# Ai-Based Detection Of Phishing Websites Using Url And Web Features

**Atharva Sarade[1], Utkarsh Borade[2], Prarthana Dhage[3], Sumit Pawar[4],Prof.Ajit Mali[5]**
Dept of Computer Science and Engineering
[1, 2, 3, 4, 5] Vishwakarma University Kondwa,Pune

***Abstract-*** *Phishing websites imitate trusted websites to persuade visitors to reveal passwords, banking details, and other private information. Because attackers can change page content and register new domains quickly, a detection method should consider more than a website's visual appearance. This paper presents a proposed machine-learning approach for identifying phishing websites by combining URL-based indicators with features collected from the webpage and its domain context. URL features may include URL length, use of an IP address, suspicious symbols, subdomain structure, and HTTPS-related indicators. Web features may include redirects, forms that request sensitive information, external-resource patterns, and selected domain or certificate metadata where these data are available. The proposed workflow covers data collection, preprocessing, feature extraction, model training, and evaluation. Candidate classifiers such as Logistic Regression, Decision Tree, Random Forest, and Support Vector Machine can be compared using precision, recall, F1-score, and ROC-AUC, with particular attention to the cost of false negatives. This document describes a research design rather than claiming completed experimental results. The final paper should report measured results from a documented dataset and reproducible tests before submission.*

***Keywords***— phishing detection, machine learning, URL analysis, website security, feature extraction

## I. INTRODUCTION

The web is used for communication, education, shopping, banking, and many other everyday activities. This convenience also creates opportunities for cybercriminals to build pages that resemble legitimate services. A phishing page may use a familiar brand name, a misleading link, or a convincing login form to make a visitor believe that the page is genuine. If the visitor trusts the page, sensitive information can be exposed to an attacker.

Blocking phishing pages is challenging because malicious URLs and hosting locations can change frequently. A simple blocklist is useful for known threats, but it may not identify a newly created page that has not yet been reported. Similarly, checking only whether a URL contains a suspicious word can produce errors: legitimate sites may have long URLs, while malicious sites can use short and ordinary-looking addresses. A more robust approach can combine multiple independent signals.

This paper proposes an AI-based detection workflow that examines both the URL and selected web features. The central idea is to use complementary evidence: URL structure provides an inexpensive first view, while page and domain features provide additional context. A trained classifier can learn patterns from labelled examples and assign a predicted class to an unseen website. The prediction should be treated as a risk signal, not as an absolute guarantee that a site is safe or malicious.

A. Objectives

- Identify URL and webpage indicators that may help distinguish phishing pages from legitimate pages.
- Describe a repeatable pipeline for cleaning data, extracting features, training classifiers, and testing predictions.
- Compare suitable machine-learning models using metrics that reflect both missed attacks and false alarms.
- Discuss practical limits, including changing attacker behaviour, incomplete page access, and privacy considerations.

## II. RELATED WORK

Earlier phishing-detection systems have used URL blacklists, manually designed rules, webpage content, and machine-learning classifiers. Blacklists can be effective when a URL is already known, but their usefulness depends on timely updates. Rule-based methods are easy to explain, although attackers may adapt their URLs to avoid individual rules. Machine-learning approaches instead learn combinations of indicators from labelled examples, but their quality depends on the representativeness of the data and on careful evaluation.

Research on phishing detection has explored lexical URL properties, host and domain information, HTML and JavaScript characteristics, and combinations of these sources. A multi-feature design can reduce dependence on one indicator. However, web features are not always available: pages may be offline, require scripts, block automated access,

or change after collection. For this reason, a practical study should document how every feature is collected and how missing values are handled. The final submitted version should expand this section with a verified review of relevant papers and cite each source accurately.

## III. PROPOSED METHODOLOGY

The proposed system follows six stages: obtain labelled examples, clean and validate the records, extract URL features, collect permitted webpage and domain features, train candidate classifiers, and evaluate them on data that were not used for training. Figure 1 can be represented as: Dataset → Preprocessing → Feature Extraction → Model Training → Testing and Metrics → Phishing/Legitimate Prediction.

### A. Dataset Preparation

The dataset should contain both phishing and legitimate URLs, with labels supported by a reliable source. Record the dataset name, collection date, label definition, number of examples in each class, and any filtering steps. Remove exact duplicates and inspect conflicting labels. To avoid overly optimistic results, split the data before fitting preprocessing steps, and group related URLs or domains where possible so that near-duplicates do not appear in both training and test sets. A time-based test split is also valuable because it better reflects the appearance of new phishing campaigns.

### B. URL-Based Features

URL features can be extracted without opening the destination page. Candidate features include total URL length; hostname length; number of dots and subdomains; presence of an IP address in place of a domain name; use of an at-sign or unusual punctuation; number of hyphens; path and query length; use of URL shortening services; and whether the scheme is HTTPS. These indicators must not be treated as definitive rules. For example, HTTPS encrypts a connection but does not prove that a website is trustworthy, and a long URL may be legitimate.

