SCIENTIFIC Publishing Limited

Journal of Intelligent Communication
https://ojs.ukscip.com/index.php/jic

Article

Securing Online Platforms: Hybrid Machine Learning Approaches for URL Phishing Detections

Ugochukwu Onwudebelu [1,*] , John O. Ugah [1,2] , Samuel Elochukwu Ezeh [1] and Olusanjo Olugbemi Fasola [3]

[1] Department of Computer Science/Informatics, Alex Ekwueme Federal University Ndufu Alike (FUNAI), Abakaliki P.M.B. 1010, Nigeria

[2] Department of Computer Science, Ebonyi State University, Abakaliki P.M.B. 053, Nigeria

[3] Department of Cybersecurity, School of Information and Communication Technology, Federal University of Technology, Minna P.M.B 65, Nigeria

[*] Correspondence: ugochukwu.onwudebelu@funai.edu.ng

Received: 12 October 2025; **Revised:** 12 December 2025; **Accepted:** 16 December 2025; **Published:** 4 January 2026

Abstract: Phishing attacks pose significant risks in the digital landscape, resulting in financial losses and sensitive information breaches. Traditional detection methods often struggle to keep pace with evolving threats, compromising their effectiveness. This study addresses these limitations by developing a robust detection system using a hybrid machine learning approach. We combine random forest, gradient boosting, and logistic regression algorithms to enhance phishing detection accuracy. A labeled dataset of URLs from Kaggle is utilized, with robust feature engineering extracting key attributes for model training. Following the CRISP-DM framework and leveraging Object-Oriented Programming principles, we develop a model that achieves strong performance metrics. The model's accuracy stands at 84%, with precision, recall, and F1-score values of 85%, 86%, and 84%, respectively. Notably, the model demonstrates excellent ability to differentiate between phishing and legitimate URLs, with an ROC AUC score of 91%. These results confirm the model's potential as a reliable phishing detection tool, capable of identifying phishing URLs effectively while minimizing false positives. Our research contributes to the development of more effective phishing detection strategies, ultimately safeguarding users and organizations from economic and reputational harm. By leveraging machine learning, we can develop more robust cybersecurity systems. Our proposed model can be seamlessly integrated into existing security frameworks to improve the detection of phishing threats.

Keywords: Hybrid Machine Learning; URL Classification; Cybersecurity; Random Forest; Gradient Boosting; Phishing Detection

1. Introduction

The Internet has emerged as one of the most transformative and rapidly expanding technologies, with the number of global users increasing from 413 million in 2000 to 4.54 billion in 2020 [1]. While it has enabled unprecedented opportunities in communication, commerce, education, and entertainment, it has also created fertile ground for cybercriminal activity. Among the most prevalent threats are phishing attacks and malware distribution [2,3]. Phishing, a malicious form of online identity theft, is designed to obtain unauthorized access to sensitive

https://doi.org/10.54963/jic.v5i1.1693

user information by impersonating legitimate organizations [4]. Attackers typically deploy deceptive emails or fraudulent websites that mimic trusted entities to manipulate users into providing sensitive information such as passwords, personal identification numbers, credit card details, or banking information [5]. The harvested data are often exploited to compromise social media accounts, email services, or financial systems, leading to severe incidents of identity theft and monetary loss [6]. The growing sophistication of phishing techniques has heightened security concerns across critical sectors such as banking, e-commerce, and education. According to Abdelhamid et al., phishing-related theft costs U.S. banks and credit card companies approximately USD 2.8 billion annually [7]. The Anti-Phishing Working Group (APWG) [8] reported over 165,772 phishing sites detected in the first quarter of 2020 alone, underscoring the scale and persistence of this threat.

Phishing is now recognized as one of the most severe challenges in cybersecurity [9]. With global internet penetration reaching 59.5% and over 4.66 billion users by early 2021, an increase of 316 million users compared to the previous year [1,10], attackers have unprecedented access to potential victims. Cybercriminals often exploit social media platforms, emails, and online services by creating fraudulent websites and disseminating malicious links accompanied by urgent or alarming messages to elicit immediate responses from unsuspecting users [11,12]. Once sensitive credentials are provided, attackers misuse them for fraudulent transactions, blackmail, or broader cybercrimes. Phishing attacks have evolved beyond credential theft to also serve as vectors for distributing malicious software such as ransomware [12]. The APWG reported a dramatic escalation in phishing incidents during the COVID-19 pandemic, with the number of attacks more than doubling in 2020. Notably, over 225,304 new phishing sites were identified in October alone, marking the highest monthly record to date [8]. Similarly, the Internet Crime Complaint Center (IC3) received 241,342 phishing-related complaints in 2020, representing reported financial losses exceeding USD 54 million [13]. These statistics underscore the urgency of developing effective phishing detection mechanisms to safeguard unsuspecting internet users and mitigate both financial and emotional harm.

The rise of data science has opened new opportunities for combating such threats, as vast digital records can now be transformed into actionable intelligence through machine learning [14]. Data-driven solutions have already shown strong utility across domains such as business analytics, IoT, cybersecurity, and financial forecasting [15–17]. Applying these techniques to phishing detection provides a promising avenue for building adaptive and intelligent security models. Existing research on phishing website detection has explored diverse approaches. Early methods employed blacklist and whitelist techniques [18], content-based filtering [19], and visual similarity analysis. More recent efforts have investigated heuristic- and machine learning–based solutions [20]. Abdelhamid et al. [7] applied Multi-label Classifier–based Associative Classification (MCAC), achieving 94.5% accuracy. However, their work was limited by a relatively small dataset of 601 legitimate and 752 phishing sites and only 16 extracted features, leaving room for more robust feature engineering. Similarly, Aydin and Baykal [21] employed Naïve Bayes and Sequential Minimal Optimization across two feature subsets, such as CFS and Consistency, achieving accuracy rates ranging between 83.69% and 95.39%. While these approaches demonstrate potential, their reliance on restricted datasets and limited feature sets constrains generalizability to real-world phishing scenarios.

This study aims to create an efficient phishing detection system by selecting the relevant features and evaluating the effectiveness of various classification algorithms. To achieve this, the study introduces a hybrid machine learning model designed to differentiate phishing websites from legitimate ones almost instantly. Unlike conventional methods that depend on search engines, external services, or analysis of DNS and web traffic, the proposed approach focuses on extracting features directly from URLs, enabling faster detection. This strategy is crucial because phishing sites generally remain active for less than 10 hours on average, with nearly half being taken down within 24 hours. Nevertheless, compromised domains often persist beyond this window, underscoring the need for rapid and adaptive detection systems. Despite widespread awareness campaigns, users continue to face several challenges in recognizing phishing websites:

i. Limited understanding of URL syntax and structure.
ii. Uncertainty regarding which websites to trust.
iii. Inability to recognize redirections or hidden URLs.
iv. Lack of time to manually inspect website addresses.
v. Difficulty distinguishing legitimate from fraudulent websites.

To address these limitations, this study sets out the following specific objectives:

a. Compile a comprehensive labeled dataset of phishing and legitimate URLs from the Kaggle repository.
b. Extract relevant URL-based features indicative of phishing activity, including lexical characteristics, length, and keyword presence.
c. Implement a hybrid machine learning architecture that combines linear and non-linear models—specifically Random Forest (RF), Gradient Boosting (GB), and Logistic Regression (LR)—to enhance detection accuracy.
d. Train and optimize the hybrid model on the preprocessed dataset.
e. Design a system architecture that integrates the trained model for real-time classification of URLs.
f. Carry out the performance evaluation of the hybrid model using standard metrics such as accuracy, precision, recall, F1-score, as well as ROC-AUC.

Significance of the research

a. Government: Assures a functional system capable of reducing cyberattacks at the national level.
b. Users: Protects individuals from online fraud and identity theft while fostering greater awareness and vigilance in online interactions.
c. Organizations: Promotes a security-conscious culture, enabling employees to identify phishing attempts and thereby reducing financial and reputational losses.
d. Cybercrime Agencies: Reduces the investigative burden by filtering out fraudulent websites automatically, allowing agencies to focus on high-priority threats.

2. Related Works

Phishing detection has received extensive research attention due to the increasing sophistication of attack vectors and their persistent threat to online transactions. Existing countermeasures can broadly be classified into user education–focused approaches and software-based detection techniques, with recent research emphasizing automated and hybrid machine learning–driven solutions to overcome the limitations of human-centered defenses.

2.1. User Education–Focused Approaches

User awareness and training constitute an early line of defense against phishing attacks. Sheng et al. [22] demonstrated that Anti-Phishing Phil, a game-based learning system, significantly improved users' ability to identify fraudulent websites. Similarly, Kumaraguru et al. [23] employed email-based training mechanisms to educate users on recognizing malicious URLs, reinforcing human vigilance alongside automated systems. Arachchilage and Love [24] further grounded user-oriented interventions in the Technology Threat Avoidance Theory (TTAT), proposing game-based strategies to promote secure behavioral responses. While these studies confirm the value of user education in reducing susceptibility, they also highlight its inherent limitations. Human-dependent solutions are susceptible to fatigue, inconsistency, and delayed responses to rapidly evolving phishing strategies, rendering education insufficient as a standalone defense mechanism.

2.2. Software-Based Approaches

To address the scalability and adaptability limitations of user-centered methods, software-based phishing detection solutions have been extensively explored. These approaches are generally categorized into list-based and machine learning–based techniques.

2.2.1. List-Based Approaches

List-based detection techniques rely on predefined blacklists and whitelists to classify URLs. Wang et al. [25] and Han et al. [26] employed whitelist-oriented domain classification, while logo and favicon matching techniques were explored by Chiew et al. [20] and Rosiello et al. [27]. Conversely, blacklist-based approaches, such as those incorporating DNS and domain registration data [28], aim to block previously identified phishing sources. Despite their simplicity and low computational cost, list-based methods suffer from a critical weakness: they are ineffective against zero-day phishing attacks, as newly registered malicious domains often evade static lists. This limitation significantly restricts their applicability in dynamic threat environments.

2.2.2. Machine Learning–Based Approaches

Machine learning (ML) techniques have emerged as a more flexible and robust alternative, leveraging URL structures, webpage content, hyperlink relationships, and third-party metadata. Mohammad et al. [29] achieved high classification accuracy using self-structuring neural networks, albeit with increased computational overhead. Taeri et al. [30] focused on network-based inference to uncover phishing URLs masquerading as legitimate entities, while Mao et al. [31] employed CSS-based similarity metrics to detect visually deceptive websites. Subsequent research advanced ML performance through neural architectures and feature engineering. Feng et al. [32] reported high accuracy using a Monte Carlo–trained neural network, while Rao and Pais [12] combined heuristic URL features with image-based verification to improve zero-day detection at the cost of runtime efficiency. Hyperlink-based analysis introduced by Jain and Gupta [11] offered improved discrimination but remained ineffective for non-HTML content. More complex deep learning models, including capsule networks [33] and CNN-based architectures [34,35], demonstrated strong detection capability but raised concerns about model complexity and interpretability. Recent studies have explored lightweight and ensemble-based solutions. Random Forest–driven [36] systems such as CatchPhish [37] achieved competitive accuracy with reduced computational overhead, while regression-based and hybrid feature transformations [18,38] improved robustness. Emerging approaches leveraging generative adversarial networks and predictive ensembles [39], as well as LLM-based phishing detection systems [40], reflect growing interest in adaptive models. Nevertheless, challenges such as dataset imbalance, dependence on third-party sources, limited generalization, and high computational requirements persist.

