural and semantic patterns in URLs.

Çolhak et al. [110] combine the Multilayer perceptron (MLP) model and two pre-trained NLP models to handle numeric and tabular data and to analyze the textual information such as page titles and content, respectively. The textual features are processed using the pre-trained NLP models, while numeric features like hyperlink counts and CSS/JS file counts are fed into the MLP. The outputs from both the NLP models and MLP are fused into a unified embedding, which is then classified using a linear classifier to determine whether the webpage is phishing or legitimate. The authors conducted extensive experiments with different NLP models such as ALBERT, BERT, CANINE, and RoBERTa on the MTLP dataset [111] and Aljofey's dataset [103] sourced from OpenPhish [71] and Alexa [27]. The findings demonstrate that this hybrid approach of combining NLP models, in particular CANINE and RoBERTa, with an MLP significantly achieved an accuracy of 97.18% and an F1 score of 96.80%.

3) GRAPH-BASED

Graph-based approaches to phishing detection have gained significant attention for their ability to model and analyze the complex relationships between various components of web pages. These methods leverage the structural properties of the web, including hyperlinks, URLs, HTML elements, and even images, to more effectively detect phishing attempts. The graph-based methods used techniques are highlighted in Table 8.

Futai et al. [112] propose a method to detect phishing by analyzing real IP traffic from Internet Service Providers (ISPs) using Graph Mining. The proposed model detects phishing threats that traditional URL analysis might miss by analyzing the relationship between URLs and their visitors and tackles the challenges of frequently updated phishing sites. The algorithm iteratively refines node reputation scores, and a reputation threshold is set to identify phishing attempts. The detection model uses traffic data from a large ISP to build an AD-URL graph of user-URL interactions. Each node in the graph is given a reputation score to detect phishing. The Belief Propagation (BP) algorithm, based on the Markov Random Field (MRF) model, refines these scores by passing information between nodes. Lately, the MRF has treated each node's reputation as influenced by its neighbors, and BP calculates the likelihood of phishing through message passing. For unknown nodes' reputations, an initial score is attributed based on prior knowledge. The model was tested on raw traffic data from a large ISP, achieving over 80% precision on real traffic data. In a small dataset, the proposed method reaches a 92% true positive rate with a 3% false positive rate, and in a large dataset, it reaches a precision of 95.3% with a 0.525 belief threshold. The method only uses user-website interactions, requiring minimal data and computational resources, making it both efficient and widely applicable.

Tan et al. [13] propose a hyperlink-based method structured around three key phases: collecting page linking data, constructing the web graph, and extracting features. The web graph consists of three node layers: the query webpage (L_0), first-order neighbors (L_1), and second-order neighbors (L_2). To distinguish between nodes, vertices on L_0 and those on L_1 and L_2 with the same domain name as the query webpage are labeled red, while the remaining vertices on L_1 are labeled blue, and those on L_2 are labeled green. In addition, the authors identified several phishing patterns and derived 17 novel graph features from the query's ego network. These features include the percentage of blue, red, and green vertices, the in-degree and out-degree of the query vertex (which respectively measure the number of hyperlinks pointing to and from the query webpage, including self-loops), and the graph's density. Following this, these features were used to train classifiers for phishing detection. In the evaluation process, the authors selected phishing URLs from PhishTank [26] and OpenPhish [71], legitimate URLs from Alexa [27] and the Common Crawl archive [113]. SVM, NB, C4.5, and RF classifiers were tested using the 17 proposed graph features, achieving accuracies of 88.30%, 91.30%, 97.80%, and 97.70%, respectively. In contrast to graph features, the accuracies of these classifiers with conventional features were 94.40%, 93.60%, 94.80%, and 95.70% in [114]. Moreover, Combining both graph and conventional feature sets resulted in 96.90%, 95.40%, 97.60%, and 98.00%. Therefore, the proposed method can be a promising enhancement to the existing machine learning-based phishing detection methods.

In [115], Ouyang et al. introduce a GNN-based phishing webpage detection method that uses the inherent structural information of HTML to capture long-range semantics. The approach combines the strengths of both RNN and GNN, enabling a deeper understanding of the intent behind HTML code. The model takes HTML documents as input and outputs its maliciousness probability. For each HTML document, a small graph is constructed based on its inherent DOM structure. To construct the HTML graph, a node represents an HTML element and is assigned features extracted from the HTML DOM. Then, the Tag features are converted into learnable vectors, while attributes are processed through an RNN to generate vector embeddings. For webpage content in the DOM tree, it is as unique nodes with a "content" tag, where text in leaf nodes is processed as an attribute. A Topology Adaptive Graph Convolutional Network (GCN) is subsequently applied to capture long-range semantic relationships within the graph. The resulting graph-level representation is then classified using a fully connected layer. The phishing dataset used for evaluation was sourced from PhishTank [26], and OpenPhish [71], and Tranco [116]. Notably, the findings indicate that the use of both RNN and GCN can better enhance the models' understanding of the intention of HTML documents and improve detection performance.

In [117], Tchakounte et al. introduce a URL-based method *Crawl-Shing*. Crawl-Shing enhances traditional crawling techniques by using isomorphic graphs to evaluate the similarity between two web pages based on their Maximum Common Subgraph (MCS). Unlike standard crawlers, which simply gather web content from search engines, Crawl-Shing incorporates a relevance evaluation module that focuses on identifying phishing content through graph analysis. This method adapts the typical crawler architecture to better align with the graph-based approach. The system begins with a keyword search, selecting the most relevant URLs, which are sent to a queue. These URLs serve as input to the downloader, which requests the web page and receives the HTML document. The system extracts URLs and their main content, passing them through a duplicate detector. Lately, only verified URLs have been pre-processed and transformed into graphs, which are compared with referral URLs from the search engine. Once this comparison is complete, the next step involves finding the largest common subgraph and evaluating its similarity for storage and presentation. The evaluation compared three focused crawlers: Vector Space Models (VSM) with cosine similarity, Breadth First Search (BFS), and a graph isomorphism-based crawler) using Harvest rate, average similarity, and precision metrics. Crawl-Shing outperformed the BFS crawler in harvest rate, indicating that this approach can be a potential solution against phishing.

Bilot et al. [118] introduced *PhishGNN*, a DL framework that uses GNNs by leveraging website hyperlink structures and other manually designed features. PhishGNN distinguishes phishing from legitimate sites by emphasizing that phishing links often lead to different domains, while legitimate links typically stay within the same domain. Hence, website classification (phishing/benign) is treated as node classification. Each node represents a URL, and edges represent links from that URL. The webpage to classify is represented as a root graph, where the root is the webpage itself. PhishGNN consists of two main steps: pre-classification, after which the feature matrix X(features of the root and child URLs) is transformed to a vector $\hat{X}$ formed of zeroes and ones for legitimate and phishing predictions, respectively and conducting a message-passing following the same propagation rule as the original GCN propagation rule. For evaluation, various GNNs and traditional ML methods are trained and tested on the dataset extracted from such as

TABLE 7. Text-based methods main characteristics.

Reference	Used Technique	Detection Mechanism	Datasets	Model	Accuracy
Xiao et al. (2021) [101]	URL-Based and Content Analysis	Combines CNN with a self-attention mechanism for URL analysis and GAN for synthetic URL generation	PhishTank + 5000 Best Websites Homepage	Self-attention CNN	95.6%
Aljofey et al. (2022) [103]	URL and HTML Feature Extraction	Analyzes URL structure, HTML body content, and hyperlink patterns	PhishTank and Stuff Gate	XGBoost	96.76%
Bozkir et al. (2023) [104]	N-Gram URL Encoding	Uses n-gram embeddings and LSTM layers for pattern detection	GramBeddings-DS	Ensemble Model(Character Embedding+ N-Gram Embedding)	98.27%
Gopali et al. (2024) [105]	Sequential Model for URL	Employs various DL models such as Multi-Head Attention and BiLSTM for learning URL patterns	Github Repository	BiLSTM (With best performance)	98%
Do et al. (2024) [107]	Transformer-Based URL Detection	Uses transformer models with TCN for sequential analysis and URL embedding	Ebbu2017, 1M-PD, PhishCrawl, 420K-PD	RasNet-TCMA-MPNet (RasNet + TCN-MHSA + MPNet)	99.71% (on 1M-PD)
Çolhak et al. (2024) [110]	Hybrid Textual and Numeric Analysis	Combines pre-trained NLP models with MLP for text analysis	Aljofey's dataset + MTLP dataset	MultiText-LP	97.18%

OpenPhish [71], PhishTank [26], and Alexa [27] to find out the combination with the best performance. Consequently, the approach achieved an accuracy of 99.7% by integrating RF as a pre-classification step with GCN$_2$ for message-passing.

Another method called *PhishDet* introduced by Ariyadasa et al. [119], which is an innovative detection framework leveraging Long-term Recurrent Convolutional Networks (LRCNs) and GCN to identify phishing websites by analyzing both URL and HTML features. PhishDet achieves a high detection accuracy of 96.42% with a real-world dataset on Mendeley [89] and demonstrates efficacy against zero-day attacks with an average detection time of 1.8 seconds. PhishDet's unique architecture, comprising URLDet and HTMLDet sub-models, automates feature selection and learns URL and HTML characteristics, which enhances adaptability to emerging phishing tactics. Specifically, URLDet processes URLs using an LRCN-based model, normalizing and encoding URL content for effective feature extraction, while HTMLDet employs a GCN-based approach to analyze HTML structures, converting them into graph representations without expert intervention. These two models, trained independently, are then combined into a single framework, which utilizes a final dense layer to classify input as legitimate or phishing. PhishDet achieves an impressive 99.53% F1-score on a public benchmark dataset [120], though periodic retraining is necessary to maintain peak performance. However, PhishDet is subject to adversarial attacks.

Commonly, researchers in this field tend to propose architectures that rely on textual features and heuristics (Hyperlinks, URLs, HTML…). Lindamulage et al. [14] present a GNN-based pioneering solution for phishing website detection based on RGB images of the websites by implementing VisionGNN architecture, which is a GNN-based approach. Inspired from [121]. The idea is to represent an image as a graph to express the pixels as nodes of a graph, and edges link neighboring nodes (pixels). Like a classical GNN, VisionGNN processes the graph by updating node features through message passing and aggregation to produce a final image-level representation for the classification task. To do so, the features were extracted through the following process: RGB images resized to 224×224 pixels and divided into patches, with each patch converted into feature vectors. The images included visual elements like layout, color, and texture, as well as node and edge features, where patches were treated as nodes in a graph and edges captured relationships between them. All four versions of the Pyramid architecture of the VisionGNN (tiny, small, medium, big) were tested on The VisualPhish dataset [92]. The accuracy values for the tiny, small, medium, and big models are respectively 93.5%, 97.4%, 91.8%, and 93.5%. This study represents the first application of VisionGNN for phishing detection, highlighting the effectiveness of GNN-based techniques in analyzing the visual features of phishing websites.

4) LLM-BASED

Phishing detection tools typically rely on pre-designed datasets of phishing web pages to evaluate the robustness of detection models. However, these tools often require learning targeted brands and adjusting their detection logic

TABLE 8. Graph-based methods main characteristics.

Reference	Used Technique	Detection Mechanism	Datasets	Model	Accuracy *(Precision)*
Futai et al. (2016) [112]	Analyzes ISP traffic through Graph Mining	Analyzes user-URL interactions with BP and reputation scoring	ISP real traffic	BP	*95.3%* (Precision, on big dataset with BP threshold 0.525)
Tan et al. (2020) [13]	Hyperlink Graph Analysis	Constructs web graphs based on first and second-order hyperlink relationships, deriving 17 novel graph features for phishing detection	PhishTank, OpenPhish, Alexa, and Common Crawl archive	SVM, NB, C4.5, RF	97.8% (C4.5 with graph features)
Ouyang et al. (2021) [115]	DOM Structure Graph Analysis	Uses RNN and GCN for long-range semantic relationships in HTML documents	PhishTank, OpenPhish, Tranco	RNN, Topology Adaptive GCN	95.50%
Tchakounte et al. (2022) [117]	Isomorphic Graph Analysis	Develops novel crawling techniques by relying on isomorphic graphs to evaluate the similarity between two web pages	Custom dataset	N/A	N/A
Bilot et al. (2022) [118]	Hyperlink redirection analysis	Constructs a redirection graph rooted from the webpage to classify and assign lexical, content and domain features to each node in the graph	OpenPhish, PhishTank, and Alexa	GCN	99.7%
Ariyadasa et al. (2022) [119]	URL and HTML code encoding	Combines LRCN and GCN to detect phishing websites using URL and HTML features	Phishtank, Web2Vec and offline anti-phishing dataset, Google search engine, A phishing dataset on Mendeley	LRCN + GCN	96.42% (A phishing dataset on Mendeley)
Lindamulage et al. (2023) [14]	Image-to-Graph Transformation	Transforms RGB images into pixel-based graphs using VisionGNN	VisualPhish Dataset	VisionGNN	97.4% (Small Model)

accordingly to identify phishing sites. In addition, they generally fail to analyze in detail the psychological manipulation induced by social engineering techniques. In this context, open-source LLMs have shown promise as content moderation tools in various areas of social computing and security, particularly in detecting phishing websites. The performance of these methods is demonstrated in Table 9.

