



A COMPREHENSIVE MACHINE LEARNING FRAMEWORK FOR PHISHING URL DETECTION USING LEXICAL, HOST-BASED AND DOMAIN-BASED FEATURES

¹Shipra Goyal, ² Dr. Rajeesh Yadav (Associate Professor)

¹Research Scholar, ²Research Supervisor

¹⁻²Department of Computer Science, Sabarmati University, Ahmedabad, Gujarat

Abstract

Phishing URLs remain a persistent cybersecurity threat because attackers can create new domains, manipulate URL strings, imitate trusted brands, and rapidly rotate hosting infrastructure. This analytical study develops a comprehensive machine learning framework for phishing URL detection by integrating three complementary feature families: lexical features derived directly from the URL string, host-based features describing network and server characteristics, and domain-based features describing registration, DNS, top-level-domain, age, and reputation signals. The study follows a literature-grounded analytical design rather than claiming newly collected primary observations. Recent public datasets and peer-reviewed studies are examined to identify robust feature groups, suitable preprocessing procedures, model families, evaluation criteria, and deployment constraints. The analysis compares Logistic Regression, Support Vector Machine, Decision Tree, Random Forest, Gradient Boosting/XGBoost, k-Nearest Neighbour, and Naive Bayes from the perspectives of predictive capability, computational cost, interpretability, and real-time suitability. Evidence from recent datasets shows that lexical features are fast and broadly available, while host- and domain-based features add contextual information that can reduce ambiguity in structurally deceptive URLs. Tree ensembles are especially well suited to mixed numerical and categorical security features, but high reported accuracy on a single dataset should not be treated as proof of generalization. Cross-dataset validation, temporal splitting, class-imbalance control, probability calibration, explainability, and false-positive analysis are therefore incorporated into the proposed framework.

Keywords: phishing URL, machine learning, lexical features, host-based features, domain-based features, cybersecurity, Random Forest, XGBoost, feature engineering

1. Introduction

Phishing is a social-engineering and technical deception technique in which an attacker attempts to persuade a user to visit a malicious resource, disclose credentials, approve a fraudulent transaction, or download harmful content. The URL is often the first machine-readable object available to a security system before a page is fully loaded. This makes URL-based detection attractive for browsers, secure web gateways, email filters, endpoint products, and threat-intelligence pipelines. Unlike static blacklists, machine learning can learn combinations of suspicious characteristics and can therefore identify previously unseen URLs that resemble known phishing patterns.

The detection problem is difficult because a malicious URL does not have one universal appearance. Attackers can use long paths, misleading subdomains, URL shortening, character substitution, embedded IP addresses, suspicious tokens, recently registered domains, uncommon top-level domains, redirects, and hosting services that make malicious infrastructure appear legitimate. Conversely, many benign websites also use long URLs, tracking parameters, content-delivery networks, redirects, and complex subdomains. A useful classifier must therefore distinguish suspicious



INTERNATIONAL JOURNAL OF ARTS, COMMERCE & EDUCATION

A Multidisciplinary Peer-Reviewed, Refereed, Open Access Journal,
<https://avjournals.com/> e-ISSN: 2347-7687, Vol 11, Issue 2, July-December 2023

combinations from ordinary web complexity rather than treating a single feature as conclusive evidence.

Recent research increasingly emphasizes feature engineering and hybrid feature sets. A 2023 systematic review in IEEE Access categorized phishing-detection features into URL/lexical, HTML/content, domain, behavioural, and derived or hybrid groups, while noting continued reliance on lexical attributes because they are inexpensive and broadly applicable (Kustiawan & Ghauth, 2023). The PhiUSIIL dataset released through the UCI Machine Learning Repository contains 235,795 URLs and 54 features, demonstrating the scale now available for supervised phishing research (Prasad & Chandra, 2024). Newer datasets such as LegitPhish and URL-Phish further emphasize structural, lexical, entropy, domain and TLD characteristics (Potpelwar et al., 2023; URL-Phish, 2023).

