International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(1s) 2026 pp. 1-14
https://doi.org/10.70917/ijcisim-2026-2010
Article

Intelligent Browser Extension for Real-Time Phishing Detection Using Hybrid Machine Learning Models

Aishwarya S. Sanap[1], Nilesh R. Wankhade[1], Sahebrao B. Bagal[1], Vidya B. Kale[1], Archana S. Kolhe[1], Tushar Jadhav[2]

[1] Department of Computer Engineering, Late G. N. Sapkal College of Engineering, Nashik, Maharashtra, India.
Emails:
aishwaryasanap18@gmail.com
hodcomp.lgnscoe@sapkalknowledgehub.org
principal.lgnscoe@sapkalknowledgehub.org
vidyakale005@gmail.com
archanakolhe59@gmail.com
[2] Department of Electronics and Telecommunication Engineering (E&TC), Vishwakarma Institute of Technology (VIT), Pune – 411037, Maharashtra, India. tushar.jadhav@vit.edu

Abstract: —Phishing attacks remain a significant cyber security threat due to their evolving nature and reliance on social engineering techniques. While machine learning-based phishing detection models have demonstrated high accuracy in offline evaluations, their real-time deployment in client-side environ- ments remains challenging. This paper presents a hybrid phishing detection framework that integrates offline machine learning analysis with real-time browser-based deployment. Multiple machine learning classifiers are evaluated using URL- based features, and the Random Forest model is identified as the most effective classifier. Feature importance analysis is employed to extract the most influential phishing indicators, which are subsequently translated into a lightweight rule-weighted detec- tion mechanism. This mechanism is implemented as a browser extension to enable real-time phishing detection without relying on external servers. Experimental results demonstrate that the proposed approach achieves high detection accuracy while maintaining low com- putational overhead. The system provides explainable detection decisions, preserves user privacy, and effectively bridges the gap between machine learning research and practical phishing defense systems suitable for real-world deployment.

Keywords: — Phishing detection, Machine learning, URL- based analysis, Random Forest, Feature importance, Browser extension, Real-time detection, Cyber security

1. Introduction

The rapid expansion of internet-based services such as online banking, e-commerce, cloud platforms, and social net- working has significantly increased users' exposure to cyber threats. Among these threats, phishing attacks remain one of the most prevalent and damaging attack vectors, exploiting social engineering techniques to deceive users into revealing sensitive information. According to the Verizon Data Breach Investigations Report 2024, phishing continues to be a leading cause of global data breaches, highlighting the urgent need for robust and adaptive phishing detection mechanisms [1], [2].

Traditional phishing detection techniques primarily rely on blacklist-based and rule-based mechanisms. While these methods are computationally efficient and easy to deploy, they suffer from several inherent limitations, including delayed blacklist updates, inability to detect zero-day phishing attacks, and poor adaptability to rapidly evolving phishing strategies [3], [4]. As attackers continuously register new domains and modify URL structures, static defense techniques become increasingly ineffective. To overcome these limitations, machine learning (ML)

based phishing detection approaches have gained significant research attention. ML models are capable of learning dis- criminative patterns from large-scale datasets and generalizing to unseen phishing URLs. Several studies have demonstrated the effectiveness of machine learning algorithms such as Decision Trees, Support Vector Machines, Random Forests, and ensemble classifiers for phishing detection using URL- based features [5]–[8]. URL-based approaches are particularly suitable for real-time detection due to their low computational overhead and independence from webpage content loading.

Recent works have explored deep learning and hybrid approaches to further enhance phishing detection performance. Attention-based neural networks, hybrid deep learning ar- chitectures, and feature fusion techniques have demonstrated improved detection accuracy and robustness [9], [10], [12], [13]. Furthermore, large-scale surveys highlight the growing role of machine learning and artificial intelligence in modern cyber security systems [11], [14]. Deep learning–based phishing detection systems incorporat- ing URL, textual, and visual features have shown promising results [17]–[19]. However, these approaches often require substantial computational resources and are predominantly evaluated in offline or server-based environments, limiting their suitability for lightweight real-time deployment. Client-side phishing detection has emerged as a promis- ing direction for immediate user protection. Browser secu- rity architectures and lightweight client-side detection models have demonstrated the feasibility of real-time phishing detec- tion during browsing [20], [21]. However, integrating high- accuracy machine learning models into resource-constrained browser environments while preserving privacy remains a challenging task [16].In this work, we propose an intelligent phishing detection framework that combines offline machine learning analysis with real-time browser extension deployment. Multiple ma- chine learning models are trained and evaluated using URL- based features extracted from publicly available phishing datasets [26], [27]. Feature importance analysis using Random Forest is employed to identify the most influential phishing indicators. These ML-derived insights are then translated into a lightweight rule-weighted detection mechanism implemented as a browser extension, enabling real-time phishing detection without relying on external servers.