Su and Su [15] evaluate BERT for detecting malicious URLs by leveraging the model's ability to understand the semantic relationships within URLs. By using URL string-based datasets, BERT tokenizes the URLs to capture semantic relationships within their structure by taking into account the significance of letters within different vocabularies. In datasets containing only URL features, RF is used to select the most important features, which are then converted into a feature string for processing by BERT. Then, BERT is fine-tuned for classification, utilizing its self-attention mechanism to analyze both URL strings and feature strings. The model achieved 98.78% accuracy on the dataset on Kaggle [122], 96.71% on a dataset available on GitHub [87], and 99.98% on the ISCX 2016 dataset [123] for binary classification, demonstrating its ability in capturing both character-level and feature-level semantics.

Li et al. [16] presented *KnowPhish* that addresses two key limitations of current reference-based phishing detectors (RBPDs): the reliance on manually curated brand knowledge bases (BKBs) and the lack of consideration for textual information alongside visual logos. KnowPhish aims to expand the scalability and brand coverage of BKBs while integrating both visual and textual analysis for more comprehensive phishing detection. The work involves building a comprehensive large-scale multimodal brand knowledge base containing 20k brands. KnowPhish enhances brand information collection by using category-based searches for high-value industries and popularity-based searches for domain rankings. This data, including logos, domain names, and aliases, is collected from sources like Wikidata, Google Image Search, and Whois services. Then, the detection system, KnowPhish Detector (KPD), integrates an LLM to generate webpage summaries, including brand intention and credential-requiring elements, and uses XLM-RoBERTa to classify whether a webpage requires credentials based on its summary. The system also uses a Logo Brand Extractor (LBE) to detect logos and a Text Brand Extractor (TBE) to identify brands in text when no logo is present. To ensure the webpage domain matches the legitimate brand's domain, KnowPhish conducts domain checking using the captured URL. The findings highlight that KnowPhish significantly enhances phishing detection, especially for logo-less phishing pages. On the TR-OP dataset, which is a manually constructed dataset with 5000 benign samples collected from Tranco [116] and 5000 phishing samples from Openphish [71], KPD with KnowPhish achieved an F1 score of 92.05% and a recall of 86.90% and an accuracy of 92.49%, outperforming alternatives like DynaPhish. Additionally, KnowPhish demonstrated superior runtime efficiency, operating 50 times faster than DynaPhish when integrated with Phishpedia and PhishIntention.

Lee et al. [124] explore the use of multimodal LLMs, such as GPT-4, GeminiPro 1.0, and Claude3, for detecting phishing webpages through brand identification and domain verification. The proposed system leverages the ability of LLMs to analyze both visual and textual content from a webpage by incorporating a two-phase detection process. In the first phase, the LLM analyzes webpage screenshots and HTML content to identify the brand associated with the page, extracting key visual elements like logos, page themes, and textual descriptions that signify the brand. In the second phase, the identified brand is cross-verified with the domain name in the URL. If the domain does not match the official domain of the brand, the page is flagged as phishing. This multimodal approach not only detects phishing attempts but also provides users with explanations and evidence supporting the detection decision, enhancing transparency. The paper showed that the F1 score achieved by GPT-4 and Claude3 are approximately 92% and 90%, respectively, while Gemini lags behind the other two to achieve a similar detection rate (as GPT and Claude), and the precision drops by more than 15%. The dataset is sourced from OpenPhish [71] and Tranco [116].

Kulkarni et al. [125] investigate the robustness of phishing detection models, including ML, DL, and LLMs models, when confronted with adversarial phishing webpages. The study introduces *PhishOracle*, a tool designed to generate adversarial phishing webpages by embedding a variety of phishing features into legitimate websites. PhishOracle aims to test the vulnerabilities of popular detection models such as Stack Model, Phishpedia, and Gemini Pro Vision by exposing them to these crafted adversarial attacks. PhishOracle's automated phishing webpage generation by manipulating HTML tags such as <form>, <a>, and <script> on legitimate webpages to introduce malicious behavior. It also includes disabling anchor (<a>) tags, replacing hypertext references (href) with lookalike characters, injecting pop-up login forms, and altering form actions to redirect user credentials. In addition, PhishOracle generates phishing URLs by modifying legitimate domain names using homoglyphs (e.g., substituting "o" with "0" or "l" with "1") and adding deceptive prefixes and suffixes like "login-" or "-secure" to make URLs appear legitimate. The Stack model, which relies on URL and HTML-based features, uses classifiers such as GBDT, XGBoost, and LightGBM to detect phishing webpages. When tested on the evasion dataset, it achieved an accuracy of 84.92%, while it achieved 98.54% on the clean dataset. However, it showed a significant drop in performance when tested against PhishOracle-generated adversarial phishing webpages, highlighting the vulnerability of traditional detection models to adversarial phishing techniques, in particular when minor content-based features are manipulated. Similarly, Phishpedia, which combines Faster-RCNN and ResNetV2 for logo detection and brand identification, achieved a precision of 76.40%. However, Gemini Pro Vision outperformed the other models with an accuracy of 95.64% on brand identification.

Roy and Nilizadeh [126] introduce *PhishLang*, an open-source, lightweight framework designed to detect phishing websites with explainable threat warnings using LLMs. PhishLang focuses on analyzing and learning contextual patterns in website source code. PhishLang starts with source code parsing, where it isolates relevant tags like <h1>, <a>, <form>, and <meta>, while excluding aesthetic elements like layouts and styles. To avoid exceeding LLM token limitations, the source code is divided into chunks using a sliding window technique. The authors evaluate multiple open-source language models, ultimately selecting MobileBERT for its efficiency and balance between speed and memory usage. PhishLang also integrates explainable blocklisting by leveraging GPT-3.5 Turbo to generate contextual phishing warnings, providing real-time detection of phishing websites on the client side with a proposed browser extension. The framework excels in resource efficiency, using up less memory and less storage compared to computationally intensive DL models. When compared to other phishing detection models like VisualPhishNet, PhishIntention, StackModel, and URLNet, PhishLang achieved an F1, accuracy score of 0.94 and a precision of 0.93 with less prediction time and small artifact size, offering the best trade-off between speed, memory efficiency, and accuracy.

Koide et al. [127] propose *ChatPhishDetector*, which leverages LLMs for phishing site detection. The system combines web crawling with tailored prompt engineering to analyze websites in both text-only and multimodal modes. It gathers data from websites using the Chrome DevTools Protocol, collecting URLs, HTML after JS execution, and screenshot images to ensure comprehensive analysis. The data is then processed into tailored prompts based on the Chain of Thought (CoT) prompting technique, which enables the LLM to act as a programmer security expert and control the inference process. The model is tasked with analyzing social engineering techniques, extracting brand names, detecting phishing activity, and producing results in JSON format. ChatPhishDetector manages the token limitations of LLMs by simplifying HTML and OCR-extracted text while retaining key information. In addition, it removes unimportant HTML elements and reduces OCR-extracted text by eliminating sentences with the smallest font size. The experimental results on a dataset of 1000 phishing and 1000 non-phishing websites (sourced from OpenPhish, PhishTank, and posts from Twitter using CrowdCanary [128]) showed that GPT-4V achieved the most outstanding accuracy of 99.2% and precision of 98.7% and a recall of 99.6%, while GPT-4 (normal mode) also performed well with 98.3% precision and 98.4% recall and accuracy. Notably, the authors emphasize the utility of LLMs, which eliminates the need for additional model training, as the LLMs handle phishing detection directly.

To further strengthen the approach proposed by Koide et al., Schesny et al. [129] propose an extension of *Chat-PhishDetector* by adapting the CoT prompting technique, incorporating URLs, and introducing additional parameters for more accurate detection. The work explores the effects of adding social pressure in prompts, such as threats of job loss for incorrect classifications. The authors further asses websites that were previously identified as non-phishing and reanalyze them through ChatGPT to classify each site based on its phishing score. They also examine whether it is necessary to use CoT prompting in advance or if a simple phishing identification question suffices. The findings of this extension include a detailed analysis of 59 phishing websites that had not been previously identified. After excluding 48 outdated or browser-detected websites, the remaining 11 were tested in three different scenarios. In Scenarios 2 (threat of job loss) and 3 (only the known content of the website was asked in combination with a simple question as to whether it was phishing or not), five of the eleven websites were successfully identified as phishing, while Scenario 1 (works as the baseline of previous work [127]) identified four websites, with one marked as "unknown." In addition, it was observed that certain phishing webpages with identical scores were classified differently, and the weighting of factors influencing the phishing score remained unclear. Consequently, the authors recommended that ChatGPT be integrated with additional security strategies and technologies to improve its reliability and interpretability.

Finally, to address the limitations in existing reference-based phishing detectors, which typically rely on predefined reference lists, Liu et al. [130] develop a novel reference-based phishing detector *PhishLLM*, that operates without an explicit predefined reference list by leveraging LLMs to infer brand-domain relationships dynamically. *PhishLLM* takes a webpage and its domain as input and reports its analysis based on the consistency between the brand and domain and the presence of a credential-taking intention on the webpage. First, it captures a screenshot of the webpage to circumvent potential HTML obfuscation and retrieves the logo on the screenshot. This logo is then described using OCR and image captioning models, generating a textual prompt to infer the associated brand using the vast brand knowledge of the LLM. To ensure brand-domain consistency. Next, PhishLLM predicts CRPs by converting the webpage into a textual description (parsing its screenshot into a sequence of phrases) and analyzing it using CoT reasoning. If credential-taking intentions are identified or inconsistencies are detected between the brand and domain, the system flags the webpage as phishing. In cases where additional verification is required, a visual-language model ranks and interacts with clickable elements to determine whether they lead to a credential-requiring page. Besides, for webpages hosted on well-known domains, a popularity validation mechanism is employed to ensure that legitimate hosting services are not mistakenly flagged as phishing sites. *PhishLLM* employs PaddleOCRV3 as the OCR model, BLIP2 as the image captioning model and "GPT-3.5-turbo-16k" for brand recognition and CRP prediction. Overall, *PhishLLM* outperforms the baselines such as Phishpedia and PhishIntention, achieving 100% precision and 75.01% recall on the DynaPD benchmark. Also, it recognized 65% of brands in the top-ranked and lower-ranked Alexa dataset, outperforming static reference lists that only detect 5%. The experimental results validate that *PhishLLM* dynamically decodes brand and domain information while analyzing webpage semantics, enabling more accurate detection of zero-day phishing attacks. The framework also incorporates mechanisms to control LLM hallucinations and validates results through search engines, leading to higher recall rates and improved detection performance compared to existing methods.

5) HYBRID

Hybrid phishing webpage detection approaches integrate existing detection techniques with features derived from both URL-based content and visuals to create a comprehensive feature vector. This vector is then used for classification. Research has demonstrated that hybrid approaches not only achieve higher accuracy but also deliver better overall performance compared to individual URL-based and content-based methods. Table 10 summarizes each paper and compares them.

Zhang et al. [131] developed *MultiPhish*, a phishing detection framework that leverages multiple modalities, including domain, favicon of the website, and URL features. MultiPhish integrates neural networks and a variational autoencoder (VAE) to optimize feature representations and enhance detection accuracy. The framework begins with collecting the necessary features, which include domain features (extracted using convolutional layers on embedded characters), favicon features (obtained via a pre-trained VGG-19 model), and URL features (such as length, special characters, and sensitive words). The domain and favicon features are fused to create a website identity representation, which is optimized using a VAE. This optimized representation is combined with the URL features to form the input for a neural network that classifies websites as phishing or legitimate. MultiPhish model achieved an accuracy of 97.79%, a precision of 96.80%, and a recall of 98.28%, surpassing methods like ANFIS and Phishdentity on a MultiPhish2020 [132] which is a dataset extracted from PhishTank [26], OpenPhish [71], and Alexa's top ranked websites [27].

Le-Nguyen et al. [133] present an automated system that compares suspicious sites to legitimate ones based on visual and semantic similarities. They first generate suspicious domain names similar to legitimate ones through a fuzzy algorithm, then extract both legitimate and suspicious

TABLE 9. LLM-based methods main characteristics.