1.1 Purpose of the Study

The purpose of the study is to construct and analytically evaluate a machine learning framework for phishing URL detection that integrates lexical, host-based, and domain-based features. The study also identifies methodological safeguards needed to prevent inflated performance claims and to support realistic deployment.

2. Objectives of the Study

- To identify and classify the principal lexical, host-based, and domain-based features used for phishing URL detection.
- To examine how different feature families contribute complementary information to phishing classification.
- To compare major supervised machine learning algorithms for mixed-feature phishing URL detection.
- To develop a reproducible preprocessing, feature-selection, training, validation, and evaluation pipeline.
- To analyze the importance of class imbalance, temporal drift, cross-dataset validation, explainability, latency, and false-positive control.
- To propose a practical integrated framework suitable for research benchmarking and real-time phishing screening.

3. Research Questions and Analytical Hypotheses

The study is guided by the following research questions: Which URL characteristics provide the most useful evidence for distinguishing phishing from legitimate URLs? Does combining lexical information with host and domain context provide a more robust basis for classification than lexical information alone? Which conventional machine learning models offer the best balance between predictive performance, computational efficiency, and interpretability? How should a phishing detector be validated so that reported performance reflects generalization rather than dataset-specific patterns?

For a future empirical implementation, the primary null hypothesis (H_0) would state that the integrated lexical-host-domain feature set does not produce a statistically meaningful improvement in phishing classification compared with a lexical-only baseline. The alternative hypothesis (H_1) would state that the integrated feature set improves one or more relevant measures such as F1-score, recall, ROC-AUC, PR-AUC, or false-positive rate. A second analytical expectation is that ensemble tree methods will often perform strongly on heterogeneous tabular features, although no algorithm should be assumed superior without dataset-specific and cross-dataset testing.

4. Conceptual Framework of Phishing URL Detection

The proposed framework treats phishing detection as a supervised binary classification problem in which each URL is represented by a feature vector and assigned a label such as phishing or legitimate. The framework is deliberately



layered. The first layer captures URL syntax; the second captures host and network context; the third captures domain identity and registration context. After cleaning and transformation, features pass through selection and model-training stages. The trained classifier outputs a phishing probability or class decision, which is then evaluated using both discrimination metrics and operational measures such as false positives and inference latency.