The main contributions of this paper are summarized as follows:

- A comparative evaluation of multiple machine learning models for phishing website detection using URL-based features.
- An explainable feature importance analysis to identify key phishing indicators influencing detection performance.
- A real-time client-side phishing detection system imple- mented as a browser extension, bridging the gap between offline ML models and practical deployment.
- Experimental validation demonstrating high detection ac- curacy with minimal computational overhead

2. Related Work

Phishing detection has attracted significant research interest due to the rapid growth of internet usage and the increasing sophistication of social engineering attacks. Early phishing detection mechanisms primarily relied on blacklist-based ap- proaches, where known phishing URLs are stored and blocked. While blacklist techniques are simple and efficient, they are inherently reactive and fail to detect newly generated or zero- day phishing attacks [3], [4]. These limitations motivated the development of intelligent detection mechanisms capable of identifying phishing websites based on their intrinsic charac- teristics. Machine learning-based phishing detection methods have been extensively studied to address the shortcomings of traditional approaches. URL-based features such as lexical structure, domain properties, and host-based attributes have been shown to be strong indicators of phishing behavior. Sahingoz et al. [5] proposed a machine learning framework using lexical URL features and demonstrated strong detection performance using ensemble classifiers. Similarly, Chiew et al. [6] Confirmed the effectiveness of structural website features for phishing detection. Abdelhamid et al. [7] introduced an associative classification approach, while Zhang et al. [8] demonstrated the effectiveness of multi-feature fusion and ensemble learning. These studies highlight the robustness and generalization capability of classical machine learning techniques for phishing detection. Recent research has further explored hybrid and advanced machine learning approaches. Adebowale et al. [9] combined Random Forest and neural networks to improve detection performance, while Niu and Zhang [10] proposed URL repre- sentation learning techniques for phishing detection. Attention- based deep neural networks have also been applied success- fully to improve detection accuracy and robustness [12], [13]. Additionally, comprehensive surveys emphasize the growing role of machine learning and artificial intelligence in modern cyber security systems [11], [14].

With the advancement of deep learning, researchers have explored neural network-based approaches to further enhance phishing detection performance. Xiang et al. [17] developed a layered hybrid model incorporating URL, textual, and image features. Alkhalil et al. [18] introduced a real-time phishing detection system based on deep learning techniques, while Bhat [19] proposed a convolutional neural network framework for phishing detection. Although these approaches achieve high detection accuracy, they often require substantial compu- tational resources and are predominantly evaluated in offline or server-based environments, limiting their suitability for lightweight real-time deployment. Hybrid approaches combining multiple feature sets and classifiers have also been proposed. Sharma [24] demonstrated that hybrid machine learning models outperform individual classifiers for phishing website detection. Sanap and Patil [25] emphasized the importance of hybrid detection strategies for enhancing phishing awareness and detection accuracy. While these approaches improve performance, their deployment com- plexity remains a challenge in client-side environments.

Client-side phishing detection using browser extensions has emerged as a promising solution for real-time user protection. The security architecture of modern browsers and lightweight client-side detection models demonstrate the feasibility of real- time phishing detection during browsing [20], [21]. Kumar [16] Proposed a client-side machine learning-based defense mechanism to detect phishing attacks during browsing. How- ever, integrating high-accuracy machine learning models into resource-constrained browser environments while preserving privacy remains a challenging task. In contrast to existing work, the present study bridges the gap between offline machine learning analysis and real-time deployment by leveraging feature importance derived from Random Forest models. Instead of executing the complete machine learning model within the browser, the most dis- criminative phishing features are translated into a lightweight, rule-weighted detection mechanism implemented as a browser extension. This design enables efficient, explainable, and real- time phishing detection while maintaining high detection ac- curacy and low computational overhead

Table 1: Comparison of Existing Phishing Detection Approaches

Study	**Method**	**Real-Time**	**Deployment**
Ma et al. [3]	ML (URL-based)	No	Offline
Sahingoz et al. [5]	Ensemble ML	No	Offline
Bhat [19]	Deep Learning	No	Server-based
Kumar [16]	ML	Yes	Browser Extension
Proposed Work	**ML + Rules**	**Yes**	**Browser Extension**

Table 1 highlights that most existing approaches focus on offline or server-based detection, whereas the proposed work emphasizes real-time client-side deployment.3

2.1 Research Gap

The existing body of research demonstrates that machine learning and deep learning models can achieve high phishing detection accuracy. However, most studies focus on offline evaluation or server-based detection frameworks [3], [5], [19]. Such systems often introduce high computational require- ments, increased latency, and dependency on backend servers, making them unsuitable for real-time deployment. Deep learning-based phishing detection systems further introduce computational overhead and reduced transparency, making them difficult to deploy in resource-constrained envi- ronments such as web browsers [17], [18]. Although client- side browser extension approaches have been explored [16], these solutions often execute full machine learning models or rely on frequent server communication, raising concerns related to efficiency, scalability, and user privacy.

Another key limitation is the insufficient use of model explain ability. While many works report high accuracy, they do not explicitly analyze feature importance or translate learned knowledge into practical real-time systems. Consequently, a clear gap exists between high-accuracy offline machine learn- ing models and lightweight, explainable, real-time phishing detection systems suitable for end-user deployment.

2.2 Problem Statement

Despite the availability of accurate machine learning models for phishing detection, practical systems capable of real-time deployment in browser environments remain limited. Existing solutions suffer from high computational overhead, delayed detection, or reliance on server-side processing. Therefore, the problem addressed in this work is to design an efficient phishing detection system that integrates machine learning intelligence into a lightweight real-time client-side deployment mechanism without compromising detection accuracy or user privacy.

2.3 Motivation

The increasing prevalence of phishing attacks and the limi- tations of existing detection mechanisms highlight the need for practical and efficient solutions. By leveraging feature importance analysis from ensemble machine learning models, it becomes possible to translate learned knowledge into a lightweight and explainable detection mechanism suitable for browser environments. This work is motivated by the need to bridge the gap between offline machine learning research and practical, user-centric phishing defense systems.

3. Proposed Methodology

The proposed phishing detection framework integrates of- fline machine learning analysis with real-time client-side de- ployment. The methodology is designed to leverage the high detection accuracy of machine learning models while ensuring efficient and lightweight real-time execution within a browser environment. The overall workflow of the proposed system consists of four major phases: data preprocessing, machine learning model training and evaluation, feature importance analysis, and real-time browser extension deployment.