2.2.3. Hybrid and Advanced Detection Approaches

Hybrid phishing detection approaches integrate multiple feature sources and classification techniques to capitalize on their complementary strengths. Do et al. [41] and Feng et al. [42] emphasized hybridization as a means to improve resilience against evasion tactics and enhance coverage across heterogeneous website components. Venugopal et al. [43] combined URL and HTML features within an ensemble framework; however, the reported performance (95.3%) was lower than that of certain standalone classifiers, highlighting that hybridization does not inherently guarantee improved accuracy. In contrast, Aljofey et al. [44] achieved 96.76% accuracy using a URL–HTML hybrid approach, though performance deteriorated when handling multilingual content. Vecliuc et al. [45] observed only marginal gains (96.5%) from integrating URL, HTML, and logo-based features, suggesting diminishing returns with increasing feature complexity. Recent deep learning–driven hybrids have reported higher detection accuracy. Web2Vec [42] integrated URL, HTML, and Document Object Model (DOM) features using CNN and LSTM architectures, achieving 99.05% accuracy. Despite strong performance, the approach suffers from black-box behavior, prolonged training time, and limited flexibility. Similarly, WebPhish [46] employed deep embeddings of raw URLs and HTML content, reporting 98.1% accuracy but showing reduced robustness to manipulated or obfuscated web content. Hybrid systems incorporating screenshots and logo analysis [18,47] further improved detection capability but exhibited susceptibility to false positives and visual obfuscation attacks. Collectively, these findings indicate that while hybrid models can enhance robustness, they often introduce trade-offs in efficiency, interpretability, and adaptability.

Beyond feature fusion, string-based URL analysis remains a core strategy, exploiting the observation that phishing URLs exhibit identifiable lexical patterns [48,49]. Empirical studies reveal a surge in phishing campaigns hosted on free website builders, accounting for approximately 81.7% of malicious domains [50]. Additionally, financially motivated malicious applications have proliferated through unregulated distribution channels such as SMS and instant messaging platforms [51,52]. Despite implementation differences, these malicious artifacts exhibit common behavioral traits, including sensitive data exfiltration and covert network activity. Investigations of malicious software repositories, such as PyPI and NPM, reveal shared authorship, reused codebases, and common remote content fetching mechanisms, highlighting coordinated attack strategies [53–55]. Advanced reinforcement learning–based techniques have framed phishing detection as a sequential decision-making problem. Double Deep Q-Networks (DDQNs) have shown promise in addressing class imbalance and concept drift in malicious URL datasets through adaptive learning [56]. Complementary to academic research, online security services such as VirusTotal provide large-scale, multi-engine analysis of URLs and files, enabling the detection of diverse threats including phishing, malware, and Trojans [57–60]. However, reliance on external services introduces dependency risks and limits real-time detection capabilities. Thus, existing hybrid and advanced phishing detection approaches demonstrate notable improvements in accuracy and robustness but often lack interpretability, computational efficiency, and cross-dataset generalization. These limitations motivate the need for a lightweight, interpretable hybrid framework that balances detection performance with adaptability to evolving phishing tactics.

2.3. Review of Hybrid Models in Machine Learning for Detecting Phishing URLs

Hybrid machine learning models have emerged as a promising solution to the limitations of single-feature phishing detection approaches. By integrating complementary information sources, these models can capture both structural and behavioral indicators of phishing activity. Prior studies highlight several advantages of hybridization, including improved detection performance through the combination of URL and hyperlink features, dynamic feature extraction capable of adapting to evolving phishing patterns, and increased versatility in identifying phishing, spoofing, and zero-day attacks. Furthermore, client-side implementations of hybrid models reduce dependency on external services, thereby minimizing latency and enhancing user privacy.

Despite these advantages, existing hybrid approaches exhibit notable limitations that restrict their practical deployment. First, empirical evidence regarding the effectiveness of hybridization remains inconsistent; while some studies demonstrate performance gains over standalone models, others report marginal improvement or even performance degradation. This inconsistency suggests that hybrid effectiveness is highly dependent on feature selection and integration strategy. Second, the trade-off between detection accuracy and computational efficiency is insufficiently explored, particularly in real-time and resource-constrained environments. Third, robustness against evasion techniques, including URL obfuscation and adversarial manipulation, is rarely evaluated systematically, limiting confidence in real-world resilience. Finally, many existing frameworks lack scalability and modularity, impeding adaptation to emerging phishing strategies and evolving web technologies.

These unresolved challenges underscore the need for a hybrid phishing detection framework that achieves a balance between accuracy, efficiency, interpretability, and robustness. Addressing this gap, the present study proposes a lightweight and modular hybrid machine learning approach that integrates URL-based and hyperlink-derived features, focusing on client-side applicability and adaptability. By emphasizing explainable feature engineering and flexible model integration, the proposed framework seeks to advance both the theoretical understanding and practical deployment of hybrid phishing detection systems.

3. Materials and Research Methodology

3.1. Research Framework

The development of the proposed hybrid feature-based URL phishing detection system integrates two complementary methodologies: the Cross-Industry Standard Process for Data Mining (CRISP-DM) and Object-Oriented Programming (OOP). The CRISP-DM framework provided a structured and iterative process for data understanding, preparation, modeling, evaluation, and deployment, ensuring systematic handling of the dataset and reproducibility of results. Conversely, the OOP paradigm was employed to design a modular, scalable, and maintainable system architecture, thereby facilitating seamless integration of multiple algorithms and future model enhancements. The synergy between CRISP-DM and OOP enabled the system to effectively capture and analyze data-driven behavioral patterns associated with phishing activities. The hybrid model integrates outputs of the three base learners (Random Forest, Gradient Boost, and Logistic Regression) using a soft-voting ensemble approach, where class probabilities from individual models are averaged to yield the final classification. This method leverages the interpretability of logistic regression and the non-linear decision power of tree-based models. During the classification phase, the hybrid model was trained using preprocessed data, validated on independent test data, and subsequently deployed for real-time predictions on unseen URLs. This process follows the standard supervised learning approach, wherein the algorithm learns to differentiate between legitimate and phishing instances through iterative optimization. **Figure 1** illustrates this conceptual workflow, analogous to traditional email classification processes where models distinguish between "spam" and "ham" (non-spam) categories.

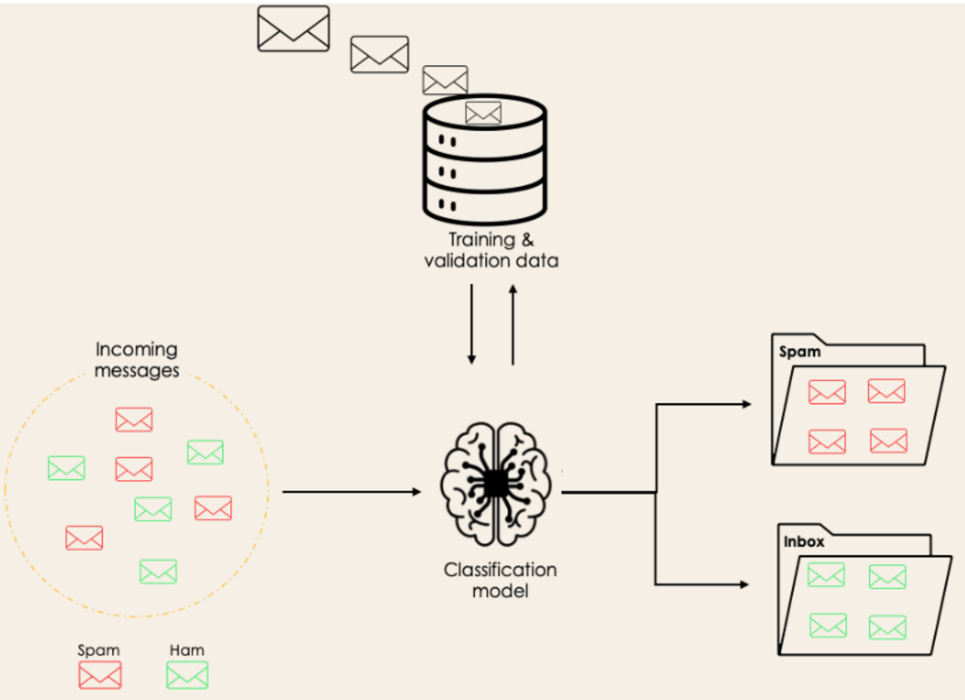
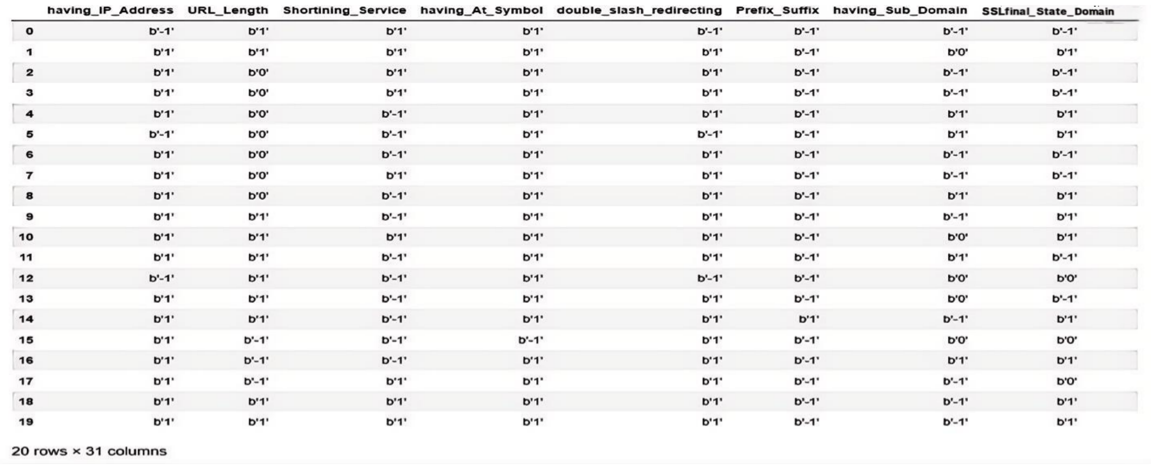


Figure 1. An instance of a Classification Model.

3.2. Dataset Description and Model Training

The experimental evaluation utilized the Kaggle phishing websites dataset as the primary benchmark for model training and validation. The dataset comprises 11,430 URL samples, evenly distributed between phishing and legitimate instances (50% each), thus ensuring balanced class representation. (i) Each URL is characterized by 87 distinct features drawn from three major categories: (ii) Structural and syntactic features (56): capturing lexical and morphological characteristics of the URL; (iii) Content-based features (24): derived from the HTML and page content of the corresponding websites; and (iv) external service features (7): obtained from WHOIS and other third-party sources to provide contextual metadata. Three ML classifiers: Random Forest (RF), Gradient Boosting (GB), and Logistic Regression (LR), were integrated into the hybrid model to leverage both linear and non-linear relationships within the data. These algorithms were trained and evaluated using the preprocessed dataset. A representative sample of the dataset is presented in **Figure 2.**



	having_IP_Address	URL_Length	Shortining_Service	having_At_Symbol	double_slash_redirecting	Prefix_Suffix	having_Sub_Domain	SSLfinal_State_Domain
0	b'-1'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'-1'	b'-1'
1	b'1'	b'1'	b'1'	b'1'	b'1'	b'-1'	b'0'	b'1'
2	b'1'	b'0'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'-1'
3	b'1'	b'0'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'-1'
4	b'1'	b'0'	b'-1'	b'1'	b'1'	b'-1'	b'1'	b'1'
5	b'-1'	b'0'	b'-1'	b'1'	b'-1'	b'-1'	b'1'	b'1'
6	b'1'	b'0'	b'-1'	b'1'	b'1'	b'-1'	b'-1'	b'-1'
7	b'1'	b'0'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'-1'
8	b'1'	b'0'	b'-1'	b'1'	b'1'	b'-1'	b'1'	b'1'
9	b'1'	b'1'	b'-1'	b'1'	b'1'	b'-1'	b'-1'	b'1'
10	b'1'	b'1'	b'1'	b'1'	b'1'	b'-1'	b'0'	b'1'
11	b'1'	b'1'	b'-1'	b'1'	b'1'	b'-1'	b'1'	b'-1'
12	b'-1'	b'1'	b'-1'	b'1'	b'-1'	b'-1'	b'0'	b'0'
13	b'1'	b'1'	b'-1'	b'1'	b'1'	b'-1'	b'0'	b'-1'
14	b'1'	b'1'	b'-1'	b'1'	b'1'	b'1'	b'-1'	b'1'
15	b'1'	b'-1'	b'-1'	b'-1'	b'1'	b'-1'	b'0'	b'0'
16	b'1'	b'-1'	b'-1'	b'1'	b'1'	b'-1'	b'1'	b'1'
17	b'1'	b'-1'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'0'
18	b'1'	b'1'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'1'
19	b'1'	b'1'	b'1'	b'1'	b'1'	b'-1'	b'-1'	b'1'

20 rows × 31 columns

Figure 2. A screenshot of the Kaggle dataset for Phighing websites.