Reference	Used Technique	Detection Mechanism	Datasets	Model	Accuracy (*Precision*)
Su et al. (2023) [15]	URL Semantic Analysis	Tokenizes and fine-tunes URLs for semantic analysis using self-attention	Kaggle, Github, ISCX 2016	BERT	99.98% (ISCX 2016)
Li et al. (2024) [16]	Multimodal input for Reference-based Phishing Detection	Integrates LLM-generated webpage summaries, domain checking, and logo/text analysis and develops an automated knowledge collection pipelines to construct knowledge base	TR-OP, SG-SCAN	GPT/LLaMA + XLM-RoBERTa	92.49% (TR-OP, Know-Phish + KPD)
Lee et al. (2024) [124]	Multimodal input for Phishing Detection	Combines text and image analysis for brand verification and detects phishing websites with domain verification based on LLM	OpenPhish + Tranco	GPT-4, GeminiPro 1.0, Claude3	*Around 90% for GPT and Claude* (Precision)
Kulkarni et al. (2024) [125]	Generates adversarial phishing webpages to test model robustness	Generates adversarial phishing websites and evaluates robustness of phishing detection of ML and LLM models	PhishOracle dataset + multiple datasets from previous works	Gemini Pro Vision	95.64% (Brand Identification)
Roy et al. (2024) [126]	Lightweight Client-Side Detection	Leverage MobileBERT to learn contextual patterns from source code and uses GPT-3.5 Turbo to create explainable warnings for users	PhishPedia dataset	MobileBERT + GPT-3.5 Turbo	96%
Koide et al. (2024) [127]	Generates prompts for LLMs based on crawled websites for phishing detection	Combines web crawling with CoT prompting for comprehensive analysis and evaluates different LLMs	OpenPhish, PhishTank, CrowdCanary (Twitter Phishing-related posts), Tranco	GPT (3.5, 4, 4V), Gemini (Pro, Pro Vision), Llama-2-70B	99.2% (GPT-4V)
Schesny et al. (2024) [129]	Adapted CoT analysis on LLM	Adapts URLs and other parameters such as social pressure with CoT prompt engineering to identify phishing websites on ChatGPT	11 undetected phishing websites from [127]	GPT-4	N/A
Liu et al. (2024) [130]	LLMs-Enabled Brand Recognition and CRP Detection	Leveraging encoded brand-domain information in LLMs without pre-defined reference list and detecting CRPs through screenshots with LLMs or VLMs	DynaPD dataset + Alexa Top 5,000 to 15,000 websites	PaddleOCRV3 + BLIP2 + GPT-3.5-turbo-16k	*100%* (Precision)

domains' HTML content and screenshots. For semantic analysis, the system is incorporated with GloVe embeddings with TF-IDF weighting to measure cosine similarity between the HTML content of suspicious and legitimate websites. On the other hand, for visual analysis, OCR is used to focus on the foreground of screenshots, comparing them using the Structural Similarity Index (SSIM). The semantic and visual similarity scores are then fed into different ML classifiers, such as SVM, DT, or RF, and the performance of these classifiers is compared. The system operates on a microservice architecture, with a web crawler continuously identifying suspicious domains and a detection module performing the analysis. The experiments achieved an accuracy of 99.82% when using RF on a dataset extracted from PhishTank and manual collection.

Chai et al. [134] introduce a multi-modal hierarchical attention model (MMHAM) designed to analyze three primary content modalities: navigation content (URLs), information content (text), and visual design (images). The objective is to automatically extract deep fraud cues from these modalities and improve the interpretability of phishing detection. MMHAM involves learning from the three modalities by first extracting and aligning features using a shared dictionary learning approach. By applying a hierarchical attention mechanism, MMHAM prioritizes important features within each modality (URL, text, and images) across all modalities, providing a more interpretable detection process. To encode the modalities, a character-level LSTM is used to encode the URL, a word-level LSTM is adopted to encode each word on the text, and ResNet-50 is employed to extract image representations. By integrating the three modalities into its hierarchical attention mechanism, MMHAM achieved an accuracy of 97.26%, a precision of 97.84%, and a recall of 96.66% on a dataset extracted from both DMOZ [135] and PhishTank [26].

Sun et al. [136] introduce *FusionNet*, a phishing detection framework that integrates multiple data modalities, such as URLs, HTML source codes, and visual features. Their goal is to learn the vector representation of these modalities for phishing classification. To achieve this, a BiLSTM with an attention mechanism is employed to capture contextual patterns for URLs that were encoded with Word2Vec at first. HTML source code features (such as the number of hyperlinks, number of empty links, and ratio of external resources…) are extracted, and then this research uses an RNN with residual connections to model the structure of the webpage, while visual features are extracted from webpage screenshots using the ResNet34 model. These features are then fused through an attention mechanism to create a unified feature representation, which is fed into a classifier to determine whether a website is phishing or legitimate. FusionNet significantly outperforms other similar multi-modal method such as *HybridDLM* (URL+HTML) [137], achieving a detection accuracy of 95.8% and an F1 score of

0.933 on the legitimate website data from a previous study *URLNet* [138] and phishing website data based on a GitHub dataset [139].

Quang et al. [140] present *Shark-Eyes*, a novel multimodal fusion framework that utilizes a multi-view approach to detect phishing websites. Shark-Eyes leverages domain features and HTML tag features to effectively differentiate phishing from legitimate sites. The framework addresses the limitations of relying solely on singular attributes such as URLs or source code by integrating DL techniques to auto-extract features across multiple modalities. It uses distinct branches to extract grayscale image attributes of domains, one-dimensional tensors representing domain structure, and tag-based DOM tree features. The CNN, GRU, and BiLSTM models are utilized to effectively extract local and sequential attributes, while the Attention mechanism emphasizes significant features. A multimodal learning approach enables Shark-Eyes to process a comprehensive range of information, enhancing phishing detection accuracy. Shark-Eyes achieved an impressive performance with an accuracy of 95.35%, a precision of 94.39%, a recall of 94.26%, and an F1-score of 94.34% on a dataset from Phishtank [26], OpenPhish [71], and Alexa [27].

V. CRITICAL REVIEW

In this section, a critical evaluation of each phishing website detection category of the proposed taxonomy is conducted, in particular by discussing the strengths and limitations of each method as demonstrated in Table 11.

A. SIGNATURE-BASED

List-based phishing detection methods offer a straightforward and efficient solution for identifying known phishing websites. These methods rely on pre-existing lists of flagged URLs, which makes them quick to implement and easy to maintain. Due to their simplicity, they are highly efficient in detecting and blocking phishing attempts from previously identified malicious sites, providing a solid defense against recurring phishing campaigns. Moreover, their low computational requirements make them suitable for real-time filtering in environments with limited resources. However, these methods are ineffective against new or evolving phishing sites, as they rely on static, pre-compiled lists that require frequent updates. Although the constant updates may be a partial solution, they pose a challenge for staying current with the ever-evolving tactics of phishers, potentially leaving gaps in protection. Also, it requires effort for humans each time and wastes storage allocation on the server side. Furthermore, list-based methods cannot detect novel phishing techniques or sites that slightly alter their URLs to avoid detection by the reference database.

Certificate-based methods leverage SSL/TLS certificates to assess the legitimacy of websites, providing a reliable mechanism for identifying suspicious sites with invalid or improperly configured certificates. These methods are particularly effective in filtering out websites that fail to meet certain certificate standards, such as expired or self-signed certificates, which are often indicators of malicious intent. Furthermore, these methods generally have low false-positive rates. However, it can be difficult to detect phishing attempts by only relying on certificate analysis, as there are no significant, general differences between the certificates of phishing and benign websites, making it difficult to distinguish between legitimate and phishing websites as discussed in [38]. For instance, phishers can easily obtain valid certificates, as CAs do not verify the legitimacy of a website's content. This means that a phishing website with a valid SSL/TLS certificate can bypass certificate-based detection, which only verifies the website's identity. Finally, it is worth noting that these methods are limited to HTTPS-based sites, leaving HTTP phishing websites undetected, which can be a minor limitation.

Phishing kit-based methods offer a unique approach to identifying phishing attacks by targeting the infrastructure behind phishing campaigns rather than just the phishing sites themselves. These methods leverage the presence of phishing kits and pre-packaged files that cybercriminals use to create and deploy phishing sites as a marker for malicious intent, which enables identifying large-scale phishing operations by tracing these kits across domains. This approach is particularly effective for the detection of multiple phishing sites that rely on the same infrastructure. Although these methods can detect sophisticated script-level techniques to evade detection and credentials exfiltration, They may miss tailored or unique phishing attacks that do not rely on pre-built kits, especially custom-developed websites. This limitation makes them less effective against targeted or high-profile attacks designed to evade conventional detection methods. They are generally less effective in identifying self-developed websites. In addition, detecting phishing kits requires in-depth knowledge of phishing infrastructure and the signatures or patterns associated with these kits, which can be resource-intensive to maintain and update. Another challenge lies in the fact that phishing-kit-based detection largely relies on observing specific files or configurations; if phishers make small alterations to these kits, they may evade detection.

To summarize, signature-based phishing detection methods offer distinct advantages in identifying certain types of phishing threats, particularly known or recurring ones. List-based methods are efficient and easy to implement but lack adaptability to new phishing sites. Certificate-based methods provide reliability for HTTPS websites but can be bypassed by attackers using legitimate certificates. Phishing-kit detection methods add another layer by targeting the infrastructure behind phishing campaigns, proving effective against large-scale or repeat attacks. However, each of these approaches has inherent limitations, as discussed previously, as they work as part of a broader, multi-layered phishing defense strategy rather than as a standalone solution.

TABLE 10. **Hybrid methods main characteristics.**

Reference	Used Technique	Detection Mechanism	Datasets	Model	Accuracy
Zhang et al. (2021) [131]	Multimodal Feature Fusion	Combines domain, favicon, and URL features using VAE and neural networks for optimized classification	MultiPhish2020	VGG-19 + VAE	97.79%
Le-Nguyen et al. (2021) [133]	Visual and Semantic Similarity Analysis	Uses fuzzy algorithms for domain generation, GloVe embeddings for semantic analysis, and SSIM for visual comparison	PhishTank + Manual collection on legitimate websites	GloVe + RF (Best accuracy)	99.82%
Chai et al. (2022) [134]	Multi-Modal Hierarchical Attention	Employs character and word-level LSTM for text, ResNet-50 for images, and hierarchical attention for modality fusion	DMOZ, PhishTank	ResNet-50 + LSTM, The classifier is Dense Layers	97.26%
Sun et al. (2023) [136]	Multimodal Feature Fusion	Combines BiLSTM for encoded URLs with Word2Vec	URLNet + A phishing dataset on GitHub + URLNet	BiLSTM + RNN + ResNet34	95.8% (Compared with HybridDLM)
Minh et al. (2023) [140]	Domain and HTML Feature Integration	Uses DL models for DOM structure analysis and domain features, incorporating attention mechanism	Alexa, OpenPhish, and PhishTank	CNN, GRU, BiLSTM, The classifier is a Hidden Layer	95.35%

TABLE 11. **Phishing detection methods main strengths and limitations.**

Method		Strengths	Limitations
Signature-Based	List-based	Efficient for known phishing websites Easy to implement	Ineffective against new, unknown phishing sites Continuous updates are required to maintain accuracy
	Certificate-based	Reliable for detecting websites with suspicious or invalid certificates Low false-positive rates for legitimate sites	Phishers can use valid certificates Do not account for the full range of phishing techniques
	Phishing-Kit	Can detect phishing at the infrastructure level Effective against large-scale phishing compaigns	Can miss unique, tailored phishing attacks Requires knowledge of phishing kits and infrastructure Not effective against self-developed sites
ML-Enabled Heuristic		Capable of detecting new and evolving phishing techniques Does not requires extensive labeled datasets	High false positives rates May require tuning for different types of attacks Subject to adversarial attacks
Deep Learning	Text-based	Can analyze text content for phishing indicators Effective against phishing sites that mimic legitimate websites through content Effective for source code analysis	Require large dataset for training Resource-intensive compared to simpler methods Can miss visual cues Subject to adversarial attacks
	Image-based	Capable of detecting phishing through visual similarity Effective against phishing sites that mimic the appearance of legitimate sites	Computationally expensive and resource intensive Performance can degrade with small visual changes by attackers Hard to keep track on the appearance of the legitimate site Subject to adversarial attacks
	Graph-based	Effective in capturing relationships between elements such as hyperlinks and structures Good for detecting hidden or indirect phishing strategies	Complex to implement Can be slow and resource-intensive Subject to adversarial attacks
	LLM-based	Strong at detecting phishing through content analysis and understanding of context Can generalize well across different types of phishing techniques	High computational requirements Prone to overfitting if not trained with diverse data Need for critical prompt engineering strategy Subject to adversarial attacks
	Hyrbid	Combines the advantages of multiple methods (e.g, text-based + image-based)	Increased complexity in implementation Subject to adversarial attacks

B. ML-ENABLED HEURISTIC

Heuristic phishing detection methods provide a flexible and adaptive approach, allowing systems to identify phishing attempts based on patterns and characteristics rather than relying on predefined signatures. A key strength of heuristic methods is their ability to detect new and evolving phishing techniques by analyzing various characteristics such as URL structures, HTML elements, and unusual webpage features. Common ML algorithms such as DT, RF, SVM, and many others are used to detect phishing attempts by leveraging the extracted features that follow a certain rule, making them less prone to static patterns of attack. In addition, these methods do not require extensive labeled datasets for training, making them a more accessible option in environments with limited data resources. However, heuristic methods come with certain limitations. Their pattern-based nature often leads to higher false-positive rates, as legitimate sites that exhibit unconventional characteristics may be incorrectly flagged as phishing, which can undermine user trust in the detection method. Also, these methods require careful tuning to balance sensitivity and specificity, as different types of phishing attacks might necessitate adjustments in detection thresholds, as many mechanisms are based on similarity matching. Moreover, Implementing ML models within heuristic systems introduces further complexity, as these models may need frequent retraining to stay current with new phishing strategies. The latter can increase computational demands and require ongoing maintenance.