3.1 System Overview

The overall architecture of the proposed system is illustrated conceptually as a two-layer framework. In the first layer, machine learning models are trained offline using a labeled phishing dataset to learn discriminative URL-based phishing patterns. In the second layer, the learned knowledge is opera- tionalized in a browser extension that performs real-time URL Analysis and alerts users upon detection of potential phishing websites. This separation enables computationally intensive learning to be performed offline, while lightweight detection logic is executed during browsing. The architecture of the

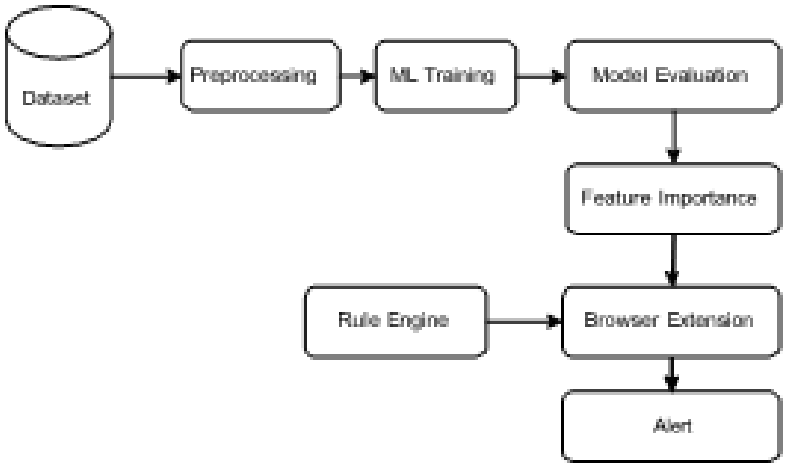

Fig. 1: Architecture of the proposed ML-based real-time phishing detection system

Proposed phishing detection system is designed as a two- layer framework that separates offline machine learning anal- ysis from online real-time detection. This separation ensures high detection accuracy while maintaining low computational overhead during browsing. In the offline learning layer, a phishing URL dataset consist- ing of both legitimate and malicious URLs is first subjected to data preprocessing and feature engineering. This step in- volves cleaning the dataset, removing redundant attributes, and preparing URL-based features that capture lexical, structural, and host-related characteristics. The processed dataset is then used to train multiple machine learning classifiers, including Logistic Regression, Decision Tree, Support Vector Machine, K-Nearest Neighbor, and Random Forest.

Following model training, the performance of each classifier is evaluated using standard metrics such as accuracy, precision, recall, and F1-score. Based on this comparative evaluation, the Random Forest model is selected as the most effective classifier due to its superior performance and robustness. An important advantage of Random Forest is its ability to provide feature importance scores, which indicate the relative contribution of each URL feature to phishing detection. The extracted feature importance serves as a bridge between offline learning and online deployment. Instead of executing the complete machine learning model within the browser, the most influential phishing indicators identified during training are mapped into a lightweight rule-based scoring mechanism. Each rule is assigned a weight proportional to the learned importance of the corresponding feature, enabling efficient and interpretable decision-making. In the online detection layer, the rule-based scoring mechanism is implemented as a client-side browser extension. The extension continuously monitors the URL of the currently visited webpage and performs real-time feature extraction. Based on the weighted scoring scheme, a cumulative phishing score is computed. If this score exceeds a predefined threshold, the extension immediately alerts the user about the potential phishing threat.

This architecture effectively combines the learning capabil- ity of machine learning models with the efficiency of client- side deployment. By decoupling model training from real- time execution, the proposed system achieves fast detection, preserves user privacy by avoiding server communication, and provides explainable phishing warnings suitable for practical use in everyday browsing environments.

3.2 Dataset and Preprocessing

The experiments are conducted using a publicly available phishing website dataset comprising 11,055 URL instances. After preprocessing, each URL is represented using 30 URL- based features that capture lexical, structural, and host-based characteristics commonly associated with phishing behavior. These features include SSL certificate status, presence of IP addresses in URLs, abnormal URL patterns, domain registra- tion properties, and hyperlink-related attributes.

Table 2: Dataset Overview

Parameter	**Value**
Total number of URLs	11,055
Number of features	30
Classes	Phishing, Legitimate
Label encoding	1 = Phishing, 0 = Legitimate
Train–test split	80% – 20%
Missing values	None

Prior to model training, data preprocessing is performed to enhance data quality and consistency. This process includes the removal of redundant attributes, handling of missing or inconsistent values, and preparation of feature representations suitable for machine learning models. Following preprocess- ing, the dataset is partitioned into training and testing subsets using an 80:20 split to ensure an unbiased and reliable evaluation of model performance. The selected features, as summarized in Table III, capture key lexical, structural, and security-related characteristics of URLs that are commonly exploited in phishing attacks. Lexical features analyze the composition and length of URLs to iden- tify obfuscation techniques, while structural features examine domain hierarchy, redirection patterns, and the presence of suspicious symbols. Security-related features, such as SSL certificate validity and domain registration properties, provide additional indicators of website trustworthiness. Collectively, these features enable efficient and accurate phishing detection without requiring webpage content loading, making the ap- proach suitable for real-time and client-side deployment.

3.3 Machine Learning Model Training

To evaluate the effectiveness of different learning paradigms, multiple machine learning classifiers are trained and compared, including Logistic Regression, Decision Tree, Support Vector Machine, K-Nearest Neighbour, and Random Forest. These models are selected due to their widespread use in phishing detection and their complementary strengths in handling linear and non-linear feature relationships.