The classification of websites is done using values 1 and −1, where 1 represents legitimate sites and −1 indicates phishing sites. Values typically range between two or three options, indicating the attribute's intensity or strength from low to high (**Table 1**).

Table 1. URL Attribute Classification.

Attribute	Attribute Type	Value
UsingIP, ShortURL, Symbol@, Redirecting, PrefixSuffix, DomainRegLen, Favicon, NonStdPort, HTTPS DomainURL, RequestURL, InfoEmail, AbnormalURL, StatusBarCust, DisableRightClick, UsingPopupWindow, IframeRedirection, AgeofDomain, DNSRecording, PageRank, GoogleIndex, StatsReport	Categorical	{−1,1}
LongURL, SubDomains, HTTPS, AnchorURL, LinksInScriptTags, ServerFormHandler, WebsiteTraffic, LinksPointingToPage	Categorical	{1,0,1}
WebsiteForwarding	Categorical	{0,1}

3.2.1. Data Pre-Processing

In this study, the preprocessing involves traditional machine learning operations—normalization and tokenization—to convert URL strings into numerical vectors. The term 'embedding' here does not refer to language model embeddings (LLM) but rather to a fixed-length numerical representation suitable for classical ML algorithms. The data preprocessing phase encompassed a series of critical operations designed to ensure data consistency, integrity, and suitability for model training. These operations included data normalization, tokenization, and word embedding. Initially, the Kaggle dataset was examined to confirm the absence of missing or inconsistent values. Subsequently, all numerical features were normalized to a uniform scale to prevent disproportionate influence from variables with larger magnitudes. For textual transformation, a character-level tokenization approach was implemented to convert URL strings into numerical representations. This was achieved using the Tokenizer function provided by the Keras library, which vectorizes text sequences into machine-interpretable arrays. To maintain uniform input dimensions, a fixed sequence length of 60 characters was adopted, reflecting the observed average URL length of 25–50 characters. URLs exceeding this threshold were truncated, while shorter URLs were zero-padded to ensure dimensional consistency. The dataset was divided into training and testing sets using an 80–20 split, with 80% allocated for training the model and 20% reserved for evaluating its performance on new, unseen data. This configuration aligns with standard machine learning practices for ensuring reliable model generalization.

3.2.2. Feature Extraction

Following preprocessing, a feature extraction module was developed to transform labeled raw URLs into embedding-based feature representations, facilitating effective model learning. Since machine learning classifiers cannot directly process textual input, this step converted URLs into structured numerical feature sets. Although the Kaggle dataset includes 87 features, this study employed a subset of 25 hybrid features (URL and hyperlink-based) relevant to client-side detection. The reduction aimed to eliminate third-party and server-dependent features, ensuring independence from external APIs and improving system scalability for real-time implementation. A total of 25 discriminative features were extracted and organized into two primary categories: URL-based features, capturing lexical, structural, and syntactic properties of the URLs, and Hyperlink-based features, reflecting internal and external link behaviors within web pages. These extracted features, summarized in **Table 2**, served as the foundation for the classification of web pages into legitimate or phishing categories. The study exclusively utilized client-side features, thereby eliminating dependencies on third-party APIs and search engines—an approach that enhances privacy, independence, and system scalability. Feature values were encoded in a binary format, where 0 represents legitimate instances and 1 indicates phishing behavior.

Table 2. Categorical Features Used in This Study.

S/N	Category	Features Name	Total Features
1	URL-based features	Domain-URL, Count of Subdomains-URL, IP Address-URL, "@" Symbol-URL, Length of URL, Depth of URL, Redirection "//"-URL, "http/https"-Domain Name, HTTPS Scheme, URL Shortener services (e.g., TinyURL), Prefix or Suffix "-" Domain, Presence of Sensitive Words, Presence of Popular Brand Names, Uppercase Letters usage, Number of Dots in URL.	15
2	Hyperlink-based features	Missing hyperlink, Site-internal link ratio, off-site Hyperlink Ratio, Embedded/Linked CSS, Questionable form submission, Blank Hyperlink, Favicon type (Internal/External logo), Page duplication rate, Duplicate footer links, HTTP request handler	10

3.2.3. URL-Based Features

A Uniform Resource Locator (URL) is a standardized format used to specify the location of digital resources such as web pages, images, audio, and video files on the Internet. **Figure 3** depicts the typical URL structure, which includes several components that together define the resource's exact location and access method. URLs start with a protocol identifier, for example, HTTP, HTTPS, or FTP, which designates the communication protocol for fetching the resource. Of these, HTTPS (Hypertext Transfer Protocol Secure) is regarded as the most secure, providing encryption and authentication to ensure data integrity during transmission.

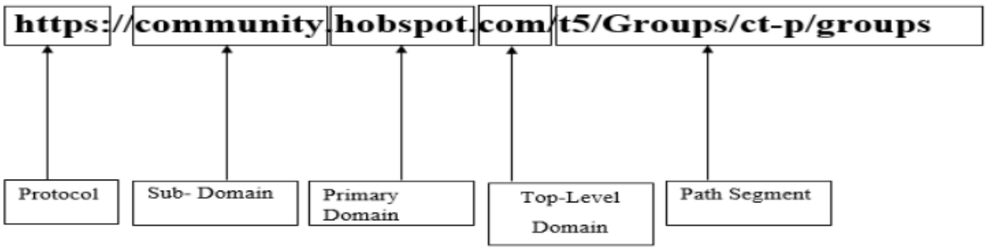


Figure 3. URL Pattern.

After the protocol, the hostname indicates the server that hosts the requested resource. The hostname (IP address) itself can be broken down into three levels: the subdomain, which comes before the main domain and often points to a specific service or department; the primary domain, representing the main organization or entity; and the top-level domain (TLD), which can be either generic (gTLD) like .com or .org, or country-code (ccTLD) such as .ng or .uk. Following this is the path segment, which designates the exact location of the resource within the server's directory, separated from the domain by a single forward slash ('/'). URLs may also contain optional parts: a query string, beginning with a question mark ('?'), that carries additional parameters or user information, and a fragment identifier, starting with a hash symbol ('#'), which usually points to a specific section of the webpage. Recognizing these URL components is crucial because phishing sites often exploit elements like subdomains, TLDs, or query strings to trick users and imitate legitimate websites. Therefore, analyzing these URL features is vital for effective phishing detection. The standard URL format is as follows:

<Protocol>://<Subdomain>.<Primarydomain>.<TLD>/<Pathdomain><?query><#fragmnt>

Phishing detection often involves identifying URL manipulations that trick users into believing they are on legitimate websites. Attackers use URL obfuscation techniques, altering key components such as the primary domain, subdomain, and path to mask malicious intent. This study extracts and encodes various URL-based features to effectively distinguish phishing from legitimate sites.

i. Domain Name: The full domain, excluding the "www." prefix, is extracted but omitted from model training due to its low discriminative value.
ii. Subdomain Count: Counts the number of dots in the hostname; legitimate URLs usually have two dots (excluding "www"). URLs with three dots are labeled suspicious (value = 0.5), and those with more than three are labeled phishing (value = 1).
iii. IP Address in Domain: Using an IP address in place of a domain is a strong phishing indicator, as legitimate sites rarely do this.
iv. "@" Symbol: The presence of "@" causes browsers to ignore preceding parts of the URL, a tactic phishers exploit. Its presence scores 1 (phishing), absence scores 0 (legitimate).
v. URL Length: Phishing URLs tend to be unusually long. URLs over 75 characters but under 100 are suspicious (0.5), while longer ones are classified as phishing.
vi. URL Depth: Measures the number of directory levels in the path; deeper paths are common in phishing URLs.
vii. Double Slash "//": An extra "//" beyond the protocol may indicate redirection to a phishing site; presence scores 1, absence 0.
viii. "http/https" in Domain: Appearance of these tokens within the domain suggests phishing (1), else (0).

ix. HTTPS in Scheme: "https" presence suggests legitimacy (0), absence or insecure protocol suggests phishing (1), though fake certificates reduce reliability.
x. URL Shorteners: Use of shortening services (e.g., TinyURL) is flagged as phishing (1), otherwise (0).
xi. Hyphen "-" in Domain: Hyphens are uncommon in legitimate domains but frequently appear in phishing URLs.
xii. Existence of Sensitive Words: Phishing URLs frequently contain trigger terms such as login, update, validate, activate, or secure to induce user urgency. A curated list of 18 such terms is used; the feature value is 1 if any term appears, otherwise 0.
xiii. Existence of Trendy Brand Names: Phishers often incorporate recognizable brand names within URLs to deceive users into believing they are visiting official sites. A list of 19 frequently targeted brands is maintained for detection.
xiv. Existence of Uppercase Letters: Legitimate URLs typically use lowercase characters. The presence of uppercase letters within a URL is treated as a phishing indicator.
xv. Number of Dots: More than two dots generally increase phishing likelihood.

3.2.4. Hyperlink-Based Feature

This section examines hyperlink properties extracted from the website's source code, using the Document Object Model (DOM) tree structure to analyze hyperlinks hierarchically and systematically (see **Figure 4**).

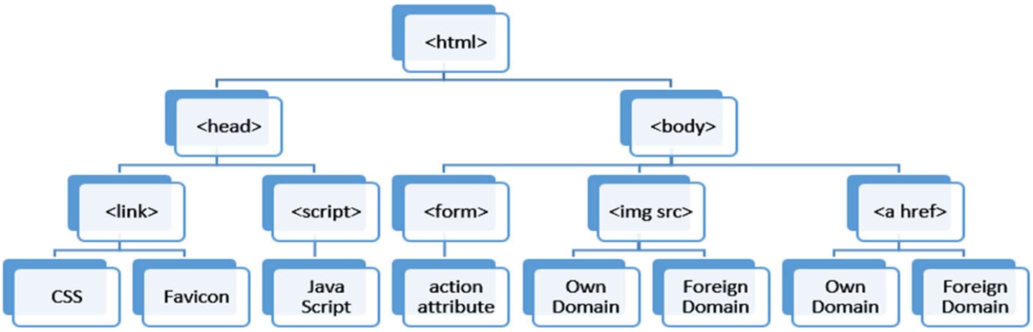


Figure 4. HTML DOM Tree.