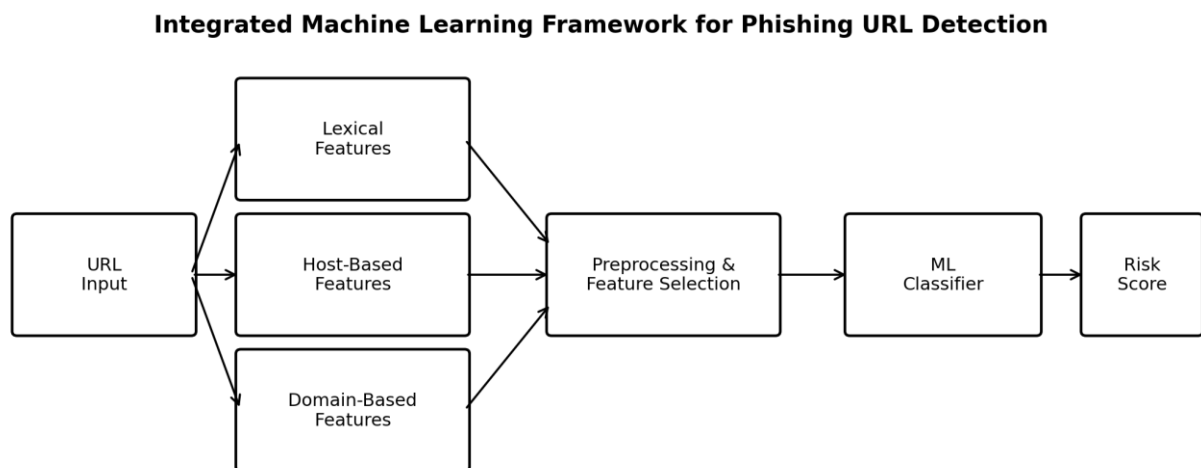


Figure 1. Integrated Machine Learning Framework for Phishing URL Detection

4.1 Lexical Features

Lexical features are derived from the characters, tokens, and structure of the URL itself. They are attractive because they can be computed quickly without contacting the destination server. Common examples include total URL length, hostname length, number of dots, slashes, hyphens, digits and special characters, number of subdomains, digit ratio, path depth, entropy, presence of an IP address, suspicious keywords, use of URL-shortening services, abnormal placement of a top-level domain, repeated characters, and brand-like tokens in the path or subdomain. LegitPhish, for example, provides 17 structural and lexical attributes, while the 2023 URL-Phish dataset provides 22 engineered numerical lexical and structural features (Potpelwar et al., 2023; URL-Phish, 2023).

The principal strength of lexical analysis is speed. It can be applied before a network request is completed, making it suitable for real-time filtering. Its weakness is ambiguity. Modern legitimate websites frequently contain long query strings, tracking parameters, content-delivery subdomains, and complex paths. Attackers can also deliberately construct short and clean-looking URLs. Lexical evidence is therefore best treated as one layer rather than a complete decision rule.

4.2 Host-Based Features

Host-based features describe properties associated with the server, DNS resolution, network location, and serving behaviour. Depending on the data-collection environment, useful variables can include DNS-record availability, IP-address characteristics, autonomous system information, redirect behaviour, response status, TLS availability, certificate-related signals, and hosting reputation. These features add infrastructure context that cannot be inferred reliably from the string alone. A URL may look ordinary yet resolve to suspicious or unstable infrastructure, while a



INTERNATIONAL JOURNAL OF ARTS, COMMERCE & EDUCATION

A Multidisciplinary Peer-Reviewed, Refereed, Open Access Journal,
<https://avjournals.com/> e-ISSN: 2347-7687, Vol 11, Issue 2, July-December 2023

complex URL hosted on a long-established platform may be benign.

Host-based extraction introduces practical costs. DNS and network queries can add latency, records may be temporarily unavailable, and live probing can create operational or safety concerns. A production system should therefore use cached or threat-intelligence-enriched host information where possible, define timeouts, and distinguish a missing value from a suspicious value. The framework treats missingness explicitly rather than automatically classifying unavailable host data as malicious.

5. Feature Taxonomy and Analytical Relevance

Feature family	Representative features	Security rationale	Main limitation
Lexical	URL length; hostname length; dots; hyphens; digits; entropy; subdomains; suspicious tokens; IP-in-URL; path depth	Captures obfuscation, token manipulation and abnormal URL construction	Legitimate modern URLs can also be long and complex
Host-based	DNS availability; IP/ASN context; redirects; TLS/certificate signals; response behaviour; hosting reputation	Adds infrastructure and serving context beyond the URL string	Requires external lookup, caching or network access; may add latency
Domain-based	Domain age; registration length; TLD; WHOIS status; registrar; popularity; DNS history; reputation	Captures identity, maturity and persistence of the registered domain	WHOIS privacy, compromised old domains and legitimate new domains create ambiguity
Derived/interaction	Digit ratio; character probability; entropy; brand-domain mismatch; age-length interactions; reputation combinations	Represents patterns that single raw variables may miss	Can increase dimensionality and risk dataset-specific overfitting

The analytical value of integration is not merely the number of features. Adding weak or redundant variables can increase noise, training time, and overfitting. The framework therefore separates feature extraction from feature selection. Correlation screening, mutual information, chi-square testing for suitable categorical variables, recursive feature elimination, L1 regularization, and tree-based importance can be used to identify compact subsets. Feature selection should be performed inside the training folds to avoid information leakage.