Despite these limitations, heuristic methods remain valuable when integrated with other detection techniques, as they can help bridge the gap between signature-based methods, which rely on pre-learned signatures and are not proactive in detecting new phishing websites.

C. DEEP LEARNING

Text-based methods are effective in analyzing the textual content, URLs, and code elements of web pages. These methods excel at detecting phishing attempts that rely on social engineering, as they can identify keywords, phrases, and language cues commonly used to deceive users, such as urgency, threats, or requests for sensitive information. Also, they can recognize phishing attempts that mimic legitimate content by examining content such as login prompts, form instructions, and warning messages. Another strength of text-based methods is their applicability to source code analysis, where they can identify suspicious inline scripts, hidden input fields, or modified HTML tags that indicate an attempt to spoof a legitimate website. To do so, DL models such as RNNs are trained to recognize text patterns, either from the URL, content, or source code of the website. These models leverage contextual, lexical cues, and sequential patterns, as they are supported by NLP techniques, which help improve phishing detection attempts. Despite these strengths, they require extensive labeled datasets for training to accurately distinguish phishing language from legitimate content, which can be time-consuming and resource-intensive to compile. In addition, the models used in text-based detection are often computationally demanding, which may limit their practicality for real-time detection in resource-constrained environments. Moreover, text-based methods are less effective at identifying phishing attempts that rely on visual cues or structural elements. For instance, attackers can bypass text-based detection by using images or other media to convey phishing messages. Text-based methods can also be hindered by language variations, as different languages may require separate training data and specialized models. Therefore, these methods are highly valuable for detecting social engineering attempts and finding text similarities through text vectorization (embeddings) to a certain extent.

Image-based methods are particularly effective at identifying phishing attempts that rely on visual mimicry, such as copying the appearance of a legitimate website to deceive users. The methods rely on analyzing the visual components of a webpage, such as logos, color schemes, layouts, and images. This approach is especially useful for detecting phishing attacks that exploit brand identity, as it allows the detection system to flag suspicious sites that are visually similar to high-profile brands. Another strength of image-based methods is their resilience to text manipulation; even if phishers alter text elements or use different terminology, the visual similarity can still trigger detection, although some works use screenshots to extract the textual elements from the website. In addition, these methods employ DL specific for image data, such as CNNs, to analyze and classify images, making them very efficient in case only images are available. However, image-based methods suffer from several limitations. First, they are computationally expensive, often requiring significant processing power and memory. Furthermore, the effectiveness of image-based methods can diminish with small alterations made by the attacker or the legitimate website itself, leading to high false positive and negative rates, respectively. For example, minor changes, such as altering logo dimensions, adjusting color schemes, or introducing slight variations in layout, can reduce the model's accuracy. Also, maintaining up-to-date reference databases of legitimate site appearances also poses a challenge, as legitimate websites frequently update their design. Besides, they face challenges when applied to multilingual and culturally diverse settings, as logos, banners, and visual styles may vary significantly across regional versions of a brand's website, requiring additional training data to accurately detect phishing across different locales. Finally, because image-based detection primarily focuses on visual similarity, it may miss phishing sites that do not attempt to replicate a brand's appearance but rely instead on other deceptive tactics, such as URL obfuscation or content manipulation.

Graph-based focuses on the structural relationships within a webpage, such as hyperlinks, domain connections, and the network of resources linked to and from the site. By representing a webpage's components as nodes and the connections between them as edges, graph-based methods can detect patterns that are often indicative of phishing. One key strength of this approach is its ability to capture the structural dependencies and interconnectivity between various webpage elements, enabling the detection of sophisticated phishing strategies that may not rely on obvious visual or textual cues. For example, a phishing webpage may have an abnormal link structure or may link back to suspicious domains, which can be flagged using graph analysis. In addition, it can analyze the entire network of connections rather than isolated elements to detect hidden or indirect phishing attempts. To deal with the graph structure data, GNNs have proven their powerful analysis and processing capabilities in the anti-phishing domain due to their ability to efficiently model relationships between entities such as domains, URLs, IP addresses, hyperlinks…etc. Unlike conventional techniques that depend on surface-level features, a considerable number of the proposed graph-based techniques relied on extracting a deeper, more extensive, and immutable set of features derived from the network traffic or the HTML and the URL of the web page that require more time and effort to be manipulated by phishers. Despite these strengths, training and constructing the graph elements require significant processing power and memory, particularly when dealing with large-scale graphs. For instance, constructing accurate graphs for each webpage requires collecting extensive data about hyperlinks and domain connections, which may not always be feasible or fast enough for high-volume phishing detection. Also, as phishing attacks evolve quickly, keeping the graph updated with real-time data can be a constraint. The dynamic nature of phishing infrastructure requires frequent updates to the graph, which may introduce delays or inconsistencies in real-time detection systems. Finally, graph-based methods are also complex to implement and require specialized knowledge of graph theory and DL techniques, which can increase development and maintenance costs.

LLM-based methods leverage the advanced language processing capabilities of LLMs, such as GPT or Gemini, to identify phishing attempts based on contextual understanding and nuanced language cues. A primary strength of LLM-based approaches is their ability to interpret the content of phishing messages with high accuracy, as these models are trained on extensive datasets that allow them to recognize subtle linguistic patterns indicative of phishing. For example, LLMs can detect the tone, urgency, or manipulative language typical of phishing attempts, such as warnings or requests for sensitive information. Furthermore, these models excel at identifying social engineering tactics embedded within the text, making them particularly effective for detecting phishing attacks that rely on persuasive language rather than overtly suspicious elements. In addition, LLM-based methods are further enhanced by their capacity to work across different languages, which is critical for detecting phishing globally. One of the most powerful strengths of LLM-based methods is their multimodal capability, which enables them to handle diverse input types beyond text, including images, source code, and even webpage structures, without the need for any training. Unfortunately, one major challenge is the high computational requirement associated with LLMs, which makes real-time detection difficult and expensive. LLMs also face challenges with multilingual phishing detection, as they may require dedicated training datasets for each language to accurately capture phishing nuances across cultures and languages. Moreover, LLMs are highly sensitive to prompt engineering. For instance, designing effective prompts for phishing detection requires expertise and careful testing to ensure that the model consistently interprets text accurately, as proven by [129].

Hybrid methods combine multiple detection approaches, such as text-based, image-based, URL-based, and graph-based techniques, to leverage the strengths of each and achieve a more comprehensive defense against phishing attacks. One significant advantage of hybrid methods is their ability to provide a multi-layered analysis. For example, while a standalone text-based method might miss phishing attempts that rely on visual cues, combining it with image-based detection can catch those visually-driven attacks. Although hybrid methods can outperform standalone solutions, the complexity of the system increases as well. For example, Real-time detection can be particularly challenging, as hybrid systems may need to process text, images, URLs, and structural data concurrently, leading to longer response times.

Overall, DL offers a powerful and adaptive solution for phishing detection. These models excel in recognizing complex patterns and nuanced features, however DL methods can be vulnerable to overfitting, especially when trained on data that may not fully represent diverse phishing strategies. Besides, along with heuristic methods, these methods are vulnerable to adversarial attacks, where small, carefully crafted perturbations are applied to input data to mislead the model into making incorrect predictions [141], [142].

D. LESSON LEARNT

One limitation of the reviewed phishing detection methods lies in their dependence on static or rigid features that can be easily bypassed by sophisticated and rapidly evolving phishing techniques. Signature-based methods, such as list-based and certificate-based approaches, are often too reliant on pre-compiled lists or certificate properties, making them susceptible to novel phishing attacks that can evade detection by leveraging new URLs or legitimate SSL/TLS certificates. Phishing-kit-based methods are similarly limited, as they fail to detect custom-developed phishing sites or slightly modified phishing kits, while requiring extensive domain knowledge for effective implementation. Heuristic and ML-based methods, though adaptable, are prone to high false-positive rates and require careful tuning to balance between sensitivity and specificity, which may not generalize well to all phishing scenarios. DL-based approaches, including text, image, and graph-based methods, offer greater adaptability but often suffer from computational demands and limited real-time applicability, as well as challenges in detecting phishing techniques that blend visual and linguistic deception. LLM-based methods offer powerful language processing capabilities but encounter limitations in prompt engineering and computational expense. Additionally, hybrid methods, while robust, increase system complexity and response times, complicating real-time application.

To address these limitations, a multi-faceted, dynamic phishing detection framework could be developed, incorporating real-time adaptive learning and lightweight anomaly detection. First, an adaptive and self-updating feature extraction layer leveraging federated learning could enable constant updates without manual intervention, allowing models to stay current with emerging phishing techniques. For signature-based methods, incorporating a dynamic URL and certificate scoring system could enable partial or probabilistic matches to detect slightly altered phishing sites. Additionally, phishing kit detection methods could benefit from building a self-update framework that continuously expands their knowledge base, capturing previously unknown phishing kits and adapting to new kit variations automatically.

Integrating ML-based anomaly detection for unexpected traffic or network patterns could further enhance heuristic approaches while reducing false positives by applying fine-tuned thresholds based on historical data. For DL and LLM-based approaches, optimizing models for edge deployment could reduce computational strain, allowing image and text analysis to run efficiently in real-time. Finally, incorporating modular and interoperable hybrid architectures could allow different detection components (e.g., text, image, and network patterns) to work concurrently, enabling faster and more comprehensive phishing detection without significantly sacrificing speed or accuracy.

VI. DISCUSSION AND FUTURE DIRECTIONS

In this section, open issues and practical considerations in the current phishing website detection systems are discussed to uncover the emerging research aspects of phishing website detection. In addition, future directions are discussed to shed light on the current issues.

A. RESILIENCE AGAINST CLOAKING ATTACKS

In recent years, phishers have employed increasingly sophisticated techniques, known as cloaking, to evade phishing website detection systems [143]. Several studies have highlighted that current anti-phishing systems are vulnerable to both *server-side and client-side cloaking attacks*. In server-side cloaking, phishers use distinct properties profiled from phishing website detection systems, such as IP addresses, referrers, and hostnames, to filter out unwanted detection requests [144], [145], [146]. By exploiting this information, phishers can prevent detection systems from accessing the phishing website. In client-side cloaking, phishers aim to verify whether real human users are interacting with the website. They typically implement pop-up alerts or CAPTCHA challenges, which phishing detection systems may fail to engage with, allowing the phishing site to remain undetected [147], [148], [149]. Additionally, more recent work also proposed uncovered cloaking techniques that exploit behavioral disparities between legitimate users and anti-phishing entities [150].

Despite more advanced and robust models used, most current phishing website detection mechanisms are not resilient to these cloaking attacks, making them less effective at detecting the evolving threats posed by phishing websites.

B. DEPLOYMENT METHODS

Phishing website detection systems can be deployed using different approaches depending on the system architecture and user requirements. These methods can be typically categorized into **server-side** and **client-side** deployments. Server-side deployment involves hosting the detection tool on a centralized server that processes user requests, while client-side deployment integrates the tool directly within the user's environment, such as a browser extension.

In Server-side deployment, generally, the detection tool operates as a *cloud-based service* or an *API*. Users can access this service through various applications, such as browsers or dedicated software, which send information about the website (such as the URL, webpage content, or metadata) to the server for analysis. In general, servers posses enough computation power to run complex DL models, and receive multiple requests at the same time, this method can leverage more computational resources and storage capabilities, allowing for the deployment of complex and resource-intensive models, such as GNNs or LLMs. The centralized approach also makes it easier to update the models regularly with the latest data, ensuring the system stays up-to-date. In addition, server-side deployment allows multiple users to access the service simultaneously, thus making it a scalable method. It can also be considered as a way for data aggregation, where information from various users and websites can be collected and used to continuously train and improve the models. However, this may raise serious privacy concerns, especially if sensitive information is transmitted. Moreover, server-side methods depend on network connectivity and may introduce latency since users must wait for the server to process the request and return a result.