The DOM facilitates the dynamic access and manipulation of web elements such as tags, IDs, classes, attributes, and structures within a webpage. During feature extraction, the DOM is instrumental in identifying and analyzing elements such as links, forms, source (src) attributes, and anchor (a) tags. **Figure 4** illustrates a typical DOM tree structure. From the DOM, ten hyperlink-based features were extracted to capture structural and behavioral differences between legitimate and phishing websites.

i. Number of Hyperlinks: Legitimate websites usually have many web pages and corresponding hyperlinks, while phishing sites tend to have fewer or hidden links. As noted in **Figure 4**, genuine sites generally contain at least one hyperlink in their source code. This feature counts all href, link, and src tags. A count of zero indicates phishing (value = 1), while any nonzero count indicates legitimacy (value = 0).
ii. Internal Hyperlink Ratio: Internal links point to the same base domain. Phishing sites often copy legitimate templates and retain internal links to the original domain. This ratio is calculated by dividing internal links by total links. A ratio of 0.5 or higher suggests a legitimate site (value = 0), and a ratio below 0.5 suggests phishing (value = 1).
iii. External Hyperlink Ratio: External links direct users to different domains. Phishing websites often have a higher proportion of external links to redirect users or load content. Legitimate sites keep this ratio low. Ratios below 0.5 are marked legitimate (0), and 0.5 or above are flagged phishing (1).
iv. Internal/External CSS Links: CSS files define a webpage's appearance. Attackers often use external CSS copied from real sites to create convincing phishing pages. If CSS files link to external domains, the feature is set to 1 (phishing); if internal, 0 (legitimate).
v. Suspicious Form Links: Phishing pages may use deceptive login forms pointing to external URLs, PHP scripts,

or placeholders like # or javascript: void(). Legitimate forms link to their own domain. This binary feature marks external or placeholder actions as suspicious (1) and local actions as legitimate (0).

vi. Null Hyperlinks: Phishing sites frequently use anchor tags linking back to the same page (e.g., <a href="#">) to trap users. This ratio of null links to total anchors above 0.34 indicates phishing (1); otherwise, it is legitimate (0).

vii. Internal/External Favicon: Favicons indicate site identity. If a favicon is linked from an external domain, it usually signals phishing, as attackers copy favicons to appear authentic. Internal favicons score 0 (legitimate), external ones score 1 (phishing).

viii. Common Page Detection Ratio: Phishers tend to redirect many anchors to a few pages, resulting in a high ratio of repeated links. A higher ratio raises suspicion of phishing.

ix. Common Page in Footer Section Ratio: Focused on footer links, this metric detects phishing by identifying repetitive or limited links commonly found in phishing footers.

x. Server Form Handler (SFH): The SFH attribute in form tags is examined. Empty, placeholder, or external domain values indicate phishing. This ternary feature assigns 0 for legitimate, 0.5 for suspicious, and 1 for phishing.

3.2.5. Hybrid Features

To boost classification performance, URL-based and hyperlink-based features were combined into a hybrid set of 25 attributes. The domain name was excluded due to its non-numerical nature and limited relevance. Supervised ML algorithms—RF, GB, and LR—were trained on labeled datasets to map feature patterns to the categories of legitimate or phishing websites. Models were validated on unseen data to improve detection accuracy.

3.3. High-Level Model Overview

The high-level model provides a broad view of the system's architecture, outlining key components, workflows, and interactions involved in phishing detection (**Figure 5**).

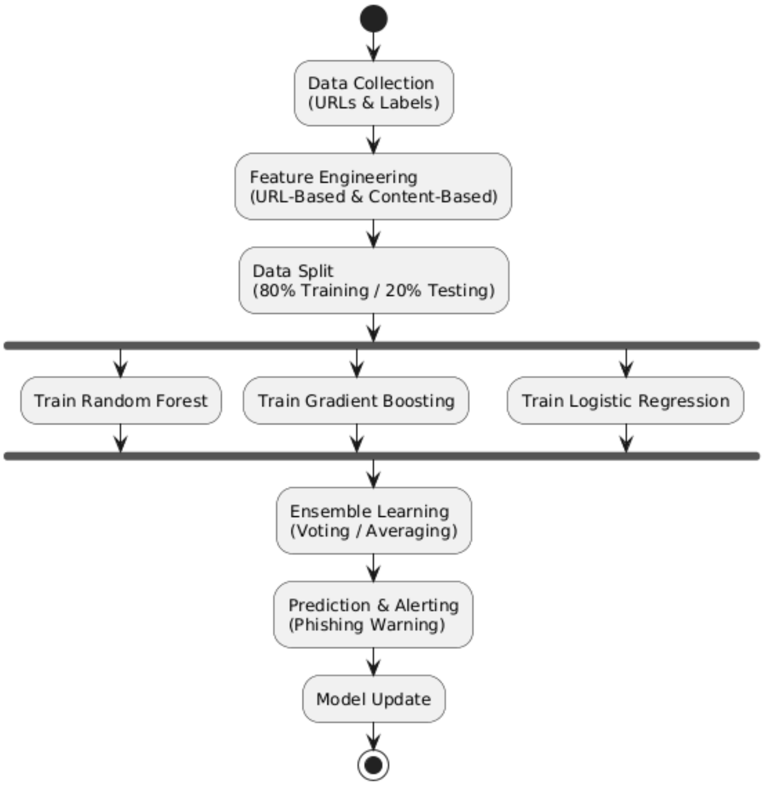


Figure 5. High-Level Model Overview.

The hybrid architecture adopts a soft-voting ensemble strategy. Each base learner (Random Forest, Gradient Boost, and Logistic Regression) outputs class probabilities. These probabilities are averaged to form the final classification decision. This ensures that both linear and non-linear decision boundaries are captured effectively. The ensemble operates in a parallel fashion rather than a stacked meta-classifier, thereby minimizing model complexity while preserving interpretability. The proposed phishing detection system combines both URL-based and hyperlink-based features and incorporates a blend of linear and non-linear machine learning algorithms—RF, GB, and LR—as depicted in **Figure 5**. This hybrid framework delivers a robust and comprehensive strategy for identifying and countering phishing threats across varied web contexts.

i. Comprehensive Feature Extraction: The system uses a dual extraction process that captures both URL-level and hyperlink-level features, enabling it to detect a wide range of phishing indicators. This includes recognizing suspicious keywords, misleading domain structures, and unusual use of URL shorteners, thus improving detection of both traditional and new phishing methods.
ii. Hybrid Learning Strategy: The approach combines the advantages of linear and non-linear models. Logistic Regression offers interpretability and models linear relationships, while ensemble methods like Random Forest and Gradient Boosting handle complex non-linear interactions. Their integration enhances model robustness and reduces overfitting risks.
iii. Adaptability and Continuous Learning: The system is designed to dynamically adapt to evolving phishing strategies by retraining on updated datasets. Continuous model evaluation ensures that new patterns in phishing behavior are incorporated, thereby maintaining model relevance in the constantly changing cybersecurity landscape.
iv. Enhanced Accuracy and Reduced False Positives: The integration of multiple algorithms in a hybrid architecture enhances detection accuracy. The system effectively balances interpretability and predictive power, achieving improved classification performance while minimizing false positives. This reduces unnecessary disruptions to legitimate users and strengthens confidence in automated phishing detection.
v. Economic and Operational Benefits: The deployment of the proposed system yields tangible operational and economic advantages. By proactively preventing phishing-induced financial losses, data breaches, and reputational harm, organizations can achieve significant returns on investment (ROI). Additionally, enhanced detection performance contributes to a safer user experience and fosters long-term trust in online platforms.

3.3.1. Hybrid Model Design and Prediction Workflow

(1) Hybrid Model Integration: Outputs from individual machine learning models are aggregated into a unified hybrid framework. This ensemble approach capitalizes on the predictive capabilities of both linear and non-linear models to deliver higher accuracy and generalizability across datasets.
(2) Prediction and Output Generation: The prediction process involves several key steps: (i) Extraction of features from the new URL using the established preprocessing pipeline. (ii) Feeding of the extracted feature vector into the trained hybrid model. (iii) Generation of a binary classification output—phishing or legitimate—accompanied by a probability-based confidence score.

3.3.2. System Architecture

Figure 6 illustrates the system's architecture. The process begins with input URLs destined for classification as legitimate or phishing. Feature extraction modules analyze attributes such as IP addresses, URL length, domain components, and favicon presence to build a structured feature vector. This vector encodes values as 1, 0, or −1, reflecting feature presence, irrelevance, or absence. These encoded features are fed into the ensemble classifiers—Logistic Regression, Gradient Boosting, and Random Forest—which are trained on labeled data to identify distinctive patterns. The combined hybrid classifier then assesses new URLs, producing classification results. Model performance is measured using accuracy, precision, recall, and F1-score to confirm the system's effectiveness in realistic phishing detection environments.

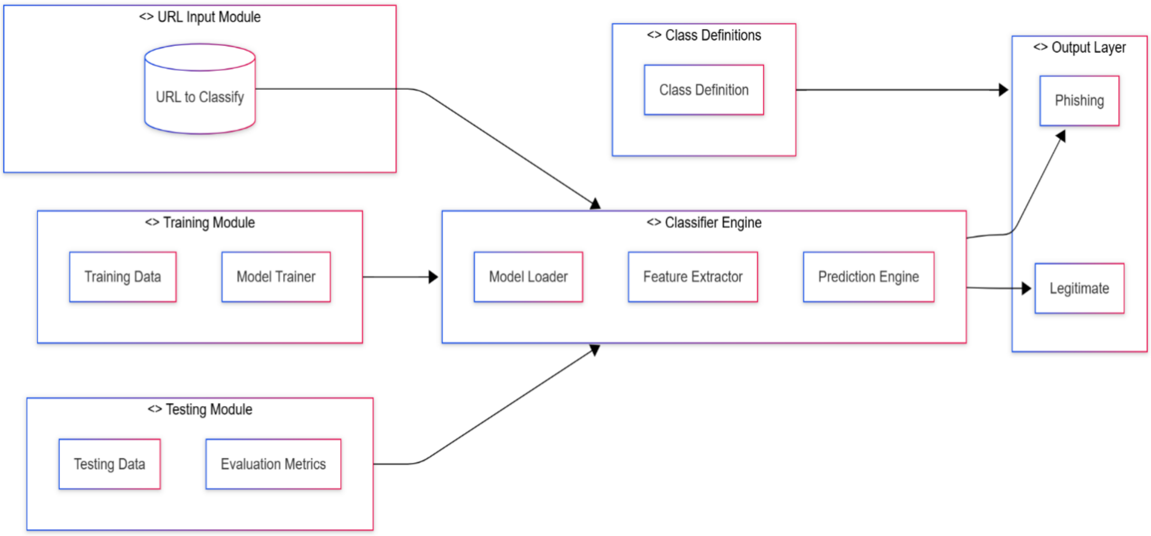


Figure 6. System Architecture.

System Sequence Diagram

A system sequence diagram (SSD) is a specific type of UML sequence diagram focused on illustrating the sequence of events generated by external actors interacting with the system, as shown in **Figure 7**. While typical sequence diagrams depict event progression over time, SSDs provide detailed sequences for distinct use case scenarios. Use case diagrams, by contrast, represent user interactions in a broader sense. The SSD focuses on the step-by-step flow within a particular use case instance.