Table 3: Description of Selected URL-Based Phishing Features

Feature	Description
Having IP Address	Checks whether an IP address is used instead of a domain name in the URL, which is a strong indicator of phishing activity.
URL Length	Determines whether the URL length exceeds a predefined threshold, as excessively long URLs are often used to conceal malicious content.
Shortening Service	Detects the use of URL shortening services that may redirect users to hidden or malicious phishing destinations.
Having "@" Symbol	Identifies the presence of the "@" symbol in the URL, which can mislead users about the actual destination address.
Double Slash Redirecting	Detects abnormal redirection patterns caused by multiple forward slashes ("//") within the URL path.
Prefix–Suffix (-)	Checks for hyphens in domain names, commonly used to imitate legitimate websites and deceive users.
Having Subdomain	Analyzes the number of subdomains to identify suspicious or deceptive domain structures.
SSL Final State	Evaluates the validity and trustworthiness of the website's SSL certificate to assess secure communication.
Domain Registration Length	Measures domain registration duration, as phishing domains typically have short lifespans.
Request URL	Examines whether external resources (e.g., images, scripts) are loaded from domains different from the primary URL.
URL of Anchor	Analyzes anchor tags to determine whether hyperlinks point to suspicious or unrelated domains.
Links in Tags	Checks whether links embedded in HTML tags reference external, abnormal, or potentially malicious domains.

Each model is trained using the training subset, and per- formance is evaluated on the test subset using standard clas- sification metrics such as accuracy, precision, recall, and F1- score. This comparative analysis enables the identification of the most suitable model for phishing detection based on both predictive performance and robustness.

3.4 Model Selection and Feature Importance Analysis

Based on experimental evaluation, the Random Forest clas- sifier demonstrates superior performance compared to other models, achieving the highest accuracy and F1-score. In addi- tion to its strong predictive capability, Random Forest provides an inherent mechanism for feature importance estimation, which is critical for model interpretability. Feature importance scores are extracted from the trained Random Forest model to identify the most influential URL features contributing to phishing detection. High-importance features such as SSL certificate state, anchor URL behavior, subdomain usage, and domain registration length are identified as key phishing indicators. This analysis not only enhances transparency but also serves as a bridge between offline learning and real-time deployment.

3.5 Translation of ML Knowledge to Real-Time Detection

Executing a full machine learning model within a browser environment is computationally expensive and impractical for real-time usage. Therefore, the proposed framework translates the feature importance knowledge derived from the Random Forest model into a lightweight, rule-weighted scoring mech- anism. Each high-

importance feature is mapped to a corresponding heuristic rule with an assigned weight proportional to its learned importance. Medium- and low-importance features are assigned lower weights. During browsing, the extension extracts relevant URL features, computes a cumulative phishing score based on these weights, and determine whether the visited website exhibits phishing characteristics.

3.6 Algorithmic Workflow

The proposed algorithmic workflow describes a machine learning–driven framework for real-time phishing detection integrated into a browser extension. Initially, phishing and legitimate URL datasets are collected and preprocessed to ensure data consistency and quality. Multiple machine learning models are then trained and evaluated using standard classification metrics such as accuracy, precision, recall, and F1-score. The best-performing model is selected based on these evaluation results. From this model, feature importance scores are extracted to determine which URL-based attributes contribute most significantly to phishing detection. These important features are mapped into weighted detection rules, forming the basis of a lightweight scoring mechanism suitable for real-time deployment. During execution, the system extracts predefined URL features from an input URL and initializes a phishing suspicion score. For each extracted feature, if phishing-related behavior is detected, the corresponding weight is added to the cumulative score. The final score is compared against a predefined threshold. If the score exceeds the threshold, a phishing warning is displayed to the user; otherwise, normal browsing is allowed. This structured approach ensures explainability, efficiency, and suitability for real-time browser-level security applications.

Algorithm 1: ML-Weighted Phishing Detection

Input: URL u

Output: Warning or Safe

1: F ← ExtractFeatures(u)

2: S ← 0

3: for each feature fi in F do

4: S ← S + wi * fi

5: end for

6: if S ≥ T then

7: return "Phishing Warning"

8: else

9: return "Safe"

10: end if

4. Mathematical Modeling

Let an input URL be denoted as: The URL is transformed into a feature vector representation:

$$F(u) = [f1, f2, f3, \dots, fn] \qquad (1)$$

where each fi represents a binary or numerical URL-based feature (e.g., IP address usage, URL length, SSL state, subdomain count, etc.). Each feature is assigned a weight wi derived from the trained

Random Forest model based on feature importance:

$$W = [w1, w2, w3, \dots, wn]$$

The phishing suspicion score for a given URL is computed as:

$$S(u) = \Sigma\,(wi * I(fi)),\ i = 1\ to\ n \qquad (2)$$

where I(fi) is an indicator function defined as:

$$I(fi) = 1, \quad if\ feature\ fi\ indicates\ phishing\ behavior$$

$$I(fi) = 0,\ otherwise \qquad (3)$$

Thus, only phishing-indicative features contribute to the score. The final classification decision function is:

$$D(u) =$$

$$Phishing, \quad if\ S(u) \geq T$$

$$Legitimate,\ otherwise \qquad (4)$$

where T is a predefined threshold selected empirically to balance detection accuracy and false positive rate. In summary, the model operates as a weighted linear scoring system where the overall phishing risk of a URL is quantified by combining feature importance weights and their corresponding phishing indicators. If the cumulative weighted score exceeds the threshold T, the URL is classified as phishing; otherwise, it is considered safe.

where T is an empirically selected threshold that balances detection accuracy and false positive rate. The mathemati- cal formulation models the phishing detection process as a weighted feature-based decision system. Each visited URL is represented as a feature vector comprising URL-derived attributes that capture lexical, structural, and security-related characteristics. The assigned weights reflect the relative im- portance of each feature, as learned from the Random Forest model during offline training. The phishing score is computed as a linear combination of feature weights and their corresponding indicator values, allowing the system to quantify the overall phishing risk associated with a URL. The indicator function ensures that only features exhibiting phishing-related behavior contribute to the final score, thereby reducing noise from benign attributes. The final classification decision is determined by comparing the computed phishing score against a predefined threshold. If the score exceeds this threshold, the URL is classified as phishing; otherwise, it is considered legitimate.

