International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 12 | Dec - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Lightweight Phishing URL Detection Using Hybrid Lexical–Metadata Features: A Machine Learning Approach

Ayan Chaudhuri

Vellore Institute of Technology, Vellore

***Abstract*— One of the most significant forms of cyber attacks is phishing websites, which exploit human users to acquire sensitive information (e.g., passwords, banking info, or personal identifiers). Although there are several detection techniques to identify phishing links through thorough and extensive analysis based on content type; however, even techniques such as deep learning will require computational resources to perform the analysis, thereby slowing progress toward the actual implementation of these techniques for detecting phishing URLs in the real-world environment. This paper presents a new lightweight and efficient machine learning model for detecting phishing URLs using very few lexical and metadata features of the URL string, and does not require access to or rendering of the actual web page.**

The complete dataset was constructed by obtaining verified OpenPhish (the most significant source of phishing detection across a large number of websites) links combined with Tranco (a legitimate domain name provider) domains. A variety of machine learning algorithms were compared - the Decision Tree algorithm, Random Forest algorithm, and Logistic Regression algorithm. Ultimately, the Logistic Regression performed best based on the evaluation criteria: overall model accuracy of 0.991, area under the receiver operating characteristic (ROC) curve score of 0.966, and an F1 score of 0.865. In addition, it achieved a favorable precision/recall tradeoff and demonstrated good computational performance when used in conjunction with human judgement to interpret results. Due to the logistic regression's simple, efficient, and robust nature as well as its strong ability to identify fraudulent activity, it is well suited for application to a variety of real-time security implementations, including as a web browser plug-in, email filter, or endpoint solution.

Keywords*— *Phishing detection, URL classification, lexical features, metadata features, machine learning, cybersecurity.

1. Introduction

Phishing remains the leading vector of the most frequently employed and devastating digital security threats. Phishing is associated with numerous security events (and incidents, including a variety of scenarios and platforms) including identity theft, identity fraud, unlawful access to computer systems, substantial economic losses to individuals and businesses, and much more. As the internet becomes increasingly sophisticated, so do the techniques utilized by cybercriminals to conduct phishing attacks (now referred to as phishing modalities), presenting a variety of attacks to multiple targets. In modern times, cybercriminals employ advanced techniques, such as social engineering, spoofed domains, masked URLs, and redirect techniques to bypass normal technical filters to prevent phishing attempts and manipulate human cognition and biases [1][2].

Globally, the frequency of breaches relating to pointers attributable to phishing type or other forms of social engineering are >80% of all reported breaches [19]. For instance, the Anti-Phishing Working Group (APWG) reported over 4.7 million phishing URLs in 2024 alone [14], representing significant volume and automated efforts in conducting phishing campaigns. Cybercriminals engage in many instances of malicious phishing by mimicking the appearance of legitimate and legitimate financial service providers and eCommerce websites, often creating content/photos used for phishing employing compound artificial intelligence (AI) [20].

For many years, there have been a number of different ways to detect phishing attacks. Traditionally, phishing detection methods relied upon one or more of the following mechanisms: content-based analyses (e.g. reviewing all portions of an Internet page), conducting WHOIS searches to identify phishers, and/or evaluating known/phished websites through various blacklist mechanisms.[1],[2]. All of these techniques work well in relation to detection of previously identified threats; however, they sometimes require frequent updates and periodic assessment of their ability to identify phishing websites using the least amount of computing resources to achieve an acceptable level of performance. In addition, attackers have developed new methods to eliminate detection barriers. Attackers may also register new domains nearly instantaneously, utilize abbreviations when creating URLs, or hide their final destination (the real website) from the victim through multiple redirections.[6]. As such, while content-driven and blacklists-based techniques to identify phishing based URLs generally are somewhat effective at identifying phishing websites, such techniques do not perform effectively when the URL is obfuscated or driven by dynamically generated content.

To solve these challenges, a novel URL Detection Model has emerged, which moves away from suppression based on content or known domains, performing suppression based on the actual URL itself, using the lexical and structural patterns of the URL.

URL Detection Models utilize an analysis of the URL string as well as any associated metadata (e.g., Domain Age, SSL Certificate, TLD Type) to allow for rapid