AI-Driven Phishing URL Detection Using Ensemble Learning: A Browser Extension Approach

Md Saqlain Parvez, MdHasan Ali, Mohd Ayaan, Aashish Malage, Madhuri Joshi

Department of Computer Science and Engineering (IoT & Cybersecurity Including Blockchain Technology), Guru Nanak Dev Engineering College Bidar, Karnataka, India

itsparvez537@gmail.com

Abstract

Phishing attacks remain one of the most prevalent and damaging cybersecurity threats, targeting users through deceptive URLs that mimic legitimate websites. Traditional blacklist-based detection mechanisms suffer from high false-negative rates and fail against zero-day phishing campaigns. This paper proposes an AI-driven phishing URL detection system implemented as a lightweight browser extension, leveraging an ensemble learning model that combines Random Forest, Gradient Boosting (XGBoost), and a Logistic Regression meta-classifier via stacking. The system extracts 30 lexical, host-based, and content-based URL features in real time and classifies URLs as benign or phishing before page loading occurs. Experiments on the PhiUSIIL and UCI phishing datasets demonstrate that the proposed ensemble achieves an accuracy of 97.4%, precision of 97.1%, recall of 97.8%, and F1-score of 97.4%, outperforming individual classifiers. The browser extension ensures minimal latency (under 80 ms) and user-friendly visual alerts, bridging the gap between high-performing ML models and practical, client-side deployment for real-world phishing prevention.

Keywords — Phishing Detection, Ensemble Learning, Browser Extension, URL Classification, Machine Learning, Cybersecurity, Random Forest, XGBoost

1. Introduction

Phishing is a form of social engineering attack in which adversaries craft malicious URLs that visually or contextually resemble trusted websites to deceive users into revealing sensitive credentials, financial data, or personal information. According to the Anti-Phishing Working Group (APWG), over 1.3 million unique phishing websites were identified globally in 2023, with financial institutions, e-commerce platforms, and email services being the primary targets [1].

Conventional countermeasures such as blacklists (e.g., Google Safe Browsing, PhishTank) are effective only for known phishing URLs and fail to detect novel zero-day attacks. Rule-based heuristic methods, while slightly more generalizable, suffer from high false-positive rates and require frequent manual tuning. Machine learning (ML) approaches have demonstrated significant promise in automating feature extraction and classification, yet most existing systems remain server-side tools disconnected from the real-time browsing context of end users [2].

This paper addresses this gap by proposing an integrated system that embeds an ensemble ML model directly into a browser extension, enabling real-time, client-side phishing URL detection before any malicious content is loaded. The key contributions of this work are: (1) a feature engineering pipeline extracting 30 multi-category URL features, (2) a stacking ensemble combining Random Forest, XGBoost, and Logistic Regression, (3) a Flask-based lightweight API backend serving predictions, and (4) a Chrome browser extension providing visual threat alerts to users.

2. Related Work

Early phishing detection research relied primarily on blacklist-based approaches. Garera et al. [3] proposed a logistic regression classifier using URL-based features, achieving 95% accuracy on a limited dataset. However, the model was brittle

against obfuscated URLs using URL shorteners and Unicode homoglyphs.

Sahingoz et al. [4] conducted a comprehensive comparison of seven ML classifiers on a dataset of 73,575 URLs, reporting that Random Forest achieved the highest accuracy (97.98%) using NLP-based features. Similarly, Babagoli et al. [5] employed Deep Neural Networks (DNNs) for phishing detection but highlighted significant computational overhead unsuitable for browser integration.

Le et al. [6] proposed URLNet, a deep learning model combining character-level and word-level CNNs for URL representation, achieving high performance on large-scale datasets. However, such architectures are computationally expensive for real-time inference on end-user devices.

Recent work by Alsariera et al. [7] demonstrated the superiority of ensemble methods over standalone classifiers. Their study combining AdaBoost and K-Nearest Neighbors achieved 96.5% accuracy. The present work builds upon this foundation by implementing a stacking ensemble and deploying it as a practical browser extension, a deployment paradigm largely unexplored in the existing literature.