5. Experimental Setup

The experimental setup outlines the systematic procedure used to validate the effectiveness of the proposed phishing detection framework. It ensures that the evaluation is unbiased, reproducible, and aligned with standard machine learning validation practices. The section explains the dataset characteristics, feature extraction strategy, data partitioning method, and classification models used for performance benchmarking.

5.1. Dataset Description

The experiments were conducted using publicly available phishing datasets comprising legitimate and phishing URLs. After preprocessing, the final dataset consisted of 11,055 URL instances described using 30 URL-based features. These features capture lexical, structural, and host-based characteristics such as the presence of SSL certificates, IP addresses within URLs, abnormal URL structures, domain registration information, and hyperlink-related attributes. Each URL instance was labeled as either phishing or legitimate, creating a binary classification problem. The dataset was carefully cleaned to remove duplicate entries and inconsistent records, ensuring high-quality training and evaluation data.

5.2 Feature Set

The feature set includes 30 engineered URL-based attributes designed to detect phishing indicators without requiring webpage content loading. These features identify suspicious patterns such as excessive URL length, unusual symbols, subdomain depth, domain age, SSL validity, and redirection behavior. By relying solely on URL-derived attributes, the system enables efficient real-time detection while minimizing computational overhead. This lightweight feature design supports browser-level implementation and ensures fast decision-making without requiring page rendering.

5.3 Training and Testing Strategy

To ensure unbiased performance evaluation, the dataset was divided into training and testing subsets using an 80:20 split. The training set was used exclusively for model learning, while the testing set remained unseen during training to evaluate generalization performance. This separation ensures that the reported results reflect real-world predictive capability rather than overfitting. The evaluation metrics include accuracy, precision, recall, and F1-score.

5.4 Machine Learning Models

Five classical supervised machine learning classifiers were implemented: Logistic Regression (LR), Decision Tree (DT), Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Random Forest (RF). These models were selected due to their widespread application in phishing detection research and their ability to handle binary classification problems effectively. Default hyperparameters were used unless otherwise specified to maintain fairness and reproducibility across model comparisons.

5.5 Evaluation Metrics

The performance of the proposed phishing detection system is evaluated using four widely accepted classification metrics: accuracy, precision, recall, and F1-score. These metrics provide a comprehensive assessment of the model's predictive capability, especially in cybersecurity applications where class imbalance is common.

Accuracy measures the overall proportion of correctly classified instances among all predictions. While it provides a general sense of performance, it may not fully reflect effectiveness when phishing samples are fewer than legitimate ones. Precision evaluates how many URLs predicted as phishing are actually phishing. High precision indicates a low false positive rate, which is important to avoid unnecessary user alerts.

Recall, also known as detection rate or sensitivity, measures the proportion of actual phishing URLs that are correctly identified. High recall is critical in security systems, as missing phishing instances (false negatives) can expose users to risk.

F1-score represents the harmonic mean of precision and recall, offering a balanced performance measure when both false positives and false negatives are important. Together, these metrics ensure a reliable and fair evaluation of the proposed system.

5.6 Implementation Environment

The implementation environment describes the technical framework used to develop and test the phishing detection system. All machine learning experiments were implemented using Python, leveraging libraries such as Scikit-learn for model training, NumPy and Pandas for data preprocessing, and Matplotlib for visualization. The real-time detection mechanism was integrated into a Google Chrome browser extension using JavaScript. This architecture separates computationally intensive model training (performed offline) from lightweight detection logic executed during browsing. Such a design ensures efficient real-time performance while maintaining strong predictive accuracy.

6. Result And Discussion

This section presents the experimental results of the pro- posed phishing detection framework, including quantitative performance evaluation of machine learning models and qualitative validation of the real-time browser extension deployment.

Table 4: Performance Comparison of Machine Learning Models

Model	**Accuracy**	**Precision**	**Recall**	**F1-score**
Random Forest	**0.976481**	**0.980331**	**0.966327**	**0.973279**
Decision Tree	0.969697	0.976017	0.955102	0.965446
SVM	0.949796	0.958817	0.926531	0.942398
KNN	0.948440	0.951983	0.930612	0.941176

Logistic Regression	0.927635	0.934322	0.900000	0.916840

Multiple machine learning classifiers were evaluated to assess their effectiveness in detecting phishing websites. The evaluated models include Logistic Regression (LR), Decision Tree (DT), Support Vector Machine (SVM), K-Nearest Neigh- bour (KNN), and Random Forest (RF). Table 4 summarizes the comparative performance results in terms of accuracy, precision, recall, and F1-score. The results show that the Random Forest classifier out-performs all other models across all evaluation metrics. The superior performance of Random Forest can be attributed to its ensemble learning capability, which combines multiple decision trees to reduce over fitting and improve generalization.

A detailed classification report of the Random Forest model is presented in Table VI. The model achieved an overall accuracy of 98% on the test dataset. For the legitimate class, the model achieved a precision of 0.97 and recall of 0.98, indicating minimal false alarms. For the phishing class, the model achieved a precision of 0.98 and recall of 0.97, demon- starting strong capability in detecting malicious websites while maintaining a low false negative rate.