Complementary Feature Layers

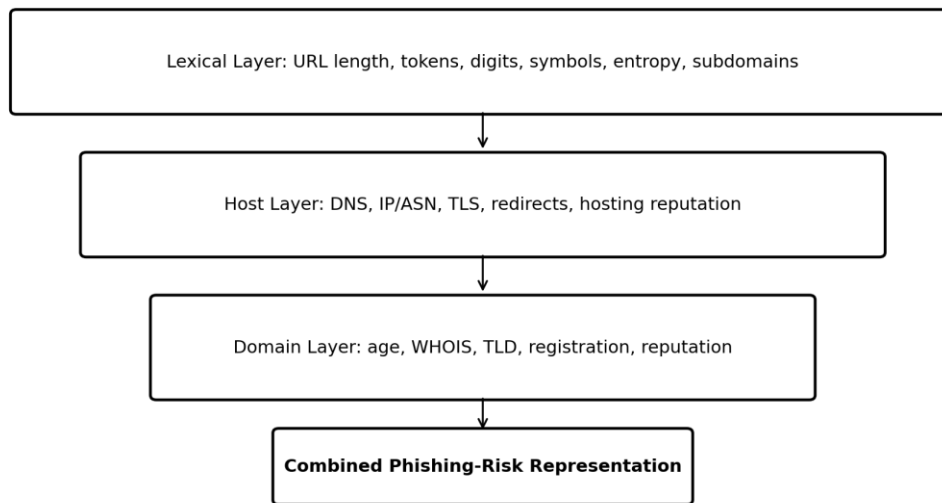


Figure 2. Complementary Lexical, Host-Based and Domain-Based Feature Layers

6. Review and Analytical Synthesis of Recent Literature

Prasad and Chandra (2024) introduced PhiUSIIL, a large phishing URL dataset containing 235,795 legitimate and phishing URLs. The UCI version records 54 features and includes variables such as URL length, domain length, IP-domain indication, HTTPS status, similarity-based attributes and multiple webpage-related signals. The scale of this dataset is important because it supports robust train-validation-test design, but its feature richness also demonstrates why researchers must clearly distinguish URL-only, domain, host and content signals when comparing models.

Kavya and Sumathi (2023) reviewed recent advances in phishing detection and described the movement from traditional machine learning toward deep learning and hybrid approaches. Their review reinforces a broader methodological point: high model complexity is not automatically equivalent to better operational detection. For structured tabular URL features, conventional models can remain highly competitive, easier to train, and easier to explain.

Kustiawan and Ghauth (2023) focused specifically on feature engineering. Their systematic review found continued dominance of URL and lexical features because of simplicity and applicability, while also identifying increasing interest in composite feature sets. The authors highlighted dataset limitations, evolving phishing tactics, high dimensionality, computational trade-offs, and the need for explainable methods such as SHAP and LIME. These issues directly motivate the present framework's emphasis on validation and feature-level explanation.

Potpelwar, Kulkarni and Waghmare (2023) presented LegitPhish, a manually verified dataset containing more than 100,000 labelled URLs with structural and lexical attributes including length, token count, entropy, subdomain use and TLD characteristics. A separate 2023 feature-engineered dataset reported 111,660 URLs with 22 numerical lexical and structural features and showed strong baseline performance for Random Forest and a MiniLM-plus-Logistic-Regression approach. These datasets support reproducible benchmarking, but they also reveal that reported scores depend strongly on sampling, labeling and feature construction.

A 2024 study on malicious URL detection combined lexical, host, content, structural, statistical and WHOIS features and evaluated Random Forest, Logistic Regression, Support Vector Machine, Naive Bayes and k-Nearest Neighbour



with multiple feature-selection strategies. The study reported strong performance for selected combinations and explicitly examined WHOIS contribution (Yadav et al., 2024). Another 2024 cross-dataset study used more than 90 candidate features spanning lexical, DNS/network, security/authentication, domain reputation and redirect behaviour, then selected common reliable indicators across datasets. This direction is methodologically significant because cross-dataset validation is harder to pass than random splitting within a single dataset.

Across the reviewed studies and datasets, three consistent methodological themes emerge. First, large modern benchmarks such as PhiUSIIL and LegitPhish make it possible to evaluate feature-engineered models at substantial scale. Second, recent reviews show that lexical features remain the most common starting point because they are fast and inexpensive, while domain, WHOIS, DNS, reputation and host signals increasingly appear in hybrid systems. Third, cross-dataset work shows that generalization should be tested explicitly because a feature that is highly predictive within one collection may reflect the dataset source rather than a stable phishing characteristic.