On the other hand, client-side deployment involves integrating the phishing detection tool directly within the user's environment, such as through a browser extension, desktop application, or mobile app. The detection tool operates locally on the user's device, analyzing websites, URLs, and other data in real-time without the need to communicate with an external server. This method offers a significant advantage in terms of privacy, as sensitive data remains on the user's device, reducing the risk associated with transmitting information over the network. Client-side tools generally use lightweight models optimized for quick performance, such as DT, LR models, or pre-trained neural networks that can run efficiently on user devices. For example, a browser extension may automatically scan URLs as users navigate the web, checking for known phishing patterns, malicious scripts, or suspicious content. This method can provide immediate and instant feedback and protection. One of the main advantages of this method is its low latency, as the detection occurs directly on the device, eliminating the need for network communication and server processing time. This makes it particularly effective for real-time protection while ensuring the user's data privacy. Although the client-side deployment is more practical, due to limited resources on user devices, complex and resource-intensive models will introduce certain bottlenecks. Besides, keeping the detection tool updated with the latest phishing patterns or model improvements requires frequent software updates, which can be less efficient compared to centralized server-side solutions. Finally, these tools must be carefully designed to minimize their impact on system performance and power usage.

C. DATASETS

Datasets play a crucial role in developing and evaluating phishing website detection systems. They provide the necessary data for training models and validating their effectiveness and are generally created based on the data sources, such as PhishTank [26] or Alexa [27]. However, while these datasets are valuable, they come with several limitations that can impact the accuracy and generalizability of the detection system.

These datasets or data sources can be biased in different ways. First, they are often limited to specific types of phishing content, typically those that target English-speaking users or use common platforms. This limitation means that phishing attempts in other languages or targeting localized platforms remain underrepresented [151]. As a result, models trained on such datasets may perform poorly when exposed to phishing attacks that differ linguistically or culturally, as discussed before. In addition, these datasets rely on historical records, making them outdated and unable to cover new and emerging phishing techniques as attackers continuously evolve their methods, especially with the emergence of AI. Many datasets, however, fail to update quickly enough to include these new patterns. Also, the signatures and indicators used in existing datasets may not cover all techniques. They often focus on well-known characteristics, such as specific domain patterns or visual similarities. Yet, they may overlook or inadequately represent newer indicators.

Phishing detection datasets may also fail to cover all coding techniques used in phishing websites, impacting the effectiveness of models that rely on analyzing source code. For instance, attackers frequently employ various obfuscation methods and dynamic content generation scripts to bypass detection systems. Common datasets often capture only basic or well-known coding patterns, such as static HTML structures or simple JS manipulations, while ignoring more sophisticated techniques like polymorphic scripts or JS that generate content dynamically based on user interactions. Without an inclusive representation of diverse coding techniques, models cannot generalize well, leading to a higher false-negative rate when detecting phishing websites.

Finally, Web technologies and website development practices are constantly evolving, with new frameworks, coding standards, and design approaches emerging regularly. This rapid pace of change makes it difficult for phishing detection systems to keep up with every new technology, thus the need for adaptive mechanisms or constant updates of the phishing detection mechanisms.

D. AI FOR PHISHING

Recently, attackers have increasingly leveraged AI to generate phishing websites, marking a significant evolution in phishing strategies. With minimal effort compared to previous methods, AI-based tools enable attackers to create more convincing and sophisticated phishing websites by automatically replicating the visual and structural elements of legitimate websites [152]. One of the key advantages attackers gain from AI is the ability to tailor phishing websites in real-time. By analyzing user behavior and preferences through AI algorithms, attackers can create personalized phishing pages that are more likely to deceive users. Moreover, AI facilitates the development of dynamic phishing sites that adapt their content and appearance depending on the victim's responses, making detection by automated systems more difficult [153]. In addition, the use of AI in phishing attacks is evolving in parallel with advancements in LLMs, which makes phishing campaigns more effective and scalable. LLMs are capable of generating realistic and convincing text content and professional code for the phishing webpage [154].

Commercial LLMs, such as OpenAI, are governed by strict policies designed to prevent malicious activities, including the generation of phishing websites. These policies often use safety filters and reinforcement learning from human feedback (RLHF) to align the models with ethical standards, ensuring they refuse inappropriate requests [155]. However, attackers have found ways to bypass these restrictions through prompt engineering. Studies have shown that adversarial prompts can be crafted to manipulate the model's responses. Techniques such as *"jailbreaking"* involve using cleverly designed prompts that alter the model's perception of the input context [156]. Alternatively, attackers can use pre-trained open-source models, which they fine-tune locally. By doing so, they can bypass external controls and modify the models to behave in ways that support phishing activities without restriction [157].

E. FUTURE DIRECTIONS

To address the aforementioned open issues, some possible future directions are discussed.

Phishing techniques continuously evolve, with attackers frequently updating tactics to bypass detection systems. To stay ahead of these threats, future phishing detection models must become more adaptive and self-learning, capable of detecting novel phishing attempts without the need for frequent manual updates or retraining. This can be achieved through several advanced learning techniques, including *reinforcement learning* (RL) [158], *unsupervised learning* [159], and *online learning* [160]. In the context of phishing detection, RL can be used so that the system can be designed to continuously interact with new web pages and receive feedback based on whether its detection decisions were correct. Over time, the RL model adjusts its decision-making process to minimize errors and improve accuracy. Also, unsupervised learning techniques, such as clustering and anomaly detection, could be used to detect new unseen phishing attempts. This type of learning can explore large volumes of unlabeled data and detect irregularities in the data. Moreover, online learning consists of a continuous update of data as it becomes available. Each new data instance is immediately incorporated into the model, allowing it to adapt in real time. This ensures the system to stay up-to-date with the latest phishing threats.

To design a robust and accurate phishing detection system, integrating multi-modal capability in the detection tool is important. Using a standalone approach (e.g., image-data only) can lead to high accuracy, but it may suffer from hidden irregularities. Multi-modal models [124] are designed to analyze these diverse data types collectively, allowing the system to identify phishing attempts with higher precision by referencing multiple indicators. For instance, models like CLIP (Contrastive Language-Image Pretraining) [161] or other transformer-based architectures are capable of understanding both textual and visual inputs. To further enhance multi-modal capabilities, building a comprehensive, diverse database of phishing attempts across different modalities is important. This database would include labeled examples of phishing emails, website screenshots, URLs, and network metadata.

In phishing detection, privacy is a critical concern, as users' browsing histories and interactions can contain sensitive information. Federated Learning (FL) mitigates this issue by ensuring that data never leaves the client device, it consist of a recent learning techniques that brings the model to data, instead of sending the data to a centralized server [162]. Each FL client takes charge of training a local model with his private local data and only shares the model updates with a central unit. One of the key challenges in deploying phishing detection systems using FL is dealing with the heterogeneity of clients. FL Clients may vary widely in terms of computation power, network connectivity, and data distribution. For example, some clients may have more up-to-date and diverse phishing data, while others may have limited or outdated samples. To address these differences, the federated learning system can employ techniques such as Personalized FL, which consists of adapting the global model to each client's specific data distribution [163]. Compression techniques that reduce the size of model updates, maintaining network efficiency without compromising performance. In addition, to ensure FL-based phishing detection systems maintain high performance, several advanced strategies can be employed, such as asynchronous updates by allowing clients to send updates asynchronously to ensure that the global model continues to evolve, even if some clients have slower connections or limited computing resources.

VII. CONCLUSION

In conclusion, this paper presents a comprehensive review of state-of-the-art phishing website detection techniques, emphasizing emerging approaches such as graph-based analysis, LLM-based detection, and phishing kit-based approaches—areas often overlooked in previous surveys. By introducing a new taxonomy and critically comparing these methods, this research highlights their strengths, limitations, and practical applications. The paper not only advances the theoretical understanding of phishing detection but also addresses the real-world challenges of current phishing detection systems, offering valuable insights for both researchers and practitioners. This work paves the way for more adaptive and effective phishing detection systems in an ever-evolving cyber threat landscape.

REFERENCES

[1] Z. Alkhalil, C. Hewage, L. Nawaf, and I. Khan, "Phishing attacks: A recent comprehensive study and a new anatomy," *Frontiers Comput. Sci.*, vol. 3, Mar. 2021, Art. no. 563060.

[2] L. James, "Banking on phishing," in *Phishing Exposed*, L. James, Ed., Burlington, NJ, USA: Syngress, 2006, ch. 1, pp. 1–35.

[3] M. Khonji, Y. Iraqi, and A. Jones, "Phishing detection: A literature survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 4, pp. 2091–2121, 4th Quart., 2013.

[4] R. Abdillah, Z. Shukur, M. Mohd, and T. M. Z. Murah, "Phishing classification techniques: A systematic literature review," *IEEE Access*, vol. 10, pp. 41574–41591, 2022.

[5] E. O. Yeboah-Boateng and P. M. Amanor, "Phishing, smishing & vishing: An assessment of threats against mobile devices," *J. Emerg. Trends Comput. Inf. Sci.*, vol. 5, no. 4, pp. 297–307, 2014.

[6] K. Thomas, F. Li, A. Zand, J. Barrett, J. Ranieri, L. Invernizzi, Y. Markov, O. Comanescu, V. Eranti, A. Moscicki, D. Margolis, V. Paxson, and E. Bursztein, "Data breaches, phishing, or malware? Understanding the risks of stolen credentials," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2017, pp. 1421–1434.

[7] K. L. Chiew, K. S. C. Yong, and C. L. Tan, "A survey of phishing attacks: Their types, vectors and technical approaches," *Expert Syst. Appl.*, vol. 106, pp. 1–20, Sep. 2018.

[8] *Anti-Phishing Working Group (APWG)*. [Online]. Available: https://apwg.org/

[9] R. Zieni, L. Massari, and M. C. Calzarossa, "Phishing or not phishing? A survey on the detection of phishing websites," *IEEE Access*, vol. 11, pp. 18499–18519, 2023.

[10] O. Sarker, A. Jayatilaka, S. Haggag, C. Liu, and M. A. Babar, "A multi-vocal literature review on challenges and critical success factors of phishing education, training and awareness," *J. Syst. Softw.*, vol. 208, Feb. 2024, Art. no. 111899.

[11] S. Gupta, A. Singhal, and A. Kapoor, "A literature survey on social engineering attacks: Phishing attack," in *Proc. Int. Conf. Comput., Commun. Autom. (ICCCA)*, Apr. 2016, pp. 537–540.

[12] L. Tang and Q. H. Mahmoud, "A survey of machine learning-based solutions for phishing website detection," *Mach. Learn. Knowl. Extraction*, vol. 3, no. 3, pp. 672–694, Aug. 2021.

[13] C. L. Tan, K. L. Chiew, K. S. C. Yong, S. N. Sze, J. Abdullah, and Y. Sebastian, "A graph-theoretic approach for the detection of phishing webpages," *Comput. Secur.*, vol. 95, Aug. 2020, Art. no. 101793.

[14] J. M. Lindamulage, L. MandiraPabasari, S. P. J. Yapa, I. S. S. Perera, and J. Krishara, "Vision GNN based phishing website detection," in *Proc. Int. Conf. Innov. Comput., Intell. Commun. Smart Electr. Syst. (ICSES)*, Dec. 2023, pp. 1–7.

[15] M.-Y. Su and K.-L. Su, "BERT-based approaches to identifying malicious URLs," *Sensors*, vol. 23, no. 20, p. 8499, Oct. 2023.

[16] Y. Li, C. Huang, S. Deng, M. L. Lock, T. Cao, N. Oo, H. W. Lim, and B. Hooi, "KnowPhish: Large language models meet multimodal knowledge graphs for enhancing reference-based phishing detection," in *Proc. 33rd USENIX Secur. Symp. (USENIX Security)*, Philadelphia, PA, USA. Berkeley, CA, USA: USENIX Association, Aug. 2024, pp. 793–810.

[17] B. Kondracki, B. A. Azad, O. Starov, and N. Nikiforakis, "Catching transparent phish: Analyzing and detecting MITM phishing toolkits," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.* New York, NY, USA: Association for Computing Machinery, Nov. 2021, pp. 36–50.

[18] H. L. J. Bijmans, T. Booij, A. Schwedersky, A. Nedgabat, and R. S. V. Wegberg, "Catching phishers by their bait: Investigating the Dutch phishing landscape through phishing kit detection," in *Proc. 30th USENIX Secur. Symp. (USENIX Security)*. Berkeley, CA, USA: USENIX Association, Aug. 2021, pp. 3757–3774.

[19] Z. Dou, I. Khalil, A. Khreishah, A. Al-Fuqaha, and M. Guizani, "Systematization of knowledge (SoK): A systematic review of software-based web phishing detection," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2797–2819, 4th Quart., 2017.

[20] C. Singh and S. Meenu, "Phishing website detection based on machine learning: A survey," in *Proc. 6th Int. Conf. Adv. Comput. Commun. Syst. (ICACCS)*, Mar. 2020, pp. 398–404. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9074400

[21] R. Zaimi, M. Hafidi, and M. Lamia, ''Survey paper: Taxonomy of website anti-phishing solutions,'' in *Proc. 7th Int. Conf. Social Netw. Anal., Manage. Secur. (SNAMS)*, Dec. 2020, pp. 1–8.