Table 5: Random Forest Classification Report

Class	**Precision**	**Recall**	**F1-score**	**Support**
Legitimate (0)	0.97	0.98	0.98	1231
Phishing (1)	0.98	0.97	0.97	980
Accuracy	0.98 (2211 samples)			
Macro Avg	0.98	0.98	0.98	2211
Weighted Avg	0.98	0.98	0.98	2211

The macro and weighted averages further confirm the bal- anced performance of the model across both classes, demon- strating the robustness of the Random Forest classifier for real- world phishing detection.

A detailed classification report of the Random Forest model is presented in Table 6. The model achieved an overall accuracy of 98% on the test dataset, indicating strong gen- eralization capability. For the legitimate class, the model achieved a precision of 0.97 and recall of 0.98, showing that legitimate websites are correctly identified with minimal false alarms. For the phishing class, the model achieved a precision of 0.98 and recall of 0.97, demonstrating strong effectiveness in detecting malicious websites while maintaining a low false negative rate. The macro and weighted averages further confirm the bal- anced performance of the classifier across both classes. These results highlight the robustness and reliability of the Random Forest model for real-world phishing detection scenarios.

Table 6: Random Forest Classification Report

Class	**Precision**	**Recall**	**F1-score**	**Support**
Legitimate (0)	0.97	0.98	0.98	1231
Phishing (1)	0.98	0.97	0.97	980
Accuracy	0.98 (2211 samples)			
Macro Avg	0.98	0.98	0.98	2211
Weighted Avg	0.98	0.98	0.98	2211

The confusion matrix of the Random Forest classifier is shown in Fig. 2. The results indicate that the majority of phishing and legitimate websites are correctly classified. Only a small number of misclassifications are observed, highlighting the reliability of the proposed approach

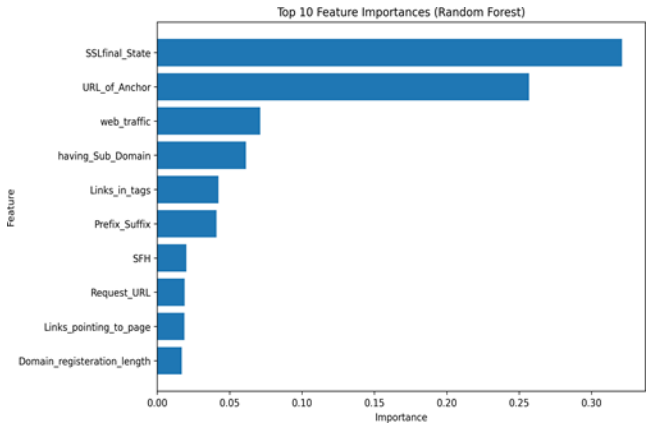

Fig. 2: Confusion matrix of the Random Forest classifier

Feature importance analysis was conducted using the Ran- dom Forest model to identify the most influential URL-based phishing indicators. Fig. 3 illustrates the top contributing features.

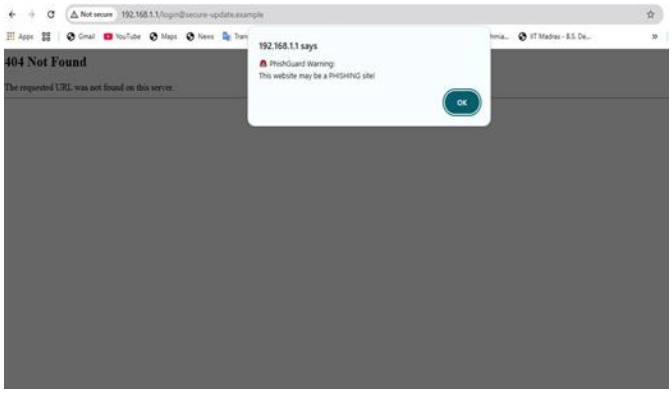

Fig. 3: Top feature importance derived from the Random Forest model

The results show that SSL certificate state, URL anchor behavior, web traffic rank, and subdomain structure play a significant role in phishing detection. This analysis provides explainability and serves as a basis for translating machine learning knowledge into lightweight real-time detection rules. To validate the practical applicability of the proposed frame- work, a Google Chrome browser extension named PhishGuard was developed. The extension implements the ML-derived rule-weighted detection mechanism for real-time URL analysis. Fig. 4 shows the successful installation of the extension in the Chrome browser environment.

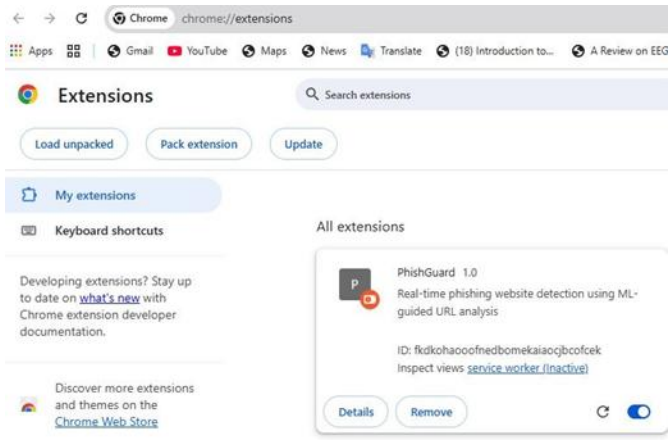

Fig. 4: PhishGuard browser extension loaded in Chrome

The effectiveness of the extension was evaluated by testing both legitimate and phishing-like URLs. Fig. 5 illustrates normal browsing behavior on a legitimate website, where no warning is triggered.