### C. Web and Domain Features

When it is safe and permitted to access a page, additional features may be gathered in an isolated analysis environment. Possible features include the number of redirects, the presence and destination of forms, whether a form requests password or payment fields, the proportion of resources loaded from other domains, and selected HTML or JavaScript patterns. Domain age, DNS information, and certificate metadata may be included when they are available from reliable sources. These values should be collected consistently and stored with timestamps because website properties can change. The crawler must not submit forms, enter real credentials, or execute untrusted content on a normal personal device.

### D. Feature Processing and Model Training

Convert extracted properties into a structured feature table. Encode categorical values consistently, scale features when required by the selected algorithm, and define a documented strategy for missing values. The same fitted preprocessing pipeline must be applied to training and test records. Candidate models include Logistic Regression as a baseline, Decision Tree for interpretable splits, Random Forest for non-linear feature combinations, and Support Vector Machine for classification in a transformed feature space. Hyperparameters should be selected using training data or cross-validation only; the held-out test set should be used for the final evaluation.

### E. Prediction Workflow

For a new URL, the system first parses the address and calculates URL features. If permitted page and domain information can be obtained, those features are added; otherwise, the system records missing values and uses the documented fallback strategy. The trained pipeline then returns a class prediction and, if calibrated, a risk score. A deployment can use this score to flag a page for additional review rather than automatically blocking every uncertain result.

## IV. EVALUATION PLAN

Model quality should be measured on a held-out test set containing examples not used to fit the model. Accuracy alone is insufficient, especially when the two classes are imbalanced. Precision measures the share of predicted phishing cases that are actually phishing; recall measures the share of phishing cases correctly identified; and the F1-score balances precision and recall. ROC-AUC can summarize ranking performance across thresholds. The confusion matrix should also be reported so readers can see false positives and false negatives.

A fair comparison should use the same data split and evaluation procedure for every candidate model. Report the number of examples per class, the decision threshold, the feature groups used, and any limitations in data collection. It is useful to compare a URL-only model with a combined URL-and-web-feature model to estimate whether the additional features provide a measurable benefit. Training time, prediction time, missing-feature rate, and performance on a later time period can provide further practical context.

A. Results to Be Added After Testing

No numerical performance results are asserted in this draft because no dataset or experimental output was provided. Before publication, run the experiments and replace this paragraph with a results table containing each model's precision, recall, F1-score, ROC-AUC, and test-set size. Discuss the observed errors and compare the combined-feature model against the URL-only baseline. Do not estimate or invent scores.

## V. DISCUSSION

Combining URL and web features may provide a broader view than relying on a single signal, but it also introduces costs. Fetching a page takes more time than analysing a URL, and some sites cannot be inspected reliably. Attackers may change their infrastructure or page code to avoid learned patterns, while legitimate websites may share features with phishing pages. A model trained on one collection period may therefore lose accuracy as the web changes.

The system should be designed with safe collection practices and data minimisation. Page inspection should use an isolated environment, avoid interacting with forms, and follow the dataset provider's terms. Any stored URLs or page snapshots should be handled carefully because query strings can sometimes contain personal or session information. A production tool should provide an explanation or supporting indicators for its warning and offer a route for human review when confidence is low.

## CONCLUSION

This paper outlines a proposed approach for detecting phishing websites by combining URL structure with selected webpage and domain indicators. It describes a pipeline for dataset preparation, feature extraction, model training, and evaluation, and identifies metrics that are more informative than accuracy alone. The approach is intended to support early identification of suspicious websites, not to replace user awareness or other security controls. The next step is to implement the pipeline, test it on a documented and appropriately split dataset, and report reproducible measurements. Only after those experiments and a verified literature review are complete should the draft be presented as a results-based research paper.

## ACKNOWLEDGMENT

The authors may acknowledge their project guide, department, institution, dataset providers, or other contributors here, where applicable. Include only people and organisations that actually supported the work.

## REFERENCES

[1] Bergholz, J. De Beer, S. Glahn, M.-F. Moens, G. Paaß, and S. Strobel, “New filtering approaches for phishing email,” Journal of Computer Security, vol. 18, no. 1, pp. 7–35, 2010.

[2] R. M. Mohammad, F. Thabtah, and L. McCluskey, “Predicting phishing websites based on self-structuring neural network,” Neural Computing and Applications, vol. 25, pp. 443–458, 2014.

[3] G. Xiang, J. Hong, C. P. Rose, and L. Cranor, “CANTINA+: A feature-rich machine learning framework for detecting phishing web sites,” ACM Transactions on Information and System Security, vol. 14, no. 2, Article 21, 2011.

[4] Anti-Phishing Working Group, “Phishing Activity Trends Reports.” Add the exact report edition, publication date, and URL consulted in the final version.

[5] Add the exact dataset citation and access date for the dataset used in the experiments.