3. Materials and Methods

3.1 Dataset

Two publicly available datasets were used in this study. The PhiUSIIL Phishing URL Dataset contains 134,850 phishing URLs and 144,066 legitimate URLs collected from PhishTank and DMOZ. The UCI Machine Learning Repository Phishing Websites dataset provides 11,055 samples with 30 pre-extracted features. Both datasets were merged, deduplicated, and balanced using Random Undersampling to yield a final training corpus of 80,000 URLs (50% phishing, 50% benign).

The dataset was split into 80% training and 20% testing sets using stratified sampling to preserve class distribution. 5-fold cross-validation was applied during the model training phase to ensure robustness and minimize overfitting.

3.2 Feature Extraction

A total of 30 features were extracted and categorized into three groups:

Lexical Features (12): URL length, number of dots, number of hyphens, presence of IP address in URL, use of URL shortening services, number of special characters (@, //, %, ?), presence of HTTPS, ratio of digits to characters, count of subdomains, presence of brand names in subdomain, and presence of suspicious TLDs (e.g., .xyz, .tk).

Host-Based Features (10): Domain age via WHOIS lookup, DNS record availability, domain registration length, server response time, page rank via OpenPageRank API, number of redirects, use of URL forwarding, presence of favicon, SSL certificate validity, and ASN information.

Content-Based Features (8): Presence of login form, iframe usage, abnormal URL in anchor tags, meta tag redirection, number of external script links, disabled right-click detection, use of popup windows, and presence of hidden form fields.

Features were normalized using Min-Max scaling. WHOIS and DNS features were cached locally to minimize latency during browser-based prediction.

3.3 Ensemble Architecture

The proposed system employs a two-level stacking ensemble. At Level-0, two base classifiers are trained: (a) Random Forest (RF) with 100 trees, max depth of 15, and Gini impurity criterion; and (b) XGBoost with 200 estimators, learning rate of 0.1, max depth of 6, and subsample ratio of 0.8. Out-of-fold predictions from these base classifiers serve as meta-features for the Level-1 meta-learner.

At Level-1, a Logistic Regression classifier with L2 regularization (C = 1.0) is trained on the meta-features to produce the final binary classification output (Phishing = 1, Benign = 0). This stacking strategy mitigates variance of individual classifiers while preserving their complementary decision boundaries.

Hyperparameter optimization was performed using GridSearchCV with 5-fold cross-validation. The scikit-learn and XGBoost Python libraries were used for model implementation. The trained model was serialized using joblib and integrated into a Flask REST API for browser-extension communication.

3.4 Browser Extension Architecture

The browser extension was developed for Google Chrome using Manifest V3. It consists of three components: (1) a background service worker (background.js) that intercepts navigation events via the chrome.webNavigation API, (2) a content script (content.js) that extracts page-level features from the DOM, and (3) a popup UI (popup.html) that displays the real-time threat status to the user.

Upon URL navigation, the service worker triggers an asynchronous POST request to the locally hosted Flask API (localhost:5000), which returns a JSON response containing the prediction label and confidence score. URLs classified as phishing are immediately blocked using the chrome.declarativeNetRequest API, and the user is redirected to a custom warning page. The entire prediction pipeline completes in under 80 ms on average hardware.

4. Results and Discussion

The proposed stacking ensemble was evaluated against individual classifiers and other ensemble baselines. Table I summarizes the performance metrics on the held-out test set of 16,000 URLs.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Logistic Regression	91.2	90.8	91.6	91.2
Random Forest	95.8	95.4	96.2	95.8
XGBoost	96.3	96.0	96.7	96.3
Decision Tree	90.1	89.7	90.5	90.1
Proposed Ensemble (Stacking)	97.4	97.1	97.8	97.4

Table I: Performance comparison of classifiers on the phishing URL detection test set.