7. Proposed Comprehensive Machine Learning Framework

7.1 Data Acquisition and Label Quality

A defensible detector begins with label quality. Phishing samples should come from verified threat-intelligence or phishing repositories, while legitimate samples should represent a broad range of normal web activity rather than only a small list of highly popular domains. Duplicate URLs, near-duplicates, dead links and contradictory labels should be identified. When timestamps are available, they should be preserved because temporal evaluation is important for a threat that changes rapidly.

7.2 Data Cleaning and Preprocessing

URLs should be normalized cautiously. Converting hostnames to a consistent case, parsing scheme/host/path/query components, handling internationalized domain names, and recording rather than deleting suspicious encodings are appropriate. Over-aggressive normalization can remove the very patterns that indicate phishing. Numerical features can be standardized for models such as Logistic Regression, SVM and k-NN, whereas tree models generally do not require scaling. Missing host or WHOIS values should be encoded explicitly and, where appropriate, accompanied by a missingness indicator.

7.3 Feature Engineering

Feature extraction should proceed in three modules. The lexical module operates locally on the URL string. The host module retrieves or joins DNS, IP, TLS and hosting information subject to operational constraints. The domain module extracts registration, TLD, age and reputation characteristics. Derived features can then capture ratios, entropy, token distributions, brand-domain mismatch and interactions. Each module should retain provenance so that later explanations can identify whether a prediction is driven mainly by URL syntax, infrastructure or domain history.

7.4 Feature Selection

Feature selection is performed only on training data. Constant and near-constant variables are removed first, followed by duplicate or strongly redundant attributes. Filter methods such as mutual information can rank variables independently of a model; embedded methods such as L1 regularization and tree-based importance can identify predictive subsets; wrapper methods such as recursive feature elimination can be used when computationally feasible.



INTERNATIONAL JOURNAL OF ARTS, COMMERCE & EDUCATION

A Multidisciplinary Peer-Reviewed, Refereed, Open Access Journal,
<https://avjournals.com/> e-ISSN: 2347-7687, Vol 11, Issue 2, July-December 2023

Stability across folds is more important than selecting a feature merely because it ranks highly once.

7.5 Model Training

The framework recommends a diverse baseline set: Logistic Regression for a transparent linear reference; Decision Tree for interpretable non-linear rules; Random Forest for robust ensemble learning; gradient boosting or XGBoost for high-capacity tabular modeling; SVM for margin-based classification; k-NN as a distance-based comparator; and Naive Bayes as a computationally light probabilistic baseline. Hyperparameters must be tuned using validation folds rather than the final test set.

7.6 Validation Strategy

Random train-test splitting is insufficient as the sole validation method because URLs from the same campaign, domain family or collection period can appear in both partitions. The framework therefore recommends stratified cross-validation for model development, followed by a locked test set, a temporal holdout where timestamps permit, and an external or cross-dataset test when compatible features are available. Grouped splitting by registered domain can further reduce leakage from closely related URLs.

8. Comparative Analysis of Candidate Machine Learning Models

Algorithm	Strengths for phishing URL detection	Limitations	Expected role
Logistic Regression	Fast, interpretable coefficients, strong baseline, probability output	Linear boundary; requires scaling/encoding; may miss complex interactions	Baseline and explainable reference
Decision Tree	Human-readable rules; handles non-linearity and mixed scales	Prone to overfitting and instability	Interpretability benchmark
Random Forest	Handles interactions, mixed features and noise; robust importance estimates	Larger model; importance can be biased; less transparent than one tree	Strong general-purpose tabular model
XGBoost / Gradient Boosting	Powerful non-linear modeling; handles complex interactions; strong tabular performance	More tuning; can overfit dataset artifacts; explanation needed	High-performance candidate
Support Vector Machine	Effective in high-dimensional spaces; strong margins	Scaling required; expensive on large datasets; probability calibration extra	Useful for medium-sized engineered datasets
k-NN	Simple and non-parametric	Slow inference at scale; sensitive to scaling and irrelevant features	Comparator after aggressive feature selection
Naive Bayes	Very fast and lightweight	Independence assumptions often unrealistic for correlated URL features	Low-cost baseline