[22] M. H. Alkawaz, S. J. Steven, A. I. Hajamydeen, and R. Ramli, ''A comprehensive survey on identification and analysis of phishing website based on machine learning methods,'' in *Proc. IEEE 11th IEEE Symp. Comput. Appl. Ind. Electron. (ISCAIE)*, Apr. 2021, pp. 82–87.

[23] C. Rajeswary and M. Thirumaran, ''A comprehensive survey of automated website phishing detection techniques: A perspective of artificial intelligence and human behaviors,'' in *Proc. Int. Conf. Sustain. Comput. Data Commun. Syst. (ICSCDS)*, Mar. 2023, pp. 420–427.

[24] S. S. Kumar, B. Swarnagowri, R. Susmitha, and R. Sujitha, ''Exploring techniques for web phishing detection: A comprehensive survey,'' in *Proc. 7th Int. Conf. Intell. Comput. Control Syst. (ICICCS)*, May 2023, pp. 1415–1420.

[25] A. Safi and S. Singh, ''A systematic literature review on phishing website detection techniques,'' *J. King Saud Univ., Comput. Inf. Sci.*, vol. 35, no. 2, pp. 590–611, Feb. 2023.

[26] CTI Group. (2006). *PhishTank*. [Online]. Available: http://www.phishtank.com/

[27] Amazon. *Alexa Top Sites*. Accessed: Jan. 22, 2021. [Online]. Available: http://www.alexa.com/topsites

[28] S. Asiri, Y. Xiao, S. Alzahrani, S. Li, and T. Li, ''A survey of intelligent detection designs of HTML URL phishing attacks,'' *IEEE Access*, vol. 11, pp. 6421–6443, 2023.

[29] Y. Justindhas, V. Raghul, S. Pramadeish, and S. Prakash, ''A comprehensive review on an ensemble-based machine learning approach for phishing website detection,'' in *Proc. 2nd Int. Conf. Comput., Commun. Control (IC4)*, Feb. 2024, pp. 1–6.

[30] A. Kulkarni, V. Balachandran, and T. Das, ''Phishing webpage detection: Unveiling the threat landscape and investigating detection techniques,'' *IEEE Commun. Surveys Tuts.*, early access, Aug. 12, 2024, doi: 10.1109/COMST.2024.3441752.

[31] H. Shirazi, B. Bezawada, and I. Ray, '''Know thy domain name': Unbiased phishing detection using domain name based features,'' in *Proc. 23nd ACM Symp. Access Control Models Technol.*, Jun. 2018, pp. 69–75.

[32] J. Spaulding, S. Upadhyaya, and A. Mohaisen, ''The landscape of domain name typosquatting: Techniques and countermeasures,'' in *Proc. 11th Int. Conf. Availability, Rel. Secur. (ARES)*, Aug. 2016, pp. 284–289.

[33] J. Kumar, A. Santhanavijayan, B. Janet, B. Rajendran, and B. S. Bindhumadhava, ''Phishing website classification and detection using machine learning,'' in *Proc. Int. Conf. Comput. Commun. Informat. (ICCCI)*, Jan. 2020, pp. 1–6.

[34] J. Reynolds, D. Kumar, Z. Ma, R. Subramanian, M. Wu, M. Shelton, J. Mason, E. Stark, and M. Bailey, ''Measuring identity confusion with uniform resource locators,'' in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, Apr. 2020, pp. 1–12.

[35] S. Jalil, M. Usman, and A. Fong, ''Highly accurate phishing URL detection based on machine learning,'' *J. Ambient Intell. Hum. Comput.*, vol. 14, no. 7, pp. 9233–9251, Jul. 2023.

[36] X. Jie, L. Haoliang, and J. Ao, ''A new model for simultaneous detection of phishing and darknet websites,'' in *Proc. 7th Int. Conf. Comput. Commun. (ICCC)*, Dec. 2021, pp. 2002–2006.

[37] K. Althobaiti, G. Rummani, and K. Vaniea, ''A review of human- and computer-facing URL phishing features,'' in *Proc. IEEE Eur. Symp. Secur. Privacy Workshops (EuroSPW)*, Jun. 2019, pp. 182–191.

[38] U. Meyer and V. Drury, ''Certified phishing: Taking a look at public key certificates of phishing websites,'' in *Proc. 15th Symp. Usable Privacy Secur. (SOUPS)*, Santa Clara, CA, USA. Berkeley, CA, USA: USENIX Association, Aug. 2019, pp. 211–223.

[39] Z. Dong, K. Kane, and L. J. Camp, ''Detection of rogue certificates from trusted certificate authorities using deep neural networks,'' *ACM Trans. Privacy Secur.*, vol. 19, no. 2, pp. 1–31, Sep. 2016.

[40] R. Verma, N. Shashidhar, and N. Hossain, ''Detecting phishing emails the natural language way,'' in *Proc. 17th Eur. Symp. Res. Comput. Secur.*, Pisa, Italy. Berlin, Germany: Springer, Sep. 2012, pp. 824–841. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-33167-1_47#citeas

[41] IBM. (2024). *What is Phishing?* [Online]. Available: https://www.ibm.com/topics/phishing

[42] J.-L. Chen, Y.-W. Ma, and K.-L. Huang, ''Intelligent visual similarity-based phishing websites detection,'' *Symmetry*, vol. 12, no. 10, p. 1681, Oct. 2020.

[43] M. Almousa, R. Furst, and M. Anwar, ''Characterizing coding style of phishing websites using machine learning techniques,'' in *Proc. 4th Int. Conf. Transdisciplinary AI (TransAI)*, Sep. 2022, pp. 101–105.

[44] E. Merlo, M. Margier, G.-V. Jourdan, and I.-V. Onut, ''Phishing kits source code similarity distribution: A case study,'' in *Proc. IEEE Int. Conf. Softw. Anal., Evol. Reeng. (SANER)*, Mar. 2022, pp. 983–994.

[45] X. Han, N. Kheir, and D. Balzarotti, ''PhishEye: Live monitoring of sandboxed phishing kits,'' in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 1402–1413.

[46] J. M. Torres, C. I. Comesa na, and P. J. García-Nieto, ''Machine learning techniques applied to cybersecurity,'' *Int. J. Mach. Learn. Cybern.*, vol. 10, no. 10, pp. 2823–2836, 2019.

[47] L. Grinsztajn, E. Oyallon, and G. Varoquaux, ''Why do tree-based models still outperform deep learning on typical tabular data?'' in *Proc. 36th Int. Conf. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates, 2024, pp. 507–520.

[48] S. S. Abdullahi, S. Yiming, S. H. Muhammad, A. Mustapha, A. M. Aminu, A. Abdullahi, M. Bello, and S. M. Aliyu, ''Deep sequence models for text classification tasks,'' in *Proc. Int. Conf. Electr., Commun., Comput. Eng. (ICECCE)*, Jun. 2021, pp. 1–6.

[49] N. Q. Do, A. Selamat, O. Krejcar, E. Herrera-Viedma, and H. Fujita, ''Deep learning for phishing detection: Taxonomy, current challenges and future directions,'' *IEEE Access*, vol. 10, pp. 36429–36463, 2022.

[50] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, ''BERT: Pre-training of deep bidirectional transformers for language understanding,'' 2018, *arXiv:1810.04805*.

[51] Y. E. Seyyar, A. G. Yavuz, and H. M. Ünver, ''An attack detection framework based on BERT and deep learning,'' *IEEE Access*, vol. 10, pp. 68633–68644, 2022.

[52] S. Sharma and K. Guleria, ''Deep learning models for image classification: Comparison and applications,'' in *Proc. 2nd Int. Conf. Adv. Comput. Innov. Technol. Eng. (ICACITE)*, Apr. 2022, pp. 1733–1738.

[53] W. Wei, Q. Ke, J. Nowak, M. Korytkowski, R. Scherer, and M. Woźniak, ''Accurate and fast URL phishing detector: A convolutional neural network approach,'' *Comput. Netw.*, vol. 178, Sep. 2020, Art. no. 107275.

[54] O. Naim, D. Cohen, and I. Ben-Gal, ''Malicious website identification using design attribute learning,'' *Int. J. Inf. Secur.*, vol. 22, no. 5, pp. 1207–1217, Oct. 2023.

[55] W. Yao, Y. Ding, and X. Li, ''Deep learning for phishing detection,'' in *Proc. IEEE Int. Conf Parallel Distrib. Process. Appl., Ubiquitous Comput. Commun., Big Data Cloud Comput., Social Comput. Netw., Sustain. Comput. Commun. (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, Dec. 2018, pp. 645–650.

[56] S. Motie and B. Raahemi, ''Financial fraud detection using graph neural networks: A systematic review,'' *Expert Syst. Appl.*, vol. 240, Apr. 2024, Art. no. 122156.

[57] X. Liu, M. Yan, L. Deng, G. Li, X. Ye, D. Fan, S. Pan, and Y. Xie, ''Survey on graph neural network acceleration: An algorithmic perspective,'' 2022, *arXiv:2202.04822*.

[58] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, ''Graph neural networks in recommender systems: A survey,'' *ACM Comput. Surv.*, vol. 55, no. 5, pp. 1–37, Dec. 2022.

[59] Z. Jin, Y. Wang, Q. Wang, Y. Ming, T. Ma, and H. Qu, ''GNNLens: A visual analytics approach for prediction error diagnosis of graph neural networks,'' *IEEE Trans. Vis. Comput. Graphics*, vol. 29, no. 6, pp. 3024–3038, Jun. 2023.

[60] G. Mani and G. B. Namomsa, ''Large language models (LLMs): Representation matters, low-resource languages and multi-modal architecture,'' in *Proc. IEEE AFRICON*, Sep. 2023, pp. 1–6.

[61] A. M. Easin, S. Sourav, and O. Tamás, ''An intelligent LLM-powered personalized assistant for digital banking using LangGraph and chain of thoughts,'' in *Proc. IEEE 22nd Jubilee Int. Symp. Intell. Syst. Informat. (SISY)*, Sep. 2024, pp. 625–630.

[62] A. Castelnovo, R. Depalmas, F. Mercorio, N. Mombelli, D. Potertì, A. Serino, A. Seveso, S. Sorrentino, and L. Viola, ''Augmenting XAI with LLMs: A case study in banking marketing recommendation,'' in *Proc. World Conf. Explainable Artif. Intell.* Cham, Switzerland: Springer, 2024, pp. 211–229. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-031-63787-2_11#citeas

[63] Y. Chen, M. Cui, D. Wang, Y. Cao, P. Yang, B. Jiang, Z. Lu, and B. Liu, ''A survey of large language models for cyber threat detection,'' *Comput. Secur.*, vol. 145, Oct. 2024, Art. no. 104016.

[64] (2022). *Google Safe Browsing*. [Online]. Available: https://safebrowsing.google.com/

[65] P. Kalaharsha and B. M. Mehtre, "Detecting phishing sites—An overview," 2021, *arXiv:2103.12739*.

[66] G. Armano et al., "Real-time client-side phishing prevention," M.S. thesis, School Sci., Aalto Univ., Espoo, Finland, 2016.

[67] D. Kim, H. Cho, Y. Kwon, A. Doupé, S. Son, G.-J. Ahn, and T. Dumitras, "Security analysis on practices of certificate authorities in the HTTPS phishing ecosystem," in *Proc. ACM Asia Conf. Comput. Commun. Secur.* New York, NY, USA: Association for Computing Machinery, May 2021, pp. 407–420.

[68] Y. Sakurai, T. Watanabe, T. Okuda, M. Akiyama, and T. Mori, "Discovering HTTPSified phishing websites using the TLS certificates footprints," in *Proc. IEEE Eur. Symp. Secur. Privacy Workshops (EuroSPW)*, Sep. 2020, pp. 522–531.

[69] J. Aas, R. Barnes, B. Case, Z. Durumeric, P. Eckersley, A. Flores-López, J. A. Halderman, J. Hoffman-Andrews, J. Kasten, E. Rescorla, S. Schoen, and B. Warren, "Let's encrypt: An automated certificate authority to encrypt the entire Web," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2019, pp. 2473–2487.

[70] *Cpanel Web Host*. Accessed: Jan. 22, 2021. [Online]. Available: https://cpanel.net

[71] OpenPhish. (2024). *OpenPhish Phishing Feeds*. [Online]. Available: https://www.openphish.com/phishing_feeds.html

[72] *Censys Search*. Accessed: Sep. 10, 2024. [Online]. Available: https://search.censys.io/

[73] K. Khan, S. U. Rehman, K. Aziz, S. Fong, and S. Sarasvady, "DBSCAN: Past, present and future," in *Proc. 5th Int. Conf. Appl. Digit. Inf. Web Technol. (ICADIWT)*, Feb. 2014, pp. 232–238.