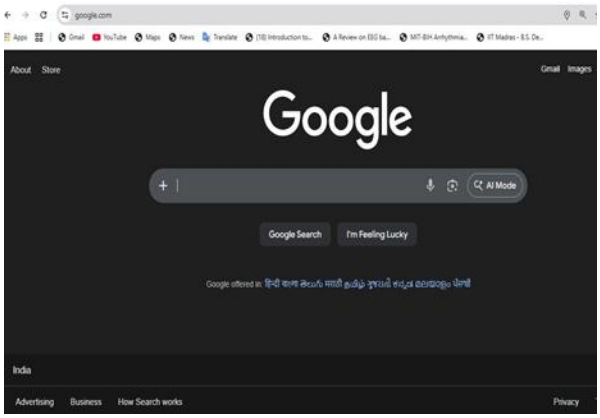

Fig. 5: Normal browsing on a legitimate website

In contrast, when a suspicious URL containing phishing characteristics was accessed, the extension immediately gen- erated a phishing alert, as shown in Fig. 6. Additional phishing detection scenarios are illustrated in Fig. 8, confirming consistent warning behavior across multiple phishing-like URLs.

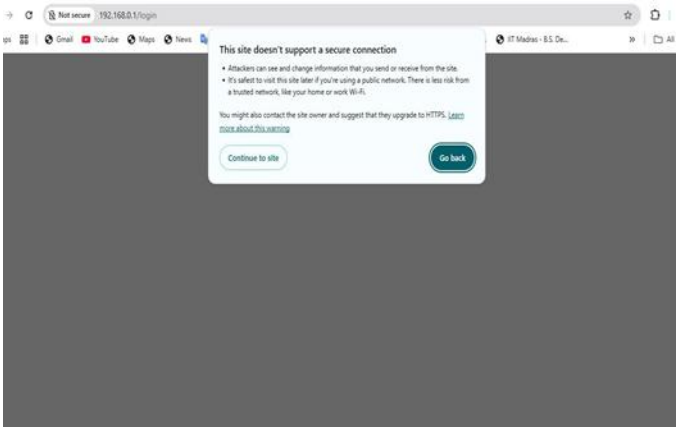

Fig. 6: Real-time phishing warning generated by the extension

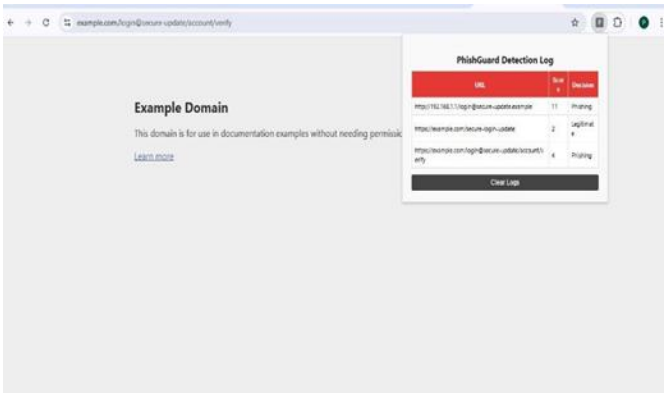

Fig. 7: PhishGuard detection log showing URL analysis, phishing score, and decision

Fig. 7 presents the internal detection log generated by the PhishGuard browser extension during real-time URL analysis. The log records the visited URLs along with their corresponding phishing scores and final classification decisions. Legitimate URLs are assigned low phishing scores and correctly classified as safe, whereas suspicious URLs containing phishing characteristics receive higher scores and are flagged as phishing. The scoring

mechanism is derived from the weighted feature importance obtained through offline Random Forest training. This detection log validates the transparency and explain- ability of the proposed system by clearly showing how phishing decisions are made in real time. It also demonstrates that the extension operates entirely on the client side without relying on external servers, thereby preserving user privacy while ensuring immediate threat detection. The implementation of the rule-based detection logic within the browser extension was verified through code inspection and debugging. Fig. 9 shows the JavaScript-based feature extraction and scoring logic implemented in the extension.

7. Discussion

The experimental results demonstrate that the proposed framework effectively bridges the gap between high-accuracy:

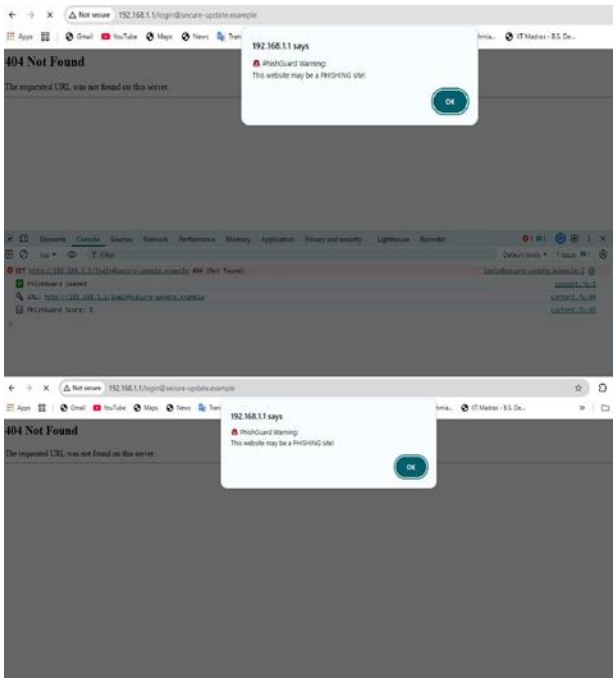

Fig. 8: Multiple phishing detection scenarios triggered by the extension

Offline machine learning models and practical real-time de- ployment. The Random Forest model provides strong detec- tion performance and interpretable feature importance, while the browser extension enables immediate phishing warnings with minimal computational overhead. This hybrid design ensures high accuracy, explain ability, user privacy, and real- world applicability, making the proposed system suitable for deployment in everyday browsing environments. Overall, the Random Forest classifier emerged as the most effective model for phishing detection. Security-related and URL-structure features were identified as the most influential indicators. The browser extension successfully detected phish- ing websites in real time, validating the practical usefulness of the proposed system.