For this problem, Random Forest and gradient boosting are analytically attractive because lexical, host and domain variables contain non-linear thresholds and interactions. For example, a newly registered domain is not automatically suspicious, but a new domain combined with high URL entropy, many subdomains and a brand-like token may be highly informative. Tree ensembles can represent such interactions naturally. Logistic Regression remains important



INTERNATIONAL JOURNAL OF ARTS, COMMERCE & EDUCATION

A Multidisciplinary Peer-Reviewed, Refereed, Open Access Journal,
<https://avjournals.com/> e-ISSN: 2347-7687, Vol 11, Issue 2, July-December 2023

because its coefficients and calibrated probabilities can reveal whether the additional complexity of ensembles is justified.

No model should be selected solely on accuracy. If phishing samples are less frequent than legitimate samples, a model can achieve deceptively high accuracy by favoring the majority class. Precision, recall, F1-score, specificity, ROC-AUC and especially precision-recall AUC should therefore be reported. Confusion matrices should accompany aggregate scores so that false positives and false negatives remain visible.

9. Model Evaluation Framework

Metric	Meaning in phishing detection	Why it matters
Accuracy	Overall proportion correctly classified	Useful only when class balance and error costs are understood
Precision	Share of predicted phishing URLs that are truly phishing	Low precision produces excessive false alarms
Recall / Sensitivity	Share of phishing URLs correctly detected	Low recall allows attacks to pass undetected
F1-score	Harmonic balance of precision and recall	Useful when both error types matter
Specificity	Share of legitimate URLs correctly recognized	Directly reflects protection against false positives
ROC-AUC	Ranking quality across thresholds	Useful general discrimination measure
PR-AUC	Precision-recall performance across thresholds	More informative under strong class imbalance
False-positive rate	Legitimate URLs incorrectly blocked or warned	Critical for usability and operational trust
Inference latency	Time needed to produce a decision	Critical for browser, email and gateway deployment

Confidence intervals or repeated-fold variability should be reported where possible. A difference of a few tenths of a percentage point is not automatically meaningful if the variance across folds is larger. Threshold selection should be performed on validation data and then frozen before final testing. Probability calibration should be examined if scores are used for risk-based decision making.

9.1 Feature-Family Ablation Analysis

Ablation testing is essential to establish whether integration genuinely adds value. At minimum, four configurations should be compared: lexical-only; lexical plus host; lexical plus domain; and lexical plus host plus domain. If the full model improves recall but sharply increases latency, the deployment design may use a two-stage cascade in which a fast lexical model screens all URLs and only uncertain cases trigger host/domain enrichment. This design can preserve much of the contextual benefit while reducing average lookup cost.

9.2 Explainability

Explainability should operate at both global and local levels. Global feature importance can identify the variables most associated with phishing across the dataset. Local explanations can show why a particular URL received a high-risk score, such as unusual entropy, suspicious subdomain depth, very recent registration, missing DNS history, or brand-domain mismatch. SHAP or similar methods can support such analysis, but explanations should be interpreted as model reasoning rather than proof that a feature caused maliciousness.



10. Security, Generalization and Operational Challenges

10.1 Concept Drift and Adversarial Adaptation

Phishing is an adversarial problem. Once defenders rely heavily on visible patterns such as excessive length or obvious suspicious words, attackers can alter those patterns. Models therefore face concept drift: the statistical relationship between features and labels changes over time. Periodic retraining, drift monitoring, rolling temporal evaluation and recent threat-intelligence feeds are necessary. A model that performs well on an old random split may fail on new campaigns.

10.2 Dataset Leakage and Duplicate Campaigns

Near-duplicate URLs can inflate test scores when members of the same phishing campaign appear in both training and test partitions. Domain-level grouping, duplicate removal and temporal separation reduce this risk. Researchers should document data sources, collection dates, deduplication rules and splitting logic so that unusually high scores can be interpreted responsibly.

