

Phishing URL Classification Using Lexical Analysis and Machine Learning with an AI-Based Detection Stage

Dr. Shilpa B. Sarvaiya¹, Neha Suresh Rao Choudhari², Reshma Satish Gupta³

Head, Department of MCA¹, Student, Department of MCA^{2,3}

Vidya Bharati Mahavidyalaya, Amravati, Maharashtra, India

¹sarvaiya.shilpa@gmail.com, ²nehachoudhari157@gmail.com ³reshmasgupta@gmail.com

Abstract: Phishing links imitate trusted brands and push users to hand over passwords, card numbers, or other private data. This paper proposes a lightweight, explainable pipeline that decides whether a URL is phishing or legitimate by reading only the address text, with no visit to the destination page. Measurable properties of the string, such as its length, digit and symbol counts, subdomain depth, suspicious keywords, raw-IP usage, and protocol, are turned into a numeric vector for a supervised classifier. Beyond the lexical stage, the paper adds an AI-based detection stage in which a character-level deep-learning model re-examines URLs that the first stage scores with low confidence, and the two scores are fused into a final allow, warn, or block decision. The work is a methodology and literature study: it contains no new experimental accuracy figures, and every number quoted from earlier work is labelled as such. Particular attention is given to duplicate removal, leakage prevention, time-aware testing, and a metric set that includes recall and false-negative rate. Four figures and six tables summarise the pipeline, the feature set, the candidate models, the evaluation protocol, and the limitations.

Keywords: Phishing URL, lexical analysis, machine learning, AI-based detection, deep learning, URL classification, cybersecurity, feature extraction.

I. INTRODUCTION

People now bank, shop, study, consult doctors, and pay bills online, and each of these activities gives criminals a chance to trick them. Phishing is among the most frequent of these tricks. The attacker sends a message or builds a web page that looks authentic, hoping the victim will type confidential details or click something harmful.

Classic defences lean on blocklists and reputation feeds. They work well for addresses that are already known, but a freshly registered phishing domain is often live for hours before any list records it. Learning-based detectors address this gap by generalising from labelled examples: they judge an address by how it looks rather than by whether it has been seen before. Lexical analysis suits an early checkpoint especially well, because the text of the URL is available instantly and nothing needs to be downloaded.

A URL carries many measurable signals, including overall length, the number of dots, hyphens and digits, how many subdomains precede the main domain, whether a bare IP address replaces a name, and whether words like login or verify appear. No single signal is decisive, since honest websites can also have long and messy addresses. A classifier earns its value by weighing many weak clues together.

This paper makes three contributions:

- a transparent lexical pipeline for phishing URL classification, with every feature defined and justified;
- a two-stage design in which an AI-based character-level model supports the lexical classifier on uncertain URLs; and



- a leakage-aware evaluation protocol with a clear list of limitations and mitigations.

No new experimental accuracy is claimed. Figures reported by earlier authors are treated strictly as literature findings, and the proposed design is offered as a blueprint for an independent implementation.

II. RELATED WORK

Earlier research has drawn on several sources of evidence: the URL string, domain registration data, page content, network behaviour, and mixtures of these. Table I lists the studies discussed in this section.

A. Lexical and Host-Based URL Analysis

An early and influential line of work by Ma et al. showed that suspicious URLs can be separated from benign ones using lexical and host-based descriptors together with online learning, without inspecting page content [5], [6]. Blum et al. studied online learning on lexical URL features for phishing detection [11], and Le, Markopoulou, and Faloutsos found with their PhishDef system that lexical-only features cost roughly 1% accuracy compared with full feature sets, while staying in the 96–98% range [10]. Abutaha et al. later studied lexical phishing detection on a collection of over one million labelled URLs, extracted 22 features, and compared Random Forest, Gradient Boosting, a neural network, and an SVM; they reported 99.89% accuracy for the SVM on their data [1]. Joshi et al. also relied on static lexical features, avoiding both crawling and blocklists, and reported an average accuracy of 92%, an average AUC of 0.98, and an average false-negative rate of 0.1% over five test sets [2]. Sahingoz et al. similarly examined phishing detection from URL-derived features, including word-based ones, with several classifiers [9].

B. Feature Engineering

Kumar blended URL attributes with natural-language processing and word-vector representations, and reported 99.75% accuracy for a Random Forest using those combined features [3]. This shows that URL text can be represented in richer ways than simple counts, at the price of more preprocessing. Kustiawan and Ghauth surveyed feature engineering for phishing website detection and sorted features into URL/lexical, HTML/content, domain, behavioural, and hybrid groups [4]. They stress that lexical features stay popular because they are cheap and broadly usable, and they point to changing attacker tactics, dataset limitations, and high-dimensional feature spaces as recurring difficulties.

C. Deep Learning and AI-Based URL Detection

Instead of relying on hand-built features, deep models learn a representation directly from the raw string. Le et al. introduced URLNet, which applies convolutional networks to both character-level and word-level views of a URL [7]. Sahoo et al. surveyed machine-learning methods for malicious URL detection and discussed feature types, learning settings, and open challenges such as concept drift [8]. These works motivate the second stage of the framework proposed here.

TABLE I
Summary of Reviewed Studies (Values as Reported by the Original Authors)

Study	Year	Data / features	Method and reported outcome
Ma et al. [5], [6]	2009	Lexical and host-based URL features	Online learning for URL classification; large-scale study (no figure reproduced here)
Blum et al. [11]	2010	Lexical URL features	Online learning for phishing URL detection
Le, Markopoulou & Faloutsos [10]	2011	URL names only (lexical features)	Lexical-only accuracy about 1% below full features, still 96–98%
Le et al. [7]	2018	Raw URL as characters and words	URLNet: convolutional model learning URL representation
Sahingoz et al. [9]	2019	URL-derived features including word-based ones	Several classifiers compared (no figure reproduced here)
Joshi et al. [2]	2019	Static lexical features, no crawling or blocklist	Average accuracy 92%, average AUC 0.98, average FNR 0.1% over five test sets



Study	Year	Data / features	Method and reported outcome
Abutaha et al. [1]	2021	More than 1 million URLs, 22 lexical features	RF, GB, NN and SVM compared; SVM reported 99.89% accuracy
Kumar [3]	2023	URL features plus NLP and word vectors	Random Forest reported 99.75% accuracy
Kustiawan & Ghauth [4]	2025	Systematic review of feature groups	Categorises URL, HTML, domain, behavioural and hybrid features

D. Research Gap

The reviewed studies show that very high scores are achievable, yet those scores do not carry over automatically to other data. When similar or repeated URLs fall on both sides of a train/test split, the model is rewarded for memory instead of understanding. Phishing campaigns also change quickly, so a detector fitted to old samples can weaken against new ones. The gap targeted here is therefore not the pursuit of one more high percentage. It is a lightweight and understandable process, with an optional AI-based second opinion, whose limits can be measured honestly.

III. PROPOSED METHODOLOGY

A. Overview

The system labels every incoming URL as phishing or legitimate using only the text of the address. Fig. 1 shows how an example address breaks into parts that supply features, and Fig. 2 shows the end-to-end experimental workflow from raw data to error analysis.

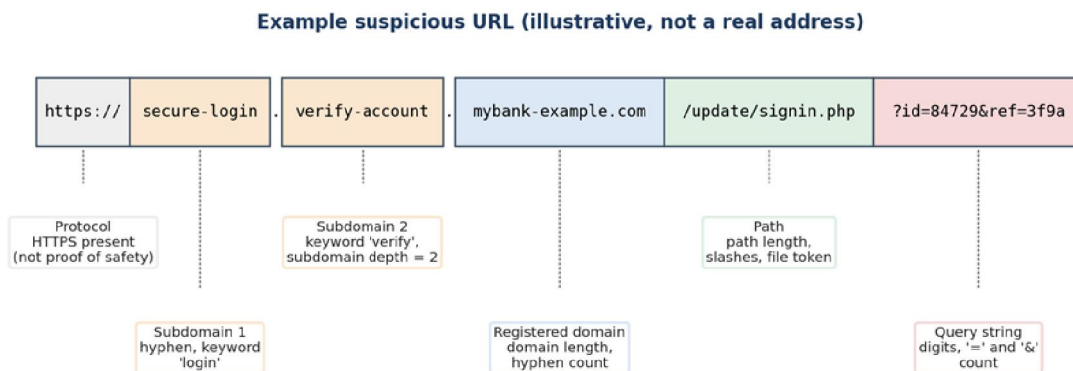


Fig. 1. Anatomy of an illustrative suspicious URL and the lexical features each part supplies.



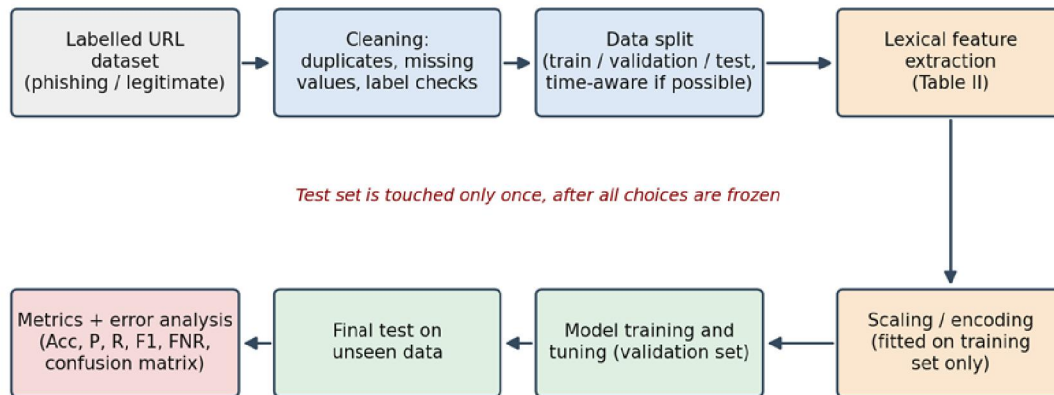


Fig. 2. Experimental workflow of the proposed framework.

B. Dataset Collection and Preparation

The study needs a labelled corpus of phishing and legitimate URLs. Before any modelling, the data should be screened for repeated addresses, contradictory labels, missing entries, and a skewed class ratio. Exact duplicates, and where practical near-duplicates that differ only in a trivial token, should be reduced to a single copy so that no URL appears in both training and testing. The corpus is then split into training, validation, and test portions, and when timestamps exist the split should follow time, with older samples for training and newer ones for testing.

Any value learned from data, such as a scaling mean or a vocabulary, must be computed on the training portion alone. The test portion stays untouched until every design decision has been fixed.

C. Lexical Feature Extraction

All features in Table II are computed directly from the URL string. Together they describe size, symbol usage, digit usage, host structure, wording, and protocol.

TABLE II
Lexical Features and the Reason Each is Informative

Feature group	Examples	Why it can signal phishing
Length	URL, hostname, path and query length	Long addresses can hide the real domain or pack in extra tokens
Special characters	Counts of dots, slashes, hyphens, underscores, '?', '=', '&', '@'	Unusual symbol density often appears in deceptive or auto-generated links
Digits	Digit count and digit-to-length ratio	Random identifiers and numeric tricks are common in throw-away domains
Host structure	Subdomain count, domain depth, hyphens in domain	Attackers stack subdomains so a trusted name appears early
IP usage	Binary flag: numeric IP instead of a name	Legitimate services rarely expose raw IP addresses to users
Keywords	login, account, verify, secure, update, payment	Words that pressure users into entering credentials
Protocol	HTTP or HTTPS flag	Weak alone: HTTPS does not prove that a site is safe



Feature group	Examples	Why it can signal phishing
Statistical	Token count, character entropy	Random-looking strings tend to have different entropy from brand names

D. Feature Representation

Each URL becomes one numeric vector. Yes/no properties are coded as 0 or 1, while counts and lengths stay as numbers. Magnitude-sensitive models such as SVM and logistic regression benefit from scaling, and the scaler is fitted on training data only so that nothing from the test set leaks into learning.

E. Stage 1: Lexical Classifiers

Several standard supervised models can be compared on the same vectors (Table III). The choice should weigh phishing recall, false alarms, interpretability, speed, and computing cost together, not accuracy alone. In a security setting a missed phishing URL usually costs more than a wrongly flagged legitimate one.

TABLE III

Candidate Stage 1 Classifiers (Qualitative Comparison)

Model	Interpretability	Prediction speed	Remark
Logistic Regression	High (coefficients)	Very fast	Strong baseline; needs scaled inputs
Decision Tree	High (readable rules)	Very fast	Easy to explain; can overfit
Random Forest	Medium (feature importance)	Fast	Robust on mixed features; common choice in the literature [1], [3], [12]
SVM	Low to medium	Moderate	Effective with scaling; reported strong in [1]; classic method [13]
Gradient Boosting	Medium	Fast	Often accurate; needs careful tuning

F. Stage 2: AI-Based Detection

The second stage adds a deep-learning model that reads the URL as a sequence of characters, and optionally words, so it can pick up patterns that hand-built counts miss, such as look-alike spellings or odd character runs [7]. Candidate architectures are a character-level CNN, a bidirectional LSTM, or a small Transformer encoder. To keep the system fast, this model is not run on every URL. Fig. 3 shows the intended flow: Stage 1 scores each URL, and only addresses whose probability falls in an uncertain band, for example between two thresholds chosen on validation data, are passed to the AI model. The two scores are then fused, by weighted voting or by a small stacking model, into allow, warn, or block.



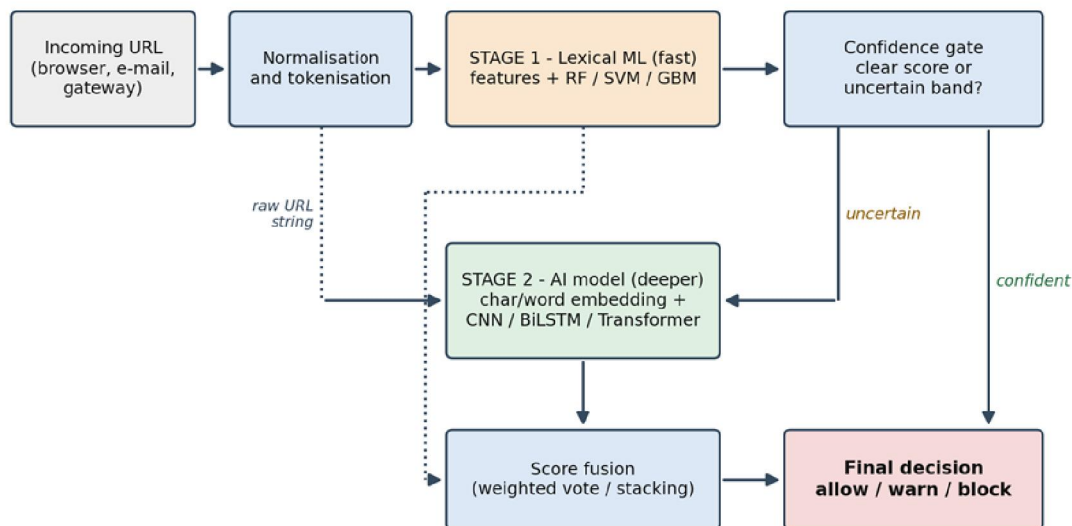


Fig. 3. Proposed two-stage architecture: lexical machine learning followed by an AI-based model for uncertain cases. The AI stage costs more computation and is harder to explain than Stage 1. Explainability tools such as SHAP [14] or LIME [15] can partly offset this, and the thresholds that decide which URLs reach Stage 2 become tunable settings that balance speed against detection quality.

G. Evaluation Measures

Accuracy is the share of all predictions that are right. Precision shows how many URLs flagged as phishing really are phishing. Recall shows how many real phishing URLs are caught. F1-score balances precision and recall. The false-negative rate gives the share of phishing URLs wrongly allowed through, which is the most dangerous error. Table IV gives the formulas, and Fig. 4 shows the confusion matrix from which they are computed. A real experiment must report this matrix, because accuracy alone can hide poor recall when classes are imbalanced.

TABLE IV
Evaluation Metrics (Phishing is the Positive Class)

Metric	Formula	Meaning for phishing detection
Accuracy	$(TP + TN) / (TP + TN + FP + FN)$	Overall correctness; misleading when classes are imbalanced
Precision	$TP / (TP + FP)$	How trustworthy a phishing alert is
Recall (TPR)	$TP / (TP + FN)$	How much phishing is actually caught
F1-score	$2 \times \text{Precision} \times \text{Recall} / (\text{Precision} + \text{Recall})$	Single balance of precision and recall
False-negative rate	$FN / (TP + FN)$	Share of phishing that slips through as legitimate
False-positive rate	$FP / (FP + TN)$	Share of legitimate URLs wrongly flagged



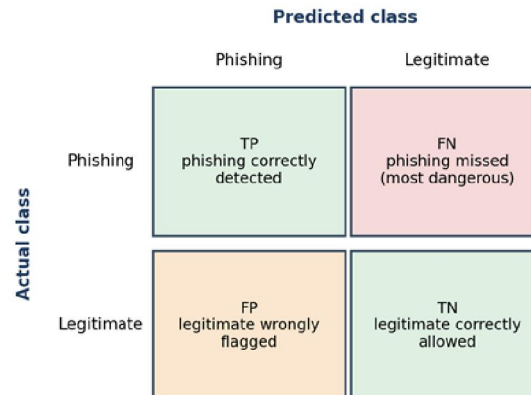


Fig. 4. Confusion-matrix template to be filled with real results in an experiment.

IV. DISCUSSION

A. Value of Lexical Analysis

Phishing URLs often differ structurally from ordinary ones: they may be unusually long, nest many subdomains, use odd symbols or numbers, or borrow a trusted brand name in the path. Because all of this is visible without contacting the site, lexical analysis works as a quick first filter. Still, no single property is dependable. A genuine service can have a long address with many parameters, while a skilful attacker can craft a short and ordinary-looking one. Combining signals through machine learning is therefore preferable to a fixed rule, and the result should be read as a risk signal, not as proof.

B. Comparing Approaches and Published Findings

The percentages in Table I came from different datasets, feature sets, preprocessing choices, and test procedures, so they cannot be ranked against each other. What they collectively show is that URL features are useful, that richer representations can help [3], [7], and that hybrid features can add robustness where more information is available [4]. Table V contrasts the three broad approaches discussed in this paper.

TABLE V
Comparison of Detection Approaches

Aspect	Blocklist / reputation	Lexical ML (Stage 1)	AI-based model (Stage 2)
New, unseen URLs	Weak until listed	Good	Good
Needs page download	No	No	No
Speed	Very fast lookup	Very fast	Slower; run selectively
Explainability	High	High	Lower; needs SHAP [14] / LIME [15]
Data and compute need	Low	Low to moderate	Higher
Main weakness	Delay in listing new attacks	Hand-built features can miss patterns	Drift and adversarial URLs

C. Limitations and Mitigations

Table VI lists the main risks of a URL-only approach and the practical steps that reduce each one.



TABLE VI
Limitations and Corresponding Mitigations

Limitation	Mitigation
Dataset sampling or source bias	Use several sources; report dataset origin and class ratio
Duplicates across train and test	Remove exact and near-duplicates before splitting
Campaigns evolve over time	Time-based split; periodic retraining and monitoring
HTTPS misread as safe	Treat the protocol as a weak feature only
No view of page content or redirects	Combine with domain, content and reputation checks
High accuracy hiding poor recall	Report recall, FNR, F1 and the confusion matrix
Adversarial or look-alike URLs	Augment training data; use the AI stage and retrain regularly

D. Practical Application

The framework can sit inside a browser extension, an e-mail security tool, or a network gateway. When a URL arrives, features are extracted and a first verdict is produced before the user opens the link. A production system should still pair this component with reputation services and other controls rather than rely on one model.

E. Future Scope

Future work can run time-based evaluation so older training data face newer campaigns, and can add domain age, DNS records, certificate details, redirect chains, page content, and reputation signals. Explainable-AI methods such as SHAP [14] or LIME [15] can show which URL traits drove a decision. Lightweight in-browser deployment, privacy-preserving feature processing, and robustness testing against adversarial URLs are further directions.

V. CONCLUSION

Phishing remains dangerous because fraudulent links can be made to look like familiar services. This paper described a lightweight pipeline that classifies URLs from their text alone, using lexical features and supervised learning, and extended it with an AI-based second stage that re-checks uncertain cases. Four figures and six tables set out the workflow, the features, the candidate models, the metrics, and the limitations.

The literature indicates that lexical features support fast screening without loading the target page, but reported performance depends heavily on data and protocol, so published accuracy values are not results of this study. An independent implementation should report accuracy, precision, recall, F1-score, false-negative rate, and a confusion matrix on its own, duplicate-free, time-aware test data.

The main strengths of the approach are simplicity, speed, and explainability, with the AI stage adding depth where the lexical model is unsure. Lexical analysis alone cannot catch every technique, so a stronger future system should combine URL evidence with domain, content, behavioural, and reputation information.

ACKNOWLEDGMENT

The authors thank their institution for academic guidance and resources, and acknowledge the researchers whose published work formed the basis of the literature reviewed here.

REFERENCES

1. M. Abutaha, M. Ababneh, K. Mahmoud, and S. Al-Haj Baddar, "URL Phishing Detection using Machine Learning Techniques based on URLs Lexical Analysis," in Proc. 12th Int. Conf. Information and Communication Systems (ICICS), 2021, pp. 180–185, doi: 10.1109/ICICS52457.2021.9464539.
2. Joshi, L. Lloyd, P. Westin, and S. Seethapathy, "Using Lexical Features for Malicious URL Detection—A Machine Learning Approach," arXiv:1910.06277, 2019, doi: 10.48550/arXiv.1910.06277.
3. J. Kumar, "Hybrid Feature-Based Machine Learning Method for Phishing URL Detection," in Proc. 3rd Int. Conf. Secure Cyber Computing and Communication (ICSCCC), 2023, pp. 222–227, doi: 10.1109/ICSCCC58608.2023.10176901.



4. Y. A. Kustiawan and K. I. Ghauth, "Feature Engineering for Phishing Website Detection Using Machine Learning: A Systematic Review," IEEE Access, vol. 13, pp. 192080–192104, 2025, doi: 10.1109/ACCESS.2025.3630334.
5. J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond Blacklists: Learning to Detect Malicious Web Sites from Suspicious URLs," in Proc. 15th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2009, pp. 1245–1254, doi: 10.1145/1557019.1557153.
6. J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Identifying Suspicious URLs: An Application of Large-Scale Online Learning," in Proc. 26th Annu. Int. Conf. Machine Learning (ICML), 2009, pp. 681–688, doi: 10.1145/1553374.1553462.
7. H. Le, Q. Pham, D. Sahoo, and S. C. H. Hoi, "URLNet: Learning a URL Representation with Deep Learning for Malicious URL Detection," arXiv:1802.03162, 2018.
8. D. Sahoo, C. Liu, and S. C. H. Hoi, "Malicious URL Detection using Machine Learning: A Survey," arXiv:1701.07179, 2017.
9. 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, doi: 10.1016/j.eswa.2018.09.029.
10. Le, A. Markopoulou, and M. Faloutsos, "PhishDef: URL Names Say It All," in Proc. IEEE INFOCOM, 2011, pp. 191–195, arXiv:1009.2275.
11. Blum, B. Wardman, T. Solorio, and G. Warner, "Lexical Feature Based Phishing URL Detection Using Online Learning," in Proc. 3rd ACM Workshop on Artificial Intelligence and Security (AISec), 2010, pp. 54–60.
12. L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.
13. Cortes and V. Vapnik, "Support-Vector Networks," Machine Learning, vol. 20, no. 3, pp. 273–297, 1995, doi: 10.1007/BF00994018.
14. S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017, pp. 4765–4774.
15. M. T. Ribeiro, S. Singh, and C. Guestrin, "“Why Should I Trust You?”: Explaining the Predictions of Any Classifier," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 1135–1144, doi: 10.1145/2939672.2939778.