8. Conclusion And Future Work

This work presented an efficient and practical phishing detection framework that bridges the gap between high- accuracy offline machine learning models and real-time client- side deployment. Multiple machine learning classifiers were evaluated using URL-based features, and the Random Forest model achieved the best overall performance in terms of accuracy, precision, recall, and F1-score. Feature importance analysis further enabled interpretability and identification of the most influential phishing indicators. To ensure real-world applicability, the learned knowledge was translated into a lightweight rule-weighted detection mechanism and deployed as a browser extension. The ex- perimental results demonstrated that the proposed system can accurately detect phishing websites in real time while maintaining low computational overhead and preserving user privacy by avoiding server-side communication. Despite its effectiveness, the proposed framework has cer- tain limitations. The current implementation primarily relies on URL-based features and does not analyze webpage content or visual cues. In addition, rule weights are derived offline and remain static during deployment.

Contributions

The key contributions of this work are summarized as follows:

• A comprehensive comparative evaluation of multiple ma- chine learning models for phishing detection using URL- based features.

• An explainable feature importance analysis using Ran- dom Forest to identify dominant phishing indicators.

• A novel translation of machine learning knowledge into a lightweight rule-weighted detection mechanism.

• Design and implementation of a real-time client-side phishing detection system as a browser extension.

• Experimental validation demonstrating high accuracy, low latency, and practical usability in real-world browsing environments.

Future work will focus on incorporating adaptive learning mechanisms to dynamically update feature weights based on evolving phishing strategies. The integration of webpage content, visual similarity analysis, and deep learning-based embedding's is another promising direction. Furthermore, ex- tending the system to support multiple browsers and mobile platforms, as well as conducting large-scale user studies, will further enhances its robustness and practical impact

References

1. Adebowale, M. A., Lwin, K. T., & Sanchez, E. (2021). Intelligent web phishing detection using random forest and neural networks. Future Generation Computer Systems, 115, 62–71. https://doi.org/10.1016/j.future.2020.08.004
2. Alkhalil, Z., Alsaedi, A., & Alharthi, A. (2023). Real-time phishing detection using deep learning. Computers & Security, 124, 102965. https://doi.org/10.1016/j.cose.2022.102965
3. Chiew, K.-L., Chang, E.-H., & Tiong, W.-K. (2018). Utilisation of website features for phishing detection. Computers & Security, 76, 1–13. https://doi.org/10.1016/j.cose.2018.02.004
4. Feng, J., & Zou, Y. (2021). Phishing detection using attention-based deep neural networks. Computers & Security, 102, 102152. https://doi.org/10.1016/j.cose.2020.102152
5. Garera, S., Provos, N., Chew, M., & Rubin, A. D. (2007). A framework for detection and measurement of phishing attacks. Proceedings of the ACM Workshop on Rapid Malcode. https://doi.org/10.1145/1314389.1314391
6. Huang, Y., & Qian, C. (2022). Hybrid deep learning model for phishing detection. IEEE Access, 10, 33421–33433. https://doi.org/10.1109/ACCESS.2022.3159874
7. Koray Sahingoz, O., Buber, E., Demir, O., & Diri, B. (2019). Machine learning based phishing detection from URLs. Expert Systems with Applications, 117, 345–357. https://doi.org/10.1016/j.eswa.2018.09.029
8. Liu, Z., & Chen, X. (2022). Explainable AI for cybersecurity: A survey. IEEE Transactions on Artificial Intelligence, 3(3), 401–417. https://doi.org/10.1109/TAI.2022.3146587
9. Ma, J., Saul, L. K., Savage, S., & Voelker, G. M. (2009). Beyond blacklists: Learning to detect malicious web sites from suspicious URLs. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1245–1254. https://doi.org/10.1145/1557019.1557153
10. Mishra, P., & Varadharajan, V. (2022). Survey on phishing detection techniques. IEEE Access, 10, 801–824. https://doi.org/10.1109/ACCESS.2021.3139458
11. Mohammad, R., Thabtah, F., & McCluskey, L. (2015). Phishing websites dataset. UCI Machine Learning Repository. https://doi.org/10.24432/C5GW2F
12. Niu, W., & Zhang, Y. (2021). URL representation learning for phishing detection. IEEE Transactions on Information Forensics and Security, 16, 3789–3803. https://doi.org/10.1109/TIFS.2021.3079155
13. Sarker, I. H. (2021). Machine learning for intelligent cybersecurity. IEEE Access, 9, 14670–14701. https://doi.org/10.1109/ACCESS.2021.3052497
14. Xiang, Y., Wang, C., & Zhao, L. (2023). Layered hybrid phishing detection using URL, text and image features. Sensors, 23, 8070. https://doi.org/10.3390/s23198070
15. Zhang, X., Zhou, J., & Wang, H. (2020). Phishing detection using multi-feature fusion and ensemble learning. IEEE Access, 8, 68348–68358. https://doi.org/10.1109/ACCESS.2020.2986124