[74] Hispasec Sistemas. *VirusTotal*. Accessed: Sep. 10, 2024. [Online]. Available: https://www.virustotal.com/gui/

[75] P. Zhang, Z. Sun, S. Kyung, H. W. Behrens, Z. L. Basque, H. Cho, A. Oest, R. Wang, T. Bao, Y. Shoshitaishvili, G.-J. Ahn, and A. Doupé, "I'm SPARTACUS, no. I'm SPARTACUS: Proactively protecting users from phishing by intentionally triggering cloaking behavior," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.* New York, NY, USA: Association for Computing Machinery, Nov. 2022, pp. 3165–3179.

[76] Phishunt. (2021). *Exposing Phishing Kits Seen From Phishunt.io*. [Online]. Available: https://github.com/danlopgom/phishing_kits

[77] W. Lee, J. Hur, and D. Kim, "Beneath the phishing scripts: A script-level analysis of phishing kits and their impact on real-world phishing websites," in *Proc. 19th ACM Asia Conf. Comput. Commun. Secur.* New York, NY, USA: Association for Computing Machinery, Jul. 2024, pp. 856–872.

[78] M. Sameen, K. Han, and S. O. Hwang, "PhishHaven—An efficient real-time AI phishing URLs detection system," *IEEE Access*, vol. 8, pp. 83425–83443, 2020.

[79] A. C. Bahnsen, I. Torroledo, L. D. Camacho, and S. Villegas, "Deepphish: Simulating malicious ai," in *Proc. APWG Symp. Electron. Crime Res. (eCrime)*, 2018, pp. 1–8.

[80] G. Sonowal and K. S. Kuppusamy, "PhiDMA—A phishing detection model with multi-filter approach," *J. King Saud Univ., Comput. Inf. Sci.*, vol. 32, no. 1, pp. 99–112, Jan. 2020.

[81] Phishload. (2016). *Legitimate URL Dataset*. [Online]. Available: http://www.medien.ifi.lmu.de/team/max.maurer/files/phishload/

[82] M. Sánchez-Paniagua, E. Fidalgo, E. Alegre, and R. Alaiz-Rodríguez, "Phishing websites detection using a novel multipurpose dataset and Web technologies features," *Expert Syst. Appl.*, vol. 207, Nov. 2022, Art. no. 118010.

[83] GVIS. *Phishing Index Login Websites Dataset (PILWD-134K)*. [Online]. Available: https://gvis.unileon.es/datasets-pilwd-134k/

[84] P. Chinnasamy, N. Kumaresan, R. Selvaraj, S. Dhanasekaran, K. Ramprathap, and S. Boddu, "An efficient phishing attack detection using machine learning algorithms," in *Proc. Int. Conf. Adv. Smart, Secure Intell. Comput. (ASSIC)*, Nov. 2022, pp. 1–6.

[85] S. Saha Roy, U. Karanjit, and S. Nilizadeh, "Phishing in the free waters: A study of phishing attacks created using free website building services," in *Proc. ACM Internet Meas. Conf.* New York, NY, USA: Association for Computing Machinery, Oct. 2023, pp. 268–281.

[86] M. Bahaghighat, M. Ghasemi, and F. Ozen, "A high-accuracy phishing website detection method based on machine learning," *J. Inf. Secur. Appl.*, vol. 77, Sep. 2023, Art. no. 103553.

[87] G. Vrbančič, I. Fister, and V. Podgorelec, "Datasets for phishing websites detection," *Data Brief*, vol. 33, Dec. 2020, Art. no. 106438.

[88] V. A. V. Adap, G. A. Castillo, E. J. M. D. Reyes, E. B. Ronquillo, and L. A. Vea, "Do not feed the phish: Phishing website detection using URL-based features," in *Proc. 5th World Symp. Softw. Eng. (WSSE)*. New York, NY, USA: Association for Computing Machinery, Sep. 2023, pp. 135–141.

[89] S. Ariyadasa, S. Fernando, and S. Fernando, "Phishing websites dataset," Mendeley Data, V1, 2021, doi: 10.17632/n96ncsr5g4.1. [Online]. Available: https://data.mendeley.com/datasets/n96ncsr5g4/1

[90] A. R. Talukder, F. Alam, S. T. Mim, and M. A. A. Emon, "Detecting phishing websites using Naïve Bayes classification," in *Proc. 3rd Int. Conf. Adv. Electr. Electron. Eng. (ICAEEE)*, Apr. 2024, pp. 1–6.

[91] T. Tiwari. *Phishing Site URLs Dataset*. Accessed: Sep. 15, 2024. [Online]. Available: https://www.kaggle.com/datasets/taruntiwarihp/phishing-site-urls

[92] S. Abdelnabi, K. Krombholz, and M. Fritz, "VisualPhishNet: Zero-day phishing website detection by visual similarity," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.* New York, NY, USA: Association for Computing Machinery, Oct. 2020, pp. 1681–1698.

[93] Similarweb LTD. *Similarweb*. Accessed: Sep. 15, 2024. [Online]. Available: https://similarweb.com/

[94] Y. Lin, R. Liu, D. M. Divakaran, J. Y. Ng, Q. Z. Chan, Y. Lu, Y. Si, F. Zhang, and J. S. Dong, "Phishpedia: A hybrid deep learning based approach to visually identify phishing webpages," in *Proc. 30th USENIX Secur. Symp. (USENIX Security)*. Berkeley, CA, USA: USENIX Association, Aug. 2021, pp. 3793–3810.

[95] N. B. Trinh, T. D. Phan, and V.-H. Pham, "Leveraging deep learning image classifiers for visual similarity-based phishing website detection," in *Proc. 11th Int. Symp. Inf. Commun. Technol.* New York, NY, USA: Association for Computing Machinery, Dec. 2022, pp. 134–141.

[96] R. Liu, Y. Lin, X. Yang, S. H. Ng, D. M. Divakaran, and J. S. Dong, "Inferring phishing intention via webpage appearance and dynamics: A deep vision based approach," in *Proc. 31st USENIX Secur. Symp. (USENIX Security)*, Boston, MA, USA. Berkeley, CA, USA: USENIX Association, Aug. 2022, pp. 1633–1650.

[97] Calidog. (2024). *Certstream—Real-Time Certificate Transparency Log Updates*. [Online]. Available: https://certstream.calidog.io/

[98] R. Liu, Y. Lin, Y. Zhang, P. H. Lee, and J. S. Dong, "Knowledge expansion and counterfactual interaction for reference-based phishing detection," in *Proc. 32nd USENIX Secur. Symp. (USENIX Security)*, Anaheim, CA, USA. Berkeley, CA, USA: USENIX Association, Aug. 2023, pp. 4139–4156.

[99] T. Niu and B. Wu, "Visual-based phishing website recognition," in *Proc. IEEE 6th Adv. Inf. Manage., Communicates, Electron. Autom. Control Conf. (IMCEC)*, vol. 6, May 2024, pp. 992–997.

[100] M. Wang, L. Song, L. Li, Y. Zhu, and J. Li, "Phishing webpage detection based on global and local visual similarity," *Expert Syst. Appl.*, vol. 252, Oct. 2024, Art. no. 124120.

[101] X. Xiao, W. Xiao, D. Zhang, B. Zhang, G. Hu, Q. Li, and S. Xia, "Phishing websites detection via CNN and multi-head self-attention on imbalanced datasets," *Comput. Secur.*, vol. 108, Sep. 2021, Art. no. 102372.

[102] (2012). *5000 BEST WEBSITES Homepage*. [Online]. Available: http://5000best.com/websites/

[103] A. Aljofey, Q. Jiang, A. Rasool, H. Chen, W. Liu, Q. Qu, and Y. Wang, "An effective detection approach for phishing websites using URL and HTML features," *Sci. Rep.*, vol. 12, no. 1, p. 8842, May 2022.

[104] A. S. Bozkir, F. C. Dalgic, and M. Aydos, "GramBeddings: A new neural network for URL based identification of phishing web pages through N-gram embeddings," *Comput. Secur.*, vol. 124, Jan. 2023, Art. no. 102964.

[105] S. Gopali, A. S. Namin, F. Abri, and K. S. Jones, "The performance of sequential deep learning models in detecting phishing websites using contextual features of URLs," in *Proc. 39th ACM/SIGAPP Symp. Appl. Comput.* New York, NY, USA: Association for Computing Machinery, Apr. 2024, pp. 1064–1066.

[106] Ebubekirbbr. (2019). *A Phishing Dataset*. [Online]. Available: https://github.com/ebubekirbbr/pdd

[107] N. Q. Do, A. Selamat, H. Fujita, and O. Krejcar, "An integrated model based on deep learning classifiers and pre-trained transformer for phishing URL detection," *Future Gener. Comput. Syst.*, vol. 161, pp. 269–285, Dec. 2024.

[108] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from URLs," *Expert Syst. Appl.*, vol. 117, pp. 345–357, Mar. 2019.

[109] H. Yuan, Z. Yang, X. Chen, Y. Li, and W. Liu, "URL2 Vec: URL modeling with character embeddings for fast and accurate phishing website detection," in *Proc. IEEE Int. Conf Parallel Distrib. Process. Appl., Ubiquitous Comput. Commun., Big Data Cloud Comput., Social Comput. Netw., Sustain. Comput. Commun. (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, Dec. 2018, pp. 265–272.

[110] F. Çolhak, M. İ. Ecevit, B. E. Uçar, R. Creutzburg, and H. Dağ, "Phishing website detection through multi-model analysis of HTML content," 2024, *arXiv:2401.04820*.

[111] F. Çolhak. *MTLP Dataset*. Accessed: Sep. 20, 2024. [Online]. Available: https://drive.google.com/file/d/1Lp3ueOd7AxmAl2Y0jJ2U2XlEFa6q8AcT/view

[112] Z. Futai, G. Yuxiang, P. Bei, P. Li, and L. Linsen, "Web phishing detection based on graph mining," in *Proc. 2nd IEEE Int. Conf. Comput. Commun. (ICCC)*, Oct. 2016, pp. 1061–1066.

[113] Common Crawl. *Open Repository of Web Crawl Data*. Accessed: Sep. 20, 2024. [Online]. Available: http://commoncrawl.org

[114] K. L. Chiew, C. L. Tan, K. Wong, K. S. C. Yong, and W. K. Tiong, "A new hybrid ensemble feature selection framework for machine learning-based phishing detection system," *Inf. Sci.*, vol. 484, pp. 153–166, May 2019.

[115] L. Ouyang and Y. Zhang, "Phishing web page detection with HTML-level graph neural network," in *Proc. IEEE 20th Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*, Oct. 2021, pp. 952–958.

[116] V. Le Pochat, T. Van Goethem, S. Tajalizadehkhoob, M. Korczynski, and W. Joosen, "Tranco: A research-oriented top sites ranking hardened against manipulation," in *Proc. Netw. Distrib. Syst. Secur. Symp.* Reston, VA, USA: Internet Society, 2019. [Online]. Available: https://www.ndss-symposium.org/ndss-paper/tranco-a-research-oriented-top-sites-ranking-hardened-against-manipulation/

[117] F. Tchakounte, J. C. Teukeng Ngnintedem, I. Damakoa, F. Ahmadou, and F. A. Kuate Fotso, "Crawl-shing: A focused crawler for fetching phishing contents based on graph isomorphism," *J. King Saud Univ., Comput. Inf. Sci.*, vol. 34, no. 10, pp. 8888–8898, Nov. 2022.

[118] T. Bilot, G. Geis, and B. Hammi, "PhishGNN: A phishing website detection framework using graph neural networks," in *Proc. 19th Int. Conf. Secur. Cryptogr.* Setúbal, Portugal: SciTePress, 2022, pp. 428–435.

[119] S. Ariyadasa, S. Fernando, and S. Fernando, "Combining long-term recurrent convolutional and graph convolutional networks to detect phishing sites using URL and HTML," *IEEE Access*, vol. 10, pp. 82355–82375, 2022.

[120] J. Feng, L. Zou, O. Ye, and J. Han, "Web2Vec: Phishing webpage detection method based on multidimensional features driven by deep learning," *IEEE Access*, vol. 8, pp. 221214–221224, 2020.

[121] K. Han, Y. Wang, J. Guo, Y. Tang, and E. Wu, "Vision GNN: An image is worth graph of nodes," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Dec. 2022, pp. 8291–8303.

[122] *Malicious URLs Dataset*. Accessed: Sep. 20, 2024. [Online]. Available: https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset

[123] *ISCX-URL 2016 Dataset*. Accessed: Sep. 20, 2024. [Online]. Available: https://www.unb.ca/cic/datasets/url-2016.html

[124] J. Lee, P. Lim, B. Hooi, and D. M. Divakaran, "Multimodal large language models for phishing webpage detection and identification," 2024, *arXiv:2408.05941*.

[125] A. Kulkarni, V. Balachandran, D. Mon Divakaran, and T. Das, "From ML to LLM: Evaluating the robustness of phishing webpage detection models against adversarial attacks," 2024, *arXiv:2407.20361*.