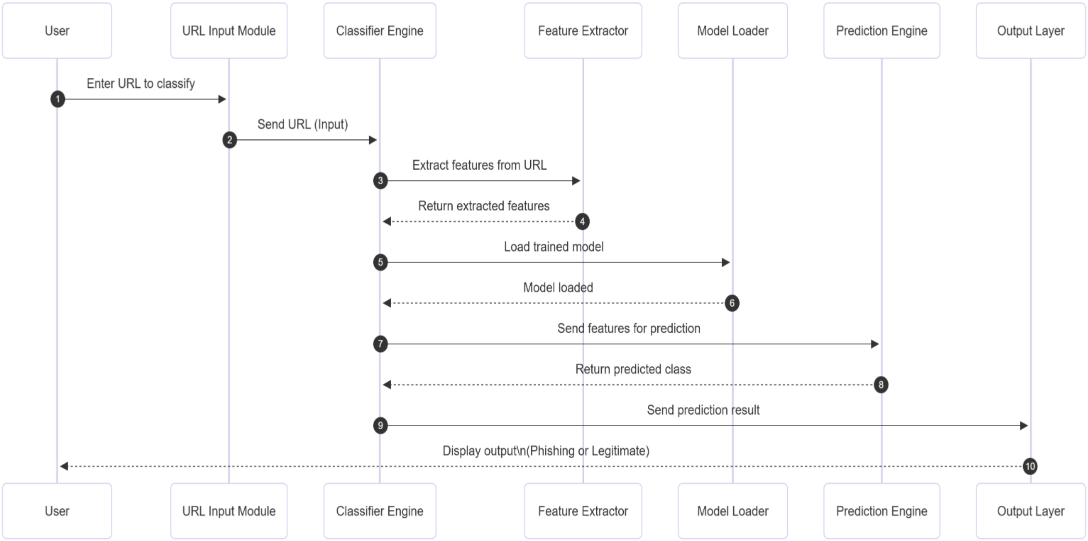


Figure 7. Sequence Diagram.

4. Discussion and Results

4.1. Model Performance Metrics

Model performance was assessed using key metrics, with their formulas presented in Equations (1)–(4):

Accuracy (Equation 1):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Precision (Equation 2):

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Recall (Equation 3):

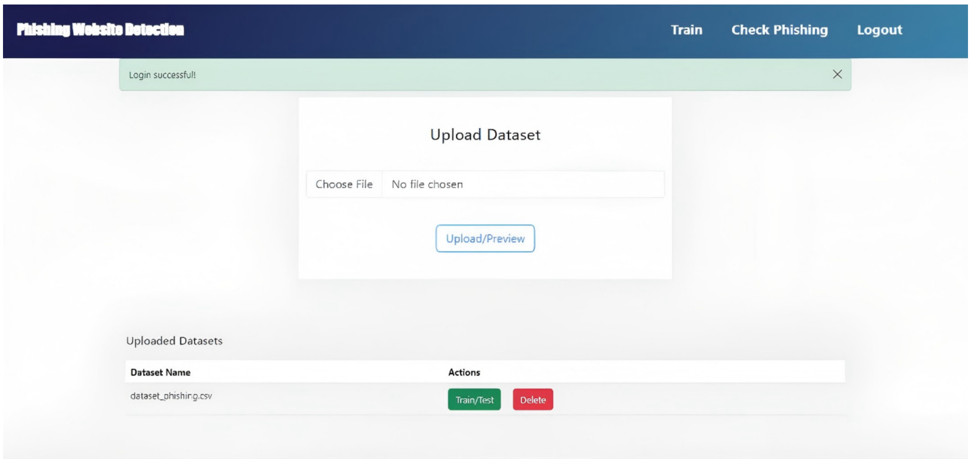
$$Recall = \frac{TP}{TP + FN} \tag{3}$$

F1-score (Equation 4):

$$F1 - score = 2 * \left(\frac{precision * Recall}{Precision + Recall}\right) \tag{4}$$

i. Input Format: The input format of the system is designed to accept datasets or URLs from users. **Figure 8** illustrates an example of an admin inputting the dataset for training on the system.

ii. Output Format: The output format shows the result of the data that the system collected from the user. **Figure 9** is a display format for an uploaded dataset.

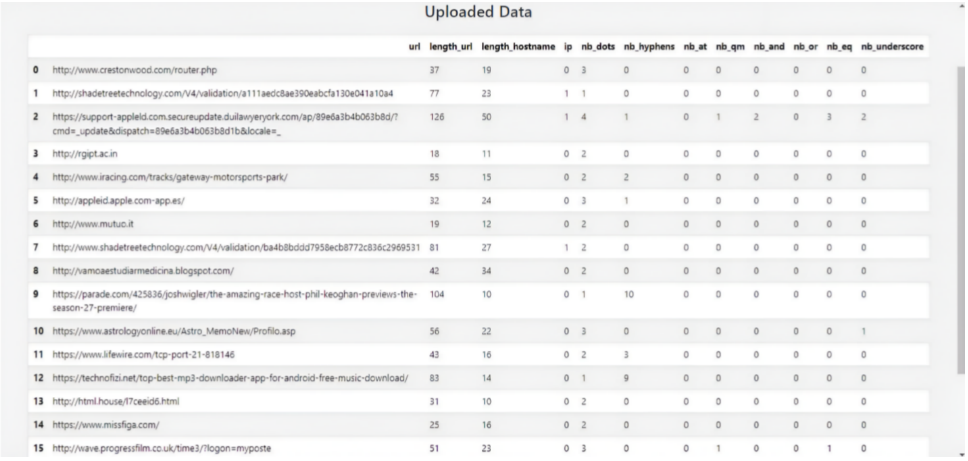


Login successful!

Uploaded Datasets

Dataset Name	Actions
dataset_phishing.csv	Train/Test Delete

Figure 8. Input format for uploading a dataset.



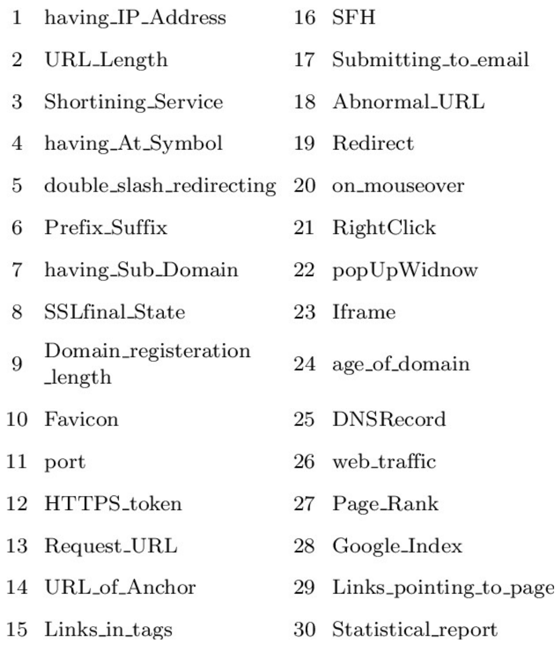
Uploaded Data

	url	length_url	length_hostname	ip	nb_dots	nb_hyphens	nb_at	nb_qm	nb_and	nb_or	nb_eq	nb_underscore
0	http://www.crestonwood.com/router.php	37	19	0	3	0	0	0	0	0	0	0
1	http://shadetreetechnology.com/V4/validation/a111aedc8ae390eabcfa130e041a10a4	77	23	1	1	0	0	0	0	0	0	0
2	https://support-appleld.com.secureupdate.duilawyeryork.com/ap/89e6a3b4b063b8d/?cmd=_update&dispatch=89e6a3b4b063b8d1b&locale=_	126	50	1	4	1	0	1	2	0	3	2
3	http://rgipt.ac.in	18	11	0	2	0	0	0	0	0	0	0
4	http://www.iracing.com/tracks/gateway-motorsports-park/	55	15	0	2	2	0	0	0	0	0	0
5	http://appleid.apple.com-app.es/	32	24	0	3	1	0	0	0	0	0	0
6	http://www.mutuo.it	19	12	0	2	0	0	0	0	0	0	0
7	http://www.shadetreetechnology.com/V4/validation/ba4b8bddd7958ecb8772c836c2969531	81	27	1	2	0	0	0	0	0	0	0
8	http://vamoaestudiarmedicina.blogspot.com/	42	34	0	2	0	0	0	0	0	0	0
9	https://parade.com/425836/joshwigler/the-amazing-race-host-phil-keoghan-previews-the-season-27-premiere/	104	10	0	1	10	0	0	0	0	0	0
10	https://www.astrologyonline.eu/Astro_MemoNew/Profilo.asp	56	22	0	3	0	0	0	0	0	0	1
11	https://www.lifewire.com/tcp-port-21-818146	43	16	0	2	3	0	0	0	0	0	0
12	https://technofizi.net/top-best-mp3-downloader-app-for-android-free-music-download/	83	14	0	1	9	0	0	0	0	0	0
13	http://html.house/l7ceeid6.html	31	10	0	2	0	0	0	0	0	0	0
14	https://www.missfiga.com/	25	16	0	2	0	0	0	0	0	0	0
15	http://wave.progressfilm.co.uk/time3/?logon=myposte	51	23	0	3	0	0	1	0	0	1	0

Figure 9. Output format of the uploaded dataset.

Dataset Selection: The initial phase involved choosing an appropriate dataset, sourced from the Kaggle repository, due to several advantages:

i. The dataset's large size offers a comprehensive foundation for training.
ii. It contains 30 features, providing a broad feature space that can improve prediction accuracy (see **Figure 10** for details).
iii. URLs are approximately evenly distributed between legitimate and phishing classes, ensuring balanced training data.



1	having_IP_Address	16	SFH
2	URL_Length	17	Submitting_to_email
3	Shortining_Service	18	Abnormal_URL
4	having_At_Symbol	19	Redirect
5	double_slash_redirecting	20	on_mouseover
6	Prefix_Suffix	21	RightClick
7	having_Sub_Domain	22	popUpWidnow
8	SSLfinal_State	23	Iframe
9	Domain_registeration _length	24	age_of_domain
10	Favicon	25	DNSRecord
11	port	26	web_traffic
12	HTTPS_token	27	Page_Rank
13	Request_URL	28	Google_Index
14	URL_of_Anchor	29	Links_pointing_to_page
15	Links_in_tags	30	Statistical_report

Figure 10. The features in the dataset.

Source: Kaggle (https://www.kaggle.com).

4.1.1. Feature Extraction

Feature values were extracted with Python libraries such as whois, requests, socket, re, ipaddress, and BeautifulSoup to obtain information on IP addresses, URL length, domain name, subdomains, favicon presence, and others. These features are stored as a list matching the dataset's format. When a new URL is input, it is converted into a Python list containing 30 elements representing these features, which the trained classifiers—Logistic Regression, Random Forest, and Gradient Boosting—then analyze for classification.

4.1.2. Feature Importance and Ablation Study

Feature importance was analyzed using the Gini impurity scores from the Random Forest model. The most influential features included URL length, presence of the '@' symbol, number of dots, HTTPS usage, and domain age. To further validate feature contribution, an ablation study was conducted by iteratively removing top-ranked features and measuring the performance drop. The F1-score declined by 4.1% when the top five features were excluded, confirming their strong predictive relevance.

4.2. Hardware and Software Requirements

Hardware:

i. CPU: Multi-core processor with at least 2.5 GHz
ii. RAM: Minimum 8 GB
iii. Storage: At least 64 GB

iv. OS: 64-bit Windows or Linux
v. Peripherals: Mouse, SVGA monitor, enhanced keyboard

Software:

vi. Programming language: Python (version 3.2 or higher)
vii. ML libraries: scikit-learn, TensorFlow, or PyTorch
viii. Data processing: Pandas, NumPy
ix. Visualization: Matplotlib, Seaborn
x. Dataset: Balanced set of phishing and legitimate URLs
xi. IDE: PyCharm or Visual Studio Code
xii. Database: MySQL
xiii. Browsers: Chrome or UC Browser

4.3. Results

This section outlines the results of the study. The evaluation of the hybrid model combining RF, GB and LR yielded promising results with balanced metrics across the board. By using the Equations (1)–(4) where: TP = 940; FP = 209; FN = 160 and TN = 977, we obtain the values in **Table 3**. Thus, the phishing detection system evaluation of the hybrid model can be seen in **Table 3**.

Table 3. Evaluation result of the hybrid model.