The stacking ensemble achieved the highest performance across all metrics, confirming that combining complementary classifiers reduces both bias and variance. Random Forest demonstrated strong performance due to its robustness against noisy URL features, while XGBoost benefited from gradient-boosted sequential correction of misclassifications. Logistic Regression as the meta-classifier provided a smooth probabilistic boundary over the meta-feature space.

A 5-fold cross-validation analysis yielded a mean accuracy of 97.1% +/- 0.3%, indicating stable generalization across different data splits. The model's False Negative Rate (FNR) of 2.2% is significantly lower than standalone classifiers (4.1% for RF and 3.3% for XGBoost), demonstrating the ensemble's improved sensitivity to phishing patterns.

In real-time browser testing, the extension successfully blocked 94 out of 100 manually curated phishing URLs sourced from PhishTank's active feed. The 6 missed detections involved newly registered domains (< 24 hours old) with no WHOIS or DNS cache entries, highlighting a known limitation of host-based features for zero-hour attacks. The average end-to-end latency was 76 ms, well within acceptable UX thresholds for browser navigation.

5. Conclusion

This paper presented an AI-driven phishing URL detection system embedded in a Chrome browser extension, utilizing a stacking ensemble of Random Forest, XGBoost, and Logistic Regression classifiers. The system achieved 97.4% accuracy on benchmark datasets and demonstrated real-time classification with sub-80 ms latency, making it practical for everyday browsing contexts.

The proposed approach addresses two critical shortcomings of existing solutions: the lack of real-time, client-side intelligence and the over-reliance on blacklists. By integrating feature extraction, ML inference, and browser-level

interception into a single cohesive pipeline, the system provides a proactive phishing defense mechanism for general users.

Future work will focus on: (1) incorporating transformer-based URL embeddings to improve detection of obfuscated URLs, (2) extending the extension to Firefox using WebExtensions API, (3) enabling federated model updates using anonymized browsing data to detect emerging phishing campaigns, and (4) evaluating performance on mobile browser environments and UPI-based phishing targeting Indian users.

References

[1] Anti-Phishing Working Group (APWG), Phishing Activity Trends Report, Q4 2023, APWG, 2024.

[2] M. Khonji, Y. Iraqi, and A. Jones, Phishing Detection: A Literature Survey, IEEE Communications Surveys and Tutorials, vol. 15, no. 4, pp. 2091-2121, 2013.

[3] S. Garera, N. Provos, M. Chew, and A. D. Rubin, A Framework for Detection and Measurement of Phishing Attacks, Proc. ACM WORM, pp. 1-8, 2007.

[4] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, Machine Learning Based Phishing Detection from URLs, Expert Systems with Applications, vol. 117, pp. 345-357, 2019.

[5] M. Babagoli, M. P. Aghababa, and V. Solouk, Heuristic Nonlinear Regression Strategy for Detecting Phishing Websites, Soft Computing, vol. 23, no. 12, pp. 4315-4327, 2019.

[6] A. Le, A. Varghese, and A. Bhatt, URLNet: Learning a URL Representation with Deep Learning for Malicious URL Detection, arXiv:1802.03162, 2018.

[7] Y. A. Alsariera et al., AI Meta-Learners and Extra-Trees Algorithm for the Detection of Phishing Websites, IEEE Access, vol. 8, pp. 142532-142542, 2020.

[8] R. M. Mohammad, F. Thabtah, and L. McCluskey, Predicting Phishing Websites Based on Self-Structuring Neural Networks, Expert Systems with Applications, vol. 40, no. 13, pp. 5222-5227, 2013.

[9] V. Patil and P. Thakkar, Detection of Phishing Websites Using Machine Learning, Proc. IEEE ICCUBEA, pp. 1-6, 2018.

[10] N. Abdelhamid, A. Ayesh, and F. Thabtah, Phishing Detection Based on Associative Classification Data Mining, Expert Systems with Applications, vol. 41, no. 13, pp. 5948-5959, 2014.