10.3 Class Imbalance

Real traffic is rarely balanced exactly between phishing and legitimate URLs. Training data may nevertheless be artificially balanced for convenience. Evaluation should therefore include realistic class proportions or threshold analysis that reflects the intended environment. Class weights, undersampling, carefully applied oversampling, and threshold adjustment can be considered, but resampling must occur only inside the training process.

10.4 Availability and Latency of External Features

Host and domain features can be unavailable, stale or slow. WHOIS privacy rules and registrar practices can reduce access to registration details. DNS results can change quickly. A robust system should cache external features with timestamps, define fallback behavior, and measure end-to-end latency. Missing external data should not cause a silent shift to a different decision rule.

11. Discussion

The literature and dataset evidence support a layered interpretation of phishing URL detection. Lexical features provide an efficient first line of analysis because they are immediately available and capture many common obfuscation strategies. Their continued prominence in recent research is therefore justified. However, lexical patterns alone cannot reliably distinguish all malicious URLs from the increasingly complex structure of legitimate web applications. Host and domain features provide additional context that can resolve some of this ambiguity.

The strongest conceptual advantage of the integrated framework is complementarity. A URL string captures presentation and construction; host features capture infrastructure; domain features capture identity and history. These are different dimensions of the same security event. Combining them allows the classifier to recognize interactions that would be invisible to a single feature family. This is consistent with recent work that integrates lexical, host, WHOIS, DNS, reputation and structural information and with cross-dataset studies that search for features stable across data sources.

At the same time, adding external features creates a deployment trade-off. Lexical extraction is local and fast, while host and domain enrichment may require network calls and may be affected by privacy, availability and caching. For



real-time systems, a cascade architecture is therefore attractive: a lightweight lexical model can immediately classify obvious low- and high-risk cases, while ambiguous cases are enriched with host and domain information and passed to a stronger ensemble model.

12. Limitations and Directions for Future Research

The present study is analytical and literature-grounded; it does not claim a newly collected experimental dataset or newly measured classifier scores. Consequently, the proposed feature interactions and model expectations should be validated on a clearly documented benchmark before they are treated as empirical findings. Differences among public datasets in collection date, labeling, feature definitions and class balance also limit direct numerical comparison across published studies.

Future research should prioritize temporal and cross-dataset testing, because generalization is the central challenge in an adversarial environment. Public benchmarks should preserve timestamps and registered-domain groups to support leakage-resistant splitting. Researchers should also report inference latency and external-query cost, not only classification metrics.

Further work can examine character-level neural networks, transformers, graph-based domain relationships and large language model representations, but these methods should be compared against strong engineered-feature baselines. Hybrid systems may be especially promising when learned URL embeddings are combined with explicit domain age, DNS, reputation and infrastructure features. Explainable AI should be used to test whether the model relies on defensible security indicators or accidental dataset artifacts.

Adversarial robustness is another important direction. Attackers can deliberately alter URL length, tokens, subdomains and hosting patterns to evade learned rules. Robustness testing should therefore include controlled perturbations and campaign-level evaluation. Privacy-preserving and lightweight feature extraction is also important for mobile devices, browsers and resource-constrained environments.

14. Conclusion and Recommendations

A comprehensive phishing URL detector should not depend on a single suspicious token, a single feature family, or a single headline accuracy value. Lexical features provide fast and useful evidence, host-based features add infrastructure context, and domain-based features add identity, age and reputation context. Their integration creates a more complete representation of phishing risk and allows machine learning models to learn interactions among URL construction, hosting behaviour and domain history.

For structured feature sets, Random Forest and gradient-boosted trees are strong candidates because they can represent non-linear interactions without extensive manual transformation. Nevertheless, transparent baselines such as Logistic Regression should always be retained, and model selection should be based on precision, recall, F1-score, PR-AUC, false-positive rate, latency and external generalization rather than accuracy alone. Feature selection must be performed within training folds, and test data must remain untouched until the final evaluation.