[126] S. S. Roy and S. Nilizadeh, "PhishLang: A lightweight, client-side phishing detection framework using MobileBERT for real-time, explainable threat mitigation," 2024, *arXiv:2408.05667*.

[127] T. Koide, N. Fukushi, H. Nakano, and D. Chiba, "Detecting phishing sites using ChatGPT," 2023, *arXiv:2306.05816*.

[128] H. Nakano, D. Chiba, T. Koide, N. Fukushi, T. Yagi, T. Hariu, K. Yoshioka, and T. Matsumoto, "Canary in Twitter mine: Collecting phishing reports from experts and non-experts," in *Proc. 18th Int. Conf. Availability, Rel. Secur.* New York, NY, USA: Association for Computing Machinery, Aug. 2023, pp. 1–12.

[129] M. Schesny, N. Lutz, T. Jägle, F. Gerschner, M. Klaiber, and A. Theissler, "Enhancing website fraud detection: A ChatGPT-based approach to phishing detection," in *Proc. IEEE 48th Annu. Comput., Softw., Appl. Conf. (COMPSAC)*. Los Alamitos, CA, USA: IEEE Computer Society, Jul. 2024, pp. 1494–1495.

[130] R. Liu, Y. Lin, X. Teoh, G. Liu, Z. Huang, and J. S. Dong, "Less defined knowledge and more true alarms: Reference-based phishing detection without a pre-defined reference list," in *Proc. 33rd USENIX Secur. Symp. (USENIX Security)*, 2024, pp. 523–540.

[131] L. Zhang, P. Zhang, L. Liu, and J. Tan, "Multiphish: Multi-modal features fusion networks for phishing detection," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 3520–3524.

[132] L. Zhang, P. Zhang, L. Liu, and J. Tan. (2020). *MultiPhish Phishing Dataset*. [Online]. Available: https://github.com/DataReleased/MultiPhish

[133] M.-K. Le-Nguyen, T.-C.-H. Nguyen, D.-T. Le, V.-H. Nguyen, L.-P. Tôn, and K. Nguyen-An, "Hunting phishing websites using a hybrid fuzzy-semantic-visual approach," in *Proc. 15th Int. Conf. Adv. Comput. Appl. (ACOMP)*, Nov. 2021, pp. 38–45.

[134] Y. Chai, Y. Zhou, W. Li, and Y. Jiang, "An explainable multi-modal hierarchical attention model for developing phishing threat intelligence," *IEEE Trans. Depend. Sec. Comput.*, vol. 19, no. 2, pp. 790–803, Mar. 2022.

[135] DMOZ. *Open Directory Project*. Accessed: Sep. 20, 2024. [Online]. Available: https://dmoztools.net

[136] Y. Sun, G. Liu, X. Han, W. Zuo, and W. Liu, "FusionNet: An effective network phishing website detection framework based on multi-modal fusion," in *Proc. IEEE Int. Conf. High Perform. Comput. Commun., Data Sci. Syst., Smart City Dependability Sensor, Cloud Big Data Syst. Appl. (HPCC/DSS/SmartCity/DependSys)*, Dec. 2023, pp. 474–481.

[137] S. Ariyadasa, S. Fernando, and S. Fernando, "Detecting phishing attacks using a combined model of LSTM and CNN," *Int. J. Adv. Appl. Sci.*, vol. 7, no. 7, pp. 56–67, Jul. 2020.

[138] H. Le, Q. Pham, D. Sahoo, and S. C. H. Hoi, "URLNet: Learning a URL representation with deep learning for malicious URL detection," 2018, *arXiv:1802.03162*.

[139] *Phishing Domain Database*. Accessed: Sep. 20, 2024. [Online]. Available: https://github.com/mitchellkrogza/Phishing.Database

[140] M. Vo Quang, D. B. T. Hai, N. T. K. Ngoc, S. N. D. Hoang, Q. N. Huu, D. P. The, and V.-H. Pham, "Shark-eyes: A multimodal fusion framework for multi-view-based phishing website detection," in *Proc. 12th Int. Symp. Inf. Commun. Technol.* New York, NY, USA: Association for Computing Machinery, Dec. 2023, pp. 793–800.

[141] H. Shirazi, B. Bezawada, I. Ray, and C. Anderson, "Adversarial sampling attacks against phishing detection," in *Proc. 33rd IFIP Annu. Conf. Data Appl. Secur. Privacy*, Charleston, SC, USA. Cham, Switzerland: Springer, Jul. 2019, pp. 83–101. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-22479-0_5#citeas

[142] A. AlEroud and G. Karabatis, "Bypassing detection of URL-based phishing attacks using generative adversarial deep neural networks," in *Proc. 6th Int. Workshop Secur. Privacy Anal.*, Mar. 2020, pp. 53–60.

[143] W. Li, S. Manickam, S. U. A. Laghari, and Y.-W. Chong, "Uncovering the cloak: A systematic review of techniques used to conceal phishing websites," *IEEE Access*, vol. 11, pp. 71925–71939, 2023.

[144] A. Oest, Y. Safaei, A. Doupé, G.-J. Ahn, B. Wardman, and K. Tyers, "PhishFarm: A scalable framework for measuring the effectiveness of evasion techniques against browser phishing blacklists," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 1344–1361.

[145] A. Oest, Y. Safaei, P. Zhang, B. Wardman, K. Tyers, Y. Shoshitaishvili, and A. Doupé, "PhishTime: Continuous longitudinal measurement of the effectiveness of anti-phishing blacklists," in *Proc. 29th USENIX Secur. Symp. (USENIX Security)*. Berkeley, CA, USA: USENIX Association, Aug. 2020, pp. 379–396.

[146] A. Oest, Y. Safei, A. Doupé, G.-J. Ahn, B. Wardman, and G. Warner, "Inside a phisher's mind: Understanding the anti-phishing ecosystem through phishing kit analysis," in *Proc. APWG Symp. Electron. Crime Res. (eCrime)*, May 2018, pp. 1–12.

[147] P. Zhang, A. Oest, H. Cho, Z. Sun, R. Johnson, B. Wardman, S. Sarker, A. Kapravelos, T. Bao, R. Wang, Y. Shoshitaishvili, A. Doupé, and G.-J. Ahn, "CrawlPhish: Large-scale analysis of client-side cloaking techniques in phishing," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2021, pp. 1109–1124.

[148] A. Oest, P. Zhang, B. Wardman, E. Nunes, J. Burgis, A. Zand, K. Thomas, A. Doupé, and G.-J. Ahn, "Sunrise to sunset: Analyzing the end-to-end life cycle and effectiveness of phishing attacks at scale," in *Proc. 29th USENIX Secur. Symp. (USENIX Security)*. Berkeley, CA, USA: USENIX Association, Aug. 2020, pp. 361–377.

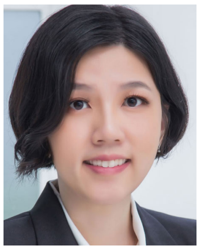
[149] S. Maroofi, M. Korczyński, and A. Duda, "Are you human? Resilience of phishing detection to evasion techniques based on human verification," in *Proc. ACM Internet Meas. Conf.*, Oct. 2020, pp. 78–86.

[150] W. Li, Y. He, Z. Wang, S. Alqahtani, and P. Nanda, "Uncovering flaws in anti-phishing blacklists for phishing websites using novel cloaking techniques," in *Proc. SECRYPT*, 2023, pp. 813–821.

[151] Y. Yuan, G. Apruzzese, and M. Conti, "Beyond the west: Revealing and bridging the gap between western and Chinese phishing website detection," *Comput. Secur.*, vol. 148, Jan. 2025, Art. no. 104115.
[152] Frontier-Enterprise. (2024). *Generative AI Phishing: Why Cybersecurity Training Must Evolve*. [Online]. Available: https://www.frontier-enterprise.com/generative-ai-phishing-why-cybersecurity-training-must-evolve/
[153] C. S. Eze and L. Shamir, "Analysis and prevention of AI-based phishing email attacks," *Electronics*, vol. 13, no. 10, p. 1839, May 2024.
[154] S. S. Roy, K. V. Naragam, and S. Nilizadeh, "Generating phishing attacks using ChatGPT," 2023, *arXiv:2305.05133*.
[155] C. Huang, Z. Zhang, B. Mao, and X. Yao, "An overview of artificial intelligence ethics," *IEEE Trans. Artif. Intell.*, vol. 4, no. 4, pp. 799–819, Aug. 2023.
[156] Deepgram. (2023). *From DAN To Universal Prompts: LLM Jailbreaking*. [Online]. Available: https://deepgram.com/learn/llm-jailbreaking
[157] S. Gallagher, B. Gelman, S. Taoufiq, T. Vörös, Y. Lee, A. Kyadige, and S. Bergeron, "Phishing and social engineering in the age of LLMs," in *Large Language Models in Cybersecurity: Threats, Exposure and Mitigation*. Cham, Switzerland: Springer, 2024, pp. 81–86.
[158] L. P. Kaelbling, M. L. Littman, and A. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, pp. 237–285, May 1996.
[159] A. M. S. N. Bibinbe, M. F. Mbouopda, G. R. M. Saleu, and E. M. Nguifo, "A survey on unsupervised learning algorithms for detecting abnormal points in streaming data," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2022, pp. 1–8.
[160] S. C. H. Hoi, D. Sahoo, J. Lu, and P. Zhao, "Online learning: A comprehensive survey," 2018, *arXiv:1802.02871*.
[161] X. Pan, T. Ye, D. Han, S. Song, and G. Huang, "Contrastive language-image pre-training with knowledge graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 22895–22910.
[162] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Gener. Comput. Syst.*, vol. 115, pp. 619–640, Oct. 2020.
[163] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 12, pp. 9587–9603, Dec. 2023.

WENHAO LI (Graduate Student Member, IEEE) received the Bachelor of Engineering degree in information security from Chengdu University of Information Technology, China, in 2019, the Master of Computer Science degree in cybersecurity from Arizona State University, and the M.B.A. degree from Webster University, USA, in 2022. He is currently pursuing the Ph.D. degree in cybersecurity with the Cybersecurity Research Centre, Universiti Sains Malaysia. He is also the CEO of Chengdu MeetSec Technology Company Ltd., where he has successfully led many cybersecurity services, projects, and developments. His current research interests include a wide range of cybersecurity topics, including anti-phishing, web security and privacy, cloud security, and the IoT security.

SELVAKUMAR MANICKAM (Member, IEEE) is currently the Director of the Cybersecurity Research Centre and a Professor of cybersecurity, the Internet of Things, industry 4.0, cloud computing, big data, and machine learning. He has authored and co-authored more than 220 papers in journals, conference proceedings, and book reviews. He has graduated 18 Ph.D. students in addition to master's and bachelor's students. He has given several keynote speeches and dozens of invited lectures and workshops at conferences, international universities, and industry. He has given talks and training on internet security, the Internet of Things, industry 4.0, IPv6, machine learning, software development, and embedded and OS kernel technologies at various organizations and seminars. He also lectures in various computer science and IT courses, including developing new courseware in tandem with current technology trends. He is involved in various organizations and forums locally and globally. Previously, he was with Intel Corporation and a few start-ups working in related areas before moving to academia. While building his profile academically, he is still very involved in industrial projects involving industrial communication protocol, robotic process automation, machine learning, and data analytics using opensource platforms. He also has experience in the building IoT, embedded, server, mobile, and web-based applications.



YUNG-WEY CHONG is currently a Senior Lecturer with the School of Computer Sciences, Universiti Sains Malaysia, where she has been a Faculty Member, since 2012. Her research interests include cyber-physical systems, wireless communications, cloud computing, and artificial intelligence. She is a Committee Member of SOI Asia, a project that utilizes satellite-based internet to support interactive multimedia communications between partner universities.

WEILAN LENG received the Bachelor of Engineering degree in information security from Chengdu University of Information Technology, China, in 2014, and the M.B.A. degree from Webster University, USA, in 2018. He has been with Chuanqing Drilling Engineering Company Ltd., China National Petroleum Corporation (CNPC), since 2014, where he is currently the Deputy Director of the Engineering Intelligent Operation Support Center, Research Institute of Drilling and Production Engineering Technology. His research interests include cybersecurity architecture, industrial control systems (ICS) security, DevSecOps, and AI for security.

PRIYADARSI NANDA (Senior Member, IEEE) is currently a Senior Lecturer with the University of Technology Sydney (UTS), with more than 33 years of experience and a strong researcher specializing in research and development of cybersecurity, the IoT security, internet traffic engineering, wireless sensor network security, and many more related areas. His most significant work has been in the area of intrusion detection and prevention systems (IDS/IPS) using image processing techniques, Sybil attack detection in the IoT based applications, and intelligent firewall design. In cybersecurity research, he has published over 130 high-quality refereed research articles, including IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, and *Future Generation Computer Systems*, and as many ERA Tier A/A* conference articles.