Precision	Recall	Accuracy	F1-Score	ROC AUC
0.85	0.86	0.84	0.84	0.91

The evaluation results indicate a promising phishing detection system with balanced performance across various metrics. Here's a detailed breakdown;

i. Precision: 0.85 indicates a high proportion (85%) of flagged phishing attempts are indeed phishing sites. The system avoids a large number of false positives that could inconvenience users.
ii. Recall: 0.86 indicates the system identifies 86% of actual phishing attempts, demonstrating some success in catching malicious URLs.
iii. F1-Score (0.84): This balanced metric reflects the trade-off between precision (avoiding false positives) and recall (catching phishing attempts).
iv. Accuracy (0.84): This signifies the system correctly classified 84% of URLs in the test data. It demonstrates a good overall ability to distinguish between phishing and legitimate websites.
v. ROC AUC (0.91): This is a very high score. An AUC of 0.91 indicates exceptional ability to distinguish between phishing and legitimate URLs overall, even though the accuracy is low.

4.3.1. Baseline Performance Comparison

To demonstrate the benefit of hybridization, **Table 4** is a short results table comparing the individual model performances (accuracy, precision, recall, F1-score, AUC) versus the hybrid model, that is, the baseline comparisons with individual classifiers (Random Forest, Gradient Boost, and Logistic Regression). The results (**Table 4**) show that the hybrid ensemble achieved an accuracy of 84%, outperforming individual models. This confirms that hybridisation improves classification performance.

Table 4. Baseline Comparison of Individual Classifiers vs. Hybrid Model.

Model	Accuracy	Precision	Recall	F1-Score	AUC
Logistic Regression	77%	78%	79%	78%	0.84
Random Forest	81%	82%	83%	82%	0.88
Gradient Boost	80%	81%	82%	81%	0.87
Hybrid (Proposed)	84%	85%	86%	84%	0.91

Logistic Regression, while interpretable, captures linear patterns and therefore performs moderately. Random Forest and Gradient Boosting perform better due to their ability to model nonlinear relationships, but both still fall short of the hybrid approach. The hybrid model integrates the strengths of both linear and non-linear classifiers, resulting in improved generalization and higher discriminative capability. The hybrid achieved a 4–7% relative improvement across key metrics, confirming that the hybridisation strategy delivers measurable performance gains.

4.3.2. Experimental Analysis

(1) Confusion Matrix (CM)

The confusion matrix is a visual representation that summarizes a classifier's correct and incorrect predictions, helping evaluate its performance. As depicted in **Figure 11**, the matrix consists of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). **Figure 12** shows the confusion matrix specific to the hybrid model used in this study.

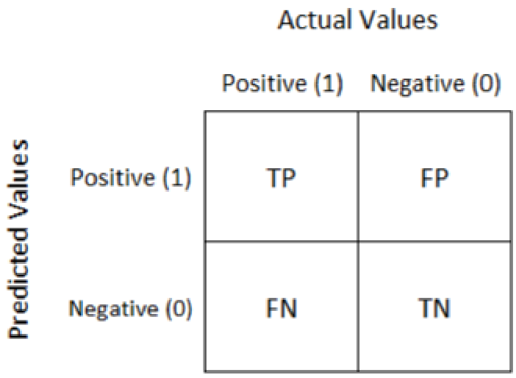


Figure 11. Confusion Matrix.

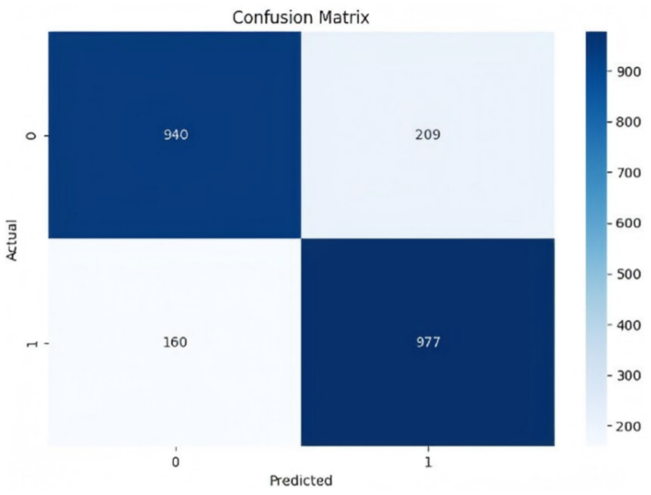


Figure 12. Confusion Matrix of the hybrid model.

(2) The Correlation Map

The correlation map illustrates the relationships between discrete values, with lighter boxes indicating strong positive correlations and darker boxes indicating weak or negative correlations. If all values were strongly correlated, classification might be unnecessary for making accurate predictions. However, the classifier leverages the differences in correlations to identify patterns and find an optimal solution. Notably, the map reveals low correlations between certain features, such as URL depth and domain length or dots in the domain and entropy. This particular feature combination proved challenging to interpret, as reflected in the map. The lack of correlation between URL length and domain length suggests that URL lengths are arbitrarily chosen, regardless of page depth. This is evident in the varying URL lengths, which can range from short to extremely long (up to 80 or 100 characters) without a discernible pattern. The correlation map for the hybrid model is presented in **Figure 13**.

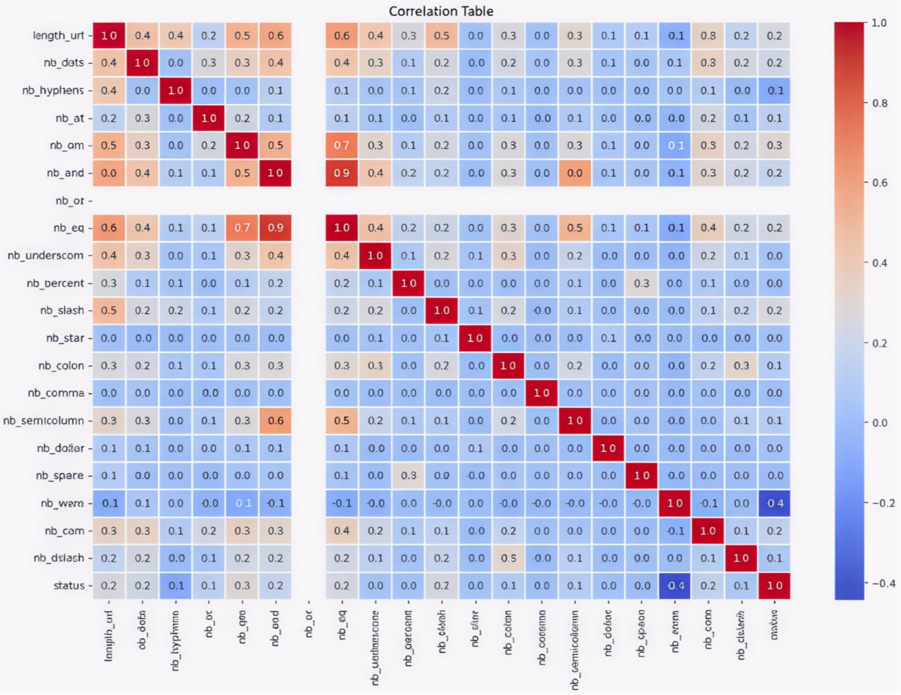


Figure 13. Correlation Map.

(3) ROC Curve

The Receiver Operating Characteristic (ROC) curve is a graphical tool that assesses the hybrid model's ability to distinguish between phishing (positive) and legitimate (negative) classes over various threshold settings. **Figure 14** illustrates the ROC curve for the hybrid model, highlighting its discriminative power across different classification cutoffs.

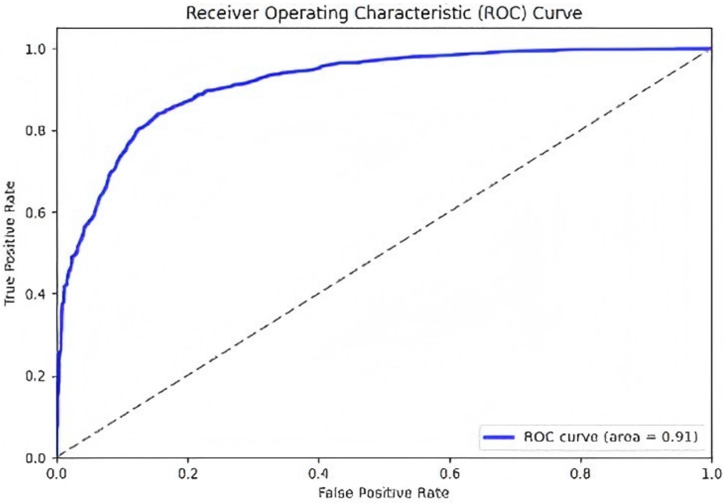


Figure 14. ROC Curve.

(4) Classification Report

The classification report compiles the hybrid model's performance results on the binary classification of phishing versus legitimate URLs. It includes key metrics such as Precision, Recall, F1-score, and Support:

i. Classes: The model is evaluated on two categories—phishing (positive) and legitimate (negative).

ii. Precision: This measures the proportion of URLs labeled as phishing that are genuinely phishing. It reflects the model's ability to reduce false positives, calculated as the ratio of true positives to the sum of true and false positives. In other words, it answers: "Of all URLs predicted to be phishing, how many were correct?"

iii. Recall (True Positive Rate): Recall indicates how effectively the model detects actual phishing URLs. It's the ratio of true positives to the sum of true positives and false negatives. It answers: "Of all phishing URLs present, how many were identified?"

iv. F1-Score: The F1-score harmonizes precision and recall, providing a single balanced measure of classification accuracy, particularly valuable when false positives and negatives carry similar costs. Higher F1 values indicate better overall performance.

v. Support: This metric shows the number of samples for each class in the evaluation set, offering insight into the class distribution and ensuring meaningful evaluation

The classification report of the hybrid phishing detection model is presented in **Figure 15**, illustrating how effectively the ensemble approach—combining Random Forest, Gradient Boosting, and Logistic Regression—balances detection accuracy, precision, and recall across both classes. This performance analysis validates the hybrid model's reliability and its potential for practical deployment in real-time phishing detection systems.

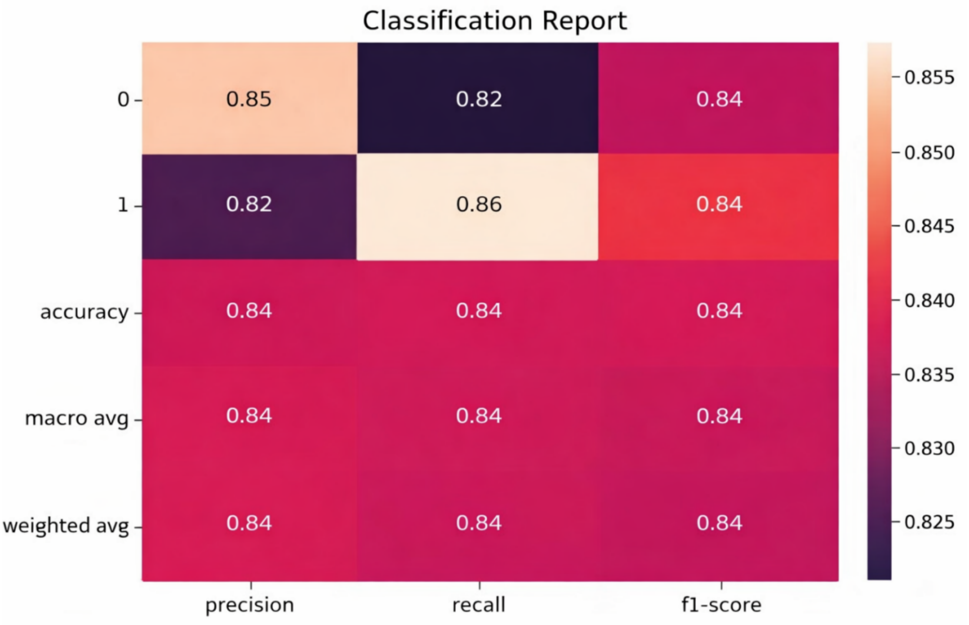


Figure 15. Classification Report.

These results suggest a well-functioning hybrid model with balanced performance in identifying phishing attempts while minimizing disruptions through false positives. However, there is room for improvement, especially in reducing misclassifications.

(5) 10-Fold Cross-Validation Results

To enhance evaluation reliability, we applied 10-fold cross-validation (**Table 5**). The dataset was randomly partitioned into 10 equal subsets; in each fold, 9 subsets were used for training and 1 for testing. This process was repeated 10 times, and the final performance metrics were computed as the mean ± standard deviation across the folds.

Table 5. 10-Fold Cross-Validation Results (Mean ± SD).

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Logistic Regression	0.772 ± 0.015	0.778 ± 0.018	0.781 ± 0.017	0.779 ± 0.016	0.842 ± 0.014
Random Forest	0.815 ± 0.012	0.823 ± 0.014	0.831 ± 0.013	0.824 ± 0.015	0.881 ± 0.011
Gradient Boosting	0.802 ± 0.014	0.812 ± 0.017	0.819 ± 0.016	0.814 ± 0.015	0.873 ± 0.013
Hybrid Model	0.842 ± 0.010	0.853 ± 0.012	0.861 ± 0.011	0.848 ± 0.012	0.912 ± 0.009

Figure 16 is the 10-fold cross-validation accuracy boxplot.

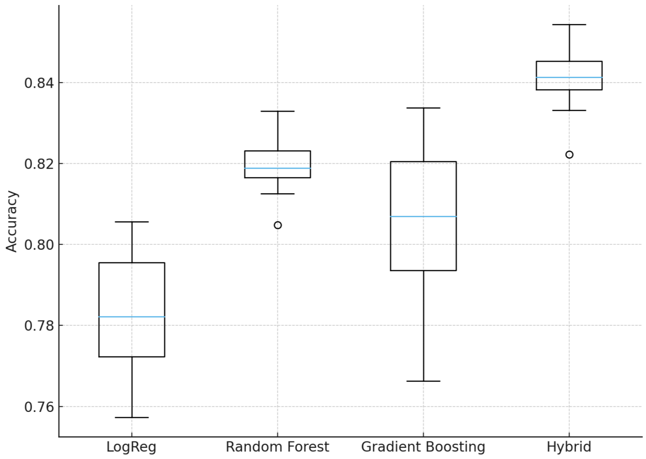


Figure 16. 10-fold Cross-Validation Accuracy Distribution.

(6) Comparative Analysis with State-of-the-Art Models

To contextualize the performance of the proposed hybrid model, we compared our results with benchmark performance values reported in existing literature for CNN, LSTM/GRU, and Transformer-based phishing detection models (**Table 6**). Although these deep-learning architectures generally achieve higher accuracy due to their ability to capture sequential and contextual information, they require significantly more computational resources and lack interpretability. In contrast, the proposed hybrid model maintains a balance of accuracy, lightweight implementation, and interpretability suitable for client-side phishing detection.

Table 6. Comparison of the Proposed Hybrid Model with Benchmarks Reported in Literature.

Model	Accuracy	F1-Score	Source
Proposed Hybrid (RF + GB + LR)	0.84	0.84	This study
CNN-based URL Classifier	0.83	0.82	Xiao et al. [34]
LSTM/GRU URL Sequence Model	0.97	0.97	Roy et al. [61]
Transformer-based URL Embedding Model (URLTran BERT)	0.997	0.997	Maneriker et al. [62]
Random Forest (baseline)	0.81	0.80	Alkawaz et al. [63]

4.4. Limitations

Although the proposed hybrid model demonstrates competitive performance, several limitations remain. First, the evaluation relies on a single publicly available dataset, which may limit generalizability to zero-day or highly obfuscated phishing URLs. Second, the method focuses exclusively on client-side URL and hyperlink features; server-side properties, website content, and DOM-level features were not incorporated. Third, the lightweight ML models used, while interpretable and efficient, may not capture complex sequential or contextual patterns as effectively as deep learning or transformer-based architectures.

5. Conclusions

Phishing has become a major cybersecurity challenge in today's fast-evolving digital world. With the widespread adoption of cashless payments, online business activities, and paperless systems, phishing threatens the integrity and trustworthiness of digital transactions. Increasingly sophisticated phishing attacks are eroding users' confidence in conducting secure financial and online operations. This study highlights the effectiveness of ML in countering phishing threats by analyzing key features and applying classification techniques. Utilizing ML's predictive power, data can be transformed into actionable insights to protect digital environments. The research centers on detecting phishing websites through a hybrid ML model that combines RF, GB, and LR algorithms. Developed in Python within the Visual Studio IDE, the model emphasizes affordability, scalability, and ease of access. By harnessing both linear and non-linear classifiers, the hybrid system improves detection accuracy. Experimental results reveal a high AUC, indicating a strong ability to distinguish phishing from legitimate URLs while balancing precision and recall effectively. These outcomes demonstrate that well-designed ML-based phishing detection systems offer adaptive, robust defenses against evolving cyber threats. The fusion of URL-based and hyperlink-based features creates a comprehensive framework that supports real-time phishing detection with valuable practical applications. While this study utilized a single Kaggle dataset, future research will expand evaluation using cross-domain datasets (e.g., UCI Phishing, PhishTank) and simulated zero-day attacks to enhance robustness and adaptability. Future work will also explore richer feature representations, experimenting with transformer-based URL embedding models, and assessing the system's resilience to adversarially crafted phishing attacks.

To further increase the proficiency, resilience, and adaptability of the proposed phishing detection system, the following recommendations are proposed:

i. Regular Monitoring and Evaluation: Continuously evaluate the system's performance against emerging phishing patterns. Regular assessments ensure that the model remains responsive to evolving attacker strategies and maintains high detection accuracy over time.

ii. Model Refinement: Enhancement of the hybrid model can be achieved through: Hyperparameter Tuning: Optimize the parameters of each classifier (RF, GB, and LR) to improve predictive accuracy and computational efficiency. Feature Engineering: Expand the feature space to include additional phishing indicators, such as content-based attributes or behavioral metrics, when sufficient data becomes available.

iii. Data Quality and Diversity: Ensure that the dataset used for model training is balanced and representative of both legitimate and phishing instances. To address class imbalance, data augmentation or synthetic sample generation techniques can be employed to enrich the dataset with varied phishing examples, improving model robustness.

iv. Integration and Deployment: Future work should explore the deployment of the model within web browsers, email gateways, and real-time filtering systems to provide active protection against phishing threats in practical use cases.

Author Contributions

Conceptualization, S.E.E. and J.O.U.; methodology, S.E.E. and U.O.; software, S.E.E.; validation, U.O., J.O.U. and O.O.F.; formal analysis, S.E.E. and U.O.; investigation, S.E.E. and U.O.; resources, U.O. and O.O.F.; data curation, S.E.E.; writing—original draft preparation, S.E.E. and J.O.U.; writing—review and editing, U.O. and O.O.F.; visualization, S.E.E. and U.O.; supervision, J.O.U.; project administration, U.O., and J.O.U. All authors have read and agreed to the published version of the manuscript.

Funding

This work received no external funding.

Institutional Review Board Statement

This study did not involve human participants, animals, or any personal data. Therefore, ethical approval from an Institutional Review Board was not required.

Informed Consent Statement

Not applicable.

Data Availability Statement

Data obtained from Kaggle (https://www.kaggle.com).

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Digital 2021: The Latest Insights Into the 'State of Digital'. Available online: https://wearesocial.com/uk/blog/2021/01/digital-2021-the-latest-insights-into-the-state-of-digital/ (accessed on 5 July 2021).
2. Debas, E.; Alhumam, N.; Riad, K. Unveiling the Dynamic Landscape of Malware Sandboxing: A Comprehensive Review. *Int. J. Adv. Comput. Sci. Appl.* **2024**, *15*, 1402–1416. [CrossRef]
3. Mahmoud, R. Redefining Malware Sandboxing: Enhancing Analysis Through Sysmon and ELK Integration. *IEEE Access* **2024**, *12*, 68624–68636. [CrossRef]
4. CISA. Phishing: What's in a Name? Available online: https://www.cisa.gov/news-events/news/phishing-whats-name (accessed on 19 September 2025).
5. CSI Today. Phishing/Scam Alert. Available online: https://csitoday.com/2024/05/phishing-scam-alert/ (accessed on 19 September 2025).
6. Gupta, B.B.; Tewari, A.; Jain, A.K.; et al. Fighting Against Phishing Attacks: State of the Art and Future Challenges. *Neural Comput. Appl.* **2017**, *28*, 3629–3654. [CrossRef]
7. Abdelhamid, N.A.; Ayesh, F.; Thabtah, F. Phishing Detection Based Associative Classification Data Mining. *Expert Syst. Appl.* **2014**, *41*, 5948–5959. [CrossRef]
8. APWG. Phishing Activity Trends Reports. Available online: https://apwg.org/trendsreports/ (accessed on 20 September 2025).
9. Sarker, I.H. Cyberlearning: Effectiveness Analysis of Machine Learning Security Modeling to Detect Cyber-Anomalies and Multi-Attacks. *Internet of Things* **2021**, *14*, 100393. [CrossRef]
10. Artashyan, A. The Number of Internet Users Worldwide Reaches 4.66 Billion. Available online: https://www.gizchina.com/featured/the-number-of-internet-users-worldwide-reaches-4-66-billion (accessed on 15 July 2025).
11. Jain, A.K.; Gupta, B.B. A Machine Learning Based Approach for Phishing Detection Using Hyperlinks Information. *J. Ambient Intell. Human Comput.* **2019**, *10*, 2015–2028. [CrossRef]
12. Rao, R.S.; Pais, A.R. Detection of Phishing Websites Using an Efficient Feature-Based Machine Learning Framework. *Neural Comput. Appl.* **2019**, *31*, 3851–3873. [CrossRef]
13. Internet Crime Complaint Center (IC3). Internet Crime Report 2021. Available online: https://www.ic3.gov/AnnualReport/Reports/2021_ic3report.pdf (accessed on 10 July 2025).
14. Sarker, I.H. Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN Comput. Sci.* **2021**, *2*, 420. [CrossRef]
15. Shi, Y.; Tian, Y.; Kou, G.; et al. *Optimization Based Data Mining: Theory and Applications*; Springer: London, UK, 2011. [CrossRef]
16. Iqbal, H.; Sarker, A.C.; Han, J.; et al. *Context-Aware Machine Learning and Mobile Data Analytics. Automated Rule-Based Services With Intelligent Decision Making*; Springer: Cham, Switzerland, 2021. [CrossRef]
17. Olson, D.L.; Shi, Y. *Introduction to Business Data Mining*; McGraw-Hill/Irwin: Boston, MA, USA, 2007.
18. Li, T.; Kou, G.; Peng, Y. Improving Malicious URLs Detection via Feature Engineering: Linear and Nonlinear Space Transformation Methods. *Inf. Syst.* **2020**, *91*, 10149419. [CrossRef]
19. Wardman, B.T.; Stallings, G.; Warner, A.; et al. High Performance Content-Based Phishing Attack Detection. In Proceedings of the 2011 eCrime Researchers Summit, San Diego, CA, USA, 7–9 November 2011; pp. 1–9. [CrossRef]
20. Chiew, K.L.; Chang, E.H.; Sze, S.N.; et al. Utilisation of Website Logo for Phishing Detection. *Comput. Secur.* **2015**, *54*, 16–26. [CrossRef]
21. Aydin, M.; Baykal, N. Feature Extraction and Classification Phishing Websites Based on URL. In Proceedings of the IEEE Conference on Communications and Network Security (CNS), Florence, Italy, 28–30 September 2015; pp. 769–770. [CrossRef]
22. Sheng, S.; Magnien, B.; Kumaraguru, P.; et al. Anti-Phishing Phil: The Design and Evaluation of a Game That Teaches People Not to Fall for Phish. In Proceedings of the 3rd Symposium on Usable Privacy and Security,

New York, NY, USA, 18–20 July 2007; pp. 88–99. [CrossRef]

23. Kumaraguru, P.; Sheng, S.; Acquisti, A.; et al. Teaching Johnny Not to Fall for Phish. *ACM Trans. Internet Technol.* **2010**, *10*, 1–31. [CrossRef]
24. Arachchilage, N.A.G.; Love, S. Security Awareness of Computer Users: A Phishing Threat Avoidance Perspective. *Comput. Hum. Behav.* **2014**, *38*, 304–312. [CrossRef]
25. Wang, X.; Zhang, R.; Yang, X.; et al. Voice Pharming Attack and the Trust of VoIP. In Proceedings of the 4th International Conference on Security and Privacy in Communication Networks, Istanbul, Turkey, 22–25 September 2008; pp. 1–11. [CrossRef]
26. Han, W.; Cao, Y.; Bertino, E.; et al. Using Automated Individual White-List to Protect Web Digital Identities. *Expert Syst. Appl.* **2012**, *39*, 11861–11869. [CrossRef]
27. Rosiello, A.P.E.; Kirda, E.; Kruegel, S.; et al. A Layout-Similarity-Based Approach for Detecting Phishing Pages. In Proceedings of the Third International Conference on Security and Privacy in Communications Networks and the Workshops - SecureComm, Nice, France, 17–21 September 2007; pp. 454–463. [CrossRef]
28. Felegyhazi, M.; Kreibich, C.; Paxson, V. On the Potential of Proactive Domain Blacklisting. In Proceedings of the 3rd USENIX conference on Large-Scale Exploits and emergent Threats: Botnets, Spyware, Worms, and More, Berkeley, CA, USA, 27 April 2010. [CrossRef]
29. Mohammad, R.M.; Thabtah, F.; McCluskey, L. Predicting Phishing Websites Based on Self-Structuring Neural Network. *Neural Comput. Appl.* **2014**, *25*, 443–458. [CrossRef]
30. Taeri, K.; Noseong, P.; Jiwon, H.; et al. Phishing URL Detection: A Network-Based Approach Robust to Evasion. In Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS ’22), Los Angeles, CA, USA, 7–11 November 2022; pp. 1769–178. [CrossRef]
31. Mao, J.; Tian, W.; Li, P.; et al. Phishing Website Detection Based on Effective CSS Features of Web Pages. In *Wireless Algorithms, Systems, and Applications*; Ma, L., Khreishah, A., Zhang, Y., et al., Eds.; Springer: Cham, Switzerland, 2017; 10251, pp. 1868–1878. [CrossRef]
32. Feng, F.; Zhou, Q.; Shen, Z.; et al. The Application of a Novel Neural Network in the Detection of Phishing Websites. *J. Ambient Intell. Human Comput.* **2024**, *15*, 1865–1879. [CrossRef]
33. Huang, Y.; Qin, J.; Wen, W. Phishing URL Detection via Capsule-Based Neural Network. In Proceedings of the 2019 IEEE 13th International Conference on Anti-Counterfeiting, Security, and Identification (ASID), Xiamen, China, 25–27 October 2019; pp. 22–26. [CrossRef]
34. Xiao, X.; Xiao, W.; Zhang, D.; et al. Phishing Websites Detection via CNN and Multi-Head Self-Attention on Imbalanced Datasets. *Comput. Secur.* **2021**, *108*, 102372. [CrossRef]
35. Aldakheel, E.A.; Zakariah, M.; Gashgari, G.; et al. A Deep Learning-Based Innovative Technique for Phishing Detection in Modern Security With Uniform Resource Locators. *Sensors* **2023**, *23*, 4403. [CrossRef]
36. Yasin, A.; Abuhasan, A. An Intelligent Classification Model for Phishing Email Detection. *Int. J. Network Secure Application* **2019**, *8*, 55–72. [CrossRef]
37. Rao, R.S.; Vaishnavi, T.; Pais, A.R. CatchPhish: Detection of Phishing Websites by Inspecting URLs. *J. Ambient Intell. Human Comput.* **2020**, *11*, 813–825. [CrossRef]
38. Babagoli, M.; Aghababa, M.P.; Solouk, V. Heuristic Nonlinear Regression Strategy for Detecting Phishing Websites. *Soft Comput.* **2019**, *23*, 4315–4327. [CrossRef]
39. Abedin, N.F.; Bawm, R.; Sarwar, T.; et al. Phishing Attack Detection Using Machine Learning Classification Techniques. In Proceedings of the 3rd International Conference on Intelligent Sustainable Systems (ICISS), Thoothukudi, India, 3–5 December 2020; pp. 1125–1130. [CrossRef]
40. Ronish, N.; Fahim, A.; Shahbaz, P. PhishEmailLLM: A Meta Model Approach to Detect Phishing Emails by Leveraging LLMs and Machine Learning Models. In Proceedings of the 2025 Australasian Computer Science Week (ACSW 2025), Brisbane, Australia, 10–13 February 2025; pp. 19–29. [CrossRef]
41. Do, N.Q.; Selamat, A.; Krejcar, O.; et al. Deep Learning for Phishing Detection: Taxonomy, Current Challenges and Future Directions. *IEEE Access* **2022**, *10*, 36429–36463. [CrossRef]
42. Feng, J.L.; Zou, O.; Ye, J.H. Web2Vesc: Phishing Webpage Detection Method Based on Multidimensional Features Driven by Deep Learning. *IEEE Access* **2020**, *8*, 221214–221224. [CrossRef]
43. Venugopal, S.; Panale, S.Y.; Agarwal, M.; et al. Detection of Malicious URLs Through an Ensemble of Machine Learning Techniques. In Proceedings of the 2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE), Brisbane, Australia, 8–10 December 2021; pp. 1–6. [CrossRef]
44. Aljofey, A.; Jiang, Q.; Rasool, A.; et al. An Effective Detection Approach for Phishing Websites Using URL and HTML Features. *Sci. Rep.* **2022**, *12*, 8842. [CrossRef]
45. Vecliuc, D.-D.; Artene, C.-G.; Tibeică, M.-N.; et al. An Experimental Study of Machine Learning for Phishing De-

tection. In *Intelligent Information and Database Systems*; Nguyen, N.T.; Chittayasothorn, S., Niyato, D., Trawiński, B., Eds.; Springer, Cham, 2021; 12672, pp. 427–439. [CrossRef]

46. Opara, C.; Chen, Y.; Wei, B. Look Before You Leap: Detecting Phishing Web Pages by Exploiting Raw URL and HTML Characteristics. *Expert Syst. Appl.* **2023**, *236*, 121183. [CrossRef]
47. Lin, Y.; Liu, R.; Divakaran, D.M.; et al. Phishpedia: A Hybrid Deep Learning Based Approach to Visually Identify Phishing Webpages. In Proceedings of the 30th USENIX Security Symposium, Online, 11–13 August 2021. Available online: https://www.usenix.org/system/files/sec21fall-lin.pdf
48. Hong, J.; Kim, T.; Liu, J.; et al. Phishing URL Detection With Lexical Features and Blacklisted Domains. In *Adaptive Autonomous Secure Cyber Systems*; Jajodia, S., Cybenko, G., Subrahmanian, V., et al., Eds.; Springer: Cham, Switzerland, 2020; pp. 253–267. [CrossRef]
49. Sahoo, D.; Liu, C.; Hoi, S. Malicious URL Detection Using Machine Learning: A Survey. *arXiv preprint* **2022**, *arXiv:1701.07179*. [CrossRef]
50. Ravindu, D.-S.; Nabeel, M.; Elvitigala, C.; et al. Compromised or Attacker-Owned: A Large Scale Classification and Study of Hosting Domains of Malicious URLs. In Proceedings of the 30th USENIX Security Symposium, Online, 11–13 August 2021. Available online: https://www.usenix.org/conference/usenixsecurity21/presentation/desilva
51. Chen, Z.; Wu, L.; Hu, Y.; et al. Lifting the Grey Curtain: Analyzing the Ecosystem of Android Scam Apps. *IEEE TDSC* **2023**, *21*, 3406–3421. [CrossRef]
52. Hong, G.Z.; Yang, S.; Yang, X.; et al. Analyzing Ground-Truth Data of Mobile Gambling Scams. In Proceedings of the 2022 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 22–26 May 2022. [CrossRef]
53. Sharma, A. More Than 200 Cryptomining Packages Flood npm and PyPI Registry. Available online: https://blog.sonatype.com/more-than-200-cryptominers-flood-npm-and-pypi-registry (accessed on 1 May 2023).
54. Sun, X.; Gao, X.; Cao, S.; et al. 1+1>2: Integrating Deep Code Behaviors with Metadata Features for Malicious PyPI Package Detection. In Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering, Sacramento, CA, USA, 27 October 2024; pp. 1159–1170. [CrossRef]
55. Guo, W.; Xu, Z.; Liu, C.; et al. An Empirical Study of Malicious Code in PyPI Ecosystem. In Proceedings of the 38th IEEE/ACM International Conference on Automated Software Engineering, Echternach, Luxembourg, 11–15 November 2023; pp. 166–177.
56. Maci, A.; Santorsola, A.; Coscia, A.; et al. Unbalanced Web Phishing Classification Through Deep Reinforcement Learning. *Computers* **2023**, *12*, 118. [CrossRef]
57. Samet. What is Virus Total? Available online: https://medium.com/@sametyorulmaz777/what-is-virus-total-70c64b7c5e95 (accessed on 10 November 2024).
58. VirusTotal. VT Intelligence. Available online: https://www.virustotal.com/gui/intelligence-overview (accessed on 10 November 2024).
59. VirusTotal. VirusTotal. Available online: https://www.virustotal.com/gui/home/upload (accessed on 17 February 2025).
60. Onwudebelu, U.; Ugah, J.O.; Eze, S.E.; et al. Developing a Security-Driven Hybrid Model for Detecting Malicious URLs. In Proceedings of the 2025 Conference of the Society for the Advancement of ICT & Comparative Knowledge (SOCTHADICKconf'25), Ibadan, Nigeria, 16–19 November 2025.
61. Roy, S.S.; Awad, A.I.; Amare, L.A.; et al. Multimodel Phishing URL Detection Using LSTM, Bidirectional LSTM, and GRU Models. *Future Internet* **2022**, *14*, 340. [CrossRef]
62. Maneriker, P.; Stokes, J.W.; Lazo, E.G.; et al. URLTran: Improving Phishing URL Detection Using Transformers. In Proceedings of the MILCOM 2021–2021 IEEE Military Communications Conference (MILCOM), San Diego, CA, USA, 29 November–2 December 2021; pp. 197–204. [CrossRef]
63. Alkawaz, M.H.; Steven, S.J.; Hajamydeen, A.I. Detecting Phishing Website Using Machine Learning. In Proceedings of the 2020 16th IEEE International Colloquium on Signal Processing & Its Applications (CSPA), Langkawi, Malaysia, 28–29 February 2020; pp. 111–114. [CrossRef]

Publisher's Note: The views, opinions, and information presented in all publications are the sole responsibility of the respective authors and contributors, and do not necessarily reflect the views of UK Scientific Publishing Limited and/or its editors. UK Scientific Publishing Limited and/or its editors hereby disclaim any liability for any harm or damage to individuals or property arising from the implementation of ideas, methods, instructions, or products mentioned in the content.