14.1 Recommendations

- Use lexical, host-based and domain-based features as complementary evidence rather than independent proof of phishing.



INTERNATIONAL JOURNAL OF ARTS, COMMERCE & EDUCATION

A Multidisciplinary Peer-Reviewed, Refereed, Open Access Journal,
<https://avjournals.com/> e-ISSN: 2347-7687, Vol 11, Issue 2, July-December 2023

- Benchmark at least one transparent baseline and one ensemble model under the same data split and feature conditions.
- Perform domain-grouped, temporal and cross-dataset evaluation wherever the data permit.
- Report confusion matrices, precision, recall, F1-score, ROC-AUC, PR-AUC and false-positive rate in addition to accuracy.
- Measure feature-extraction and inference latency so that model quality is evaluated in operational terms.
- Use feature selection and explainability to reduce redundant variables and detect reliance on dataset artifacts.
- Monitor concept drift and retrain using recent, verified phishing and legitimate URLs.
- Consider a two-stage architecture in which lexical screening is followed by host/domain enrichment for uncertain cases.



INTERNATIONAL JOURNAL OF ARTS, COMMERCE & EDUCATION

A Multidisciplinary Peer-Reviewed, Refereed, Open Access Journal,
<https://avjournals.com/> e-ISSN: 2347-7687, Vol 11, Issue 2, July-December 2023

References

- Kavya, S., & Sumathi, D. (2023). Staying ahead of phishers: A review of recent advances and emerging methodologies in phishing detection. *Artificial Intelligence Review*, 58, 50. <https://doi.org/10.1007/s10462-024-11055-z>
- Kustiawan, Y. A., & Ghauth, K. I. (2023). Feature engineering for phishing website detection using machine learning: A systematic review. *IEEE Access*, 13, 192080-192104. <https://doi.org/10.1109/ACCESS.2023.3630334>
- Potpelwar, R. S., Kulkarni, U. V., & Waghmare, J. M. (2023). LegitPhish: A large-scale annotated dataset for URL-based phishing detection. *Data in Brief*, 63, 111972. <https://doi.org/10.1016/j.dib.2023.111972>
- Prasad, A., & Chandra, S. (2024). PhiUSIIL: A diverse security profile empowered phishing URL detection framework based on similarity index and incremental learning. *Computers & Security*, 136, 103545. <https://doi.org/10.1016/j.cose.2023.103545>
- Prasad, A., & Chandra, S. (2024). PhiUSIIL Phishing URL (Website) [Dataset]. UCI Machine Learning Repository. <https://doi.org/10.1016/j.cose.2023.103545>
- Tian, Y., Yu, Y., Sun, J., & Wang, Y. (2023). From past to present: A survey of malicious URL detection techniques, datasets and code repositories. *Computer Science Review*, 58, 100810. <https://doi.org/10.1016/j.cosrev.2023.100810>
- URL-Phish dataset authors. (2023). A feature-engineered dataset of benign and phishing URLs for machine learning and large language models evaluation. *Data in Brief*, 63, 112162. <https://doi.org/10.1016/j.dib.2023.112162>
- Yadav, D. K., Tejash, M., Panda, S., Mishra, P., Nayak, S., & Sekh, A. A. (2024). Phishing and legitimate websites dataset [Data set]. Zenodo.
- Yadav, D. K., et al. (2024). Machine learning-based malicious URL detection using feature selection techniques and WHOIS features. *Frontiers in Computer Science*.
- Yi, et al. (2024). Phishing URL detection and interpretability with machine learning: A cross-dataset approach. *Security and Privacy*.
- Phishing URL detection in real-time: Machine learning and deep learning approaches on a custom threat intelligence dataset. (2024). *Procedia Computer Science*, 283, 4859-4879. <https://doi.org/10.1016/j.procs.2024.06.541>
- XGBoost-based URL phishing detection method with cross-dataset validation. (2024). *IEEE Access*. <https://doi.org/10.1109/ACCESS.2024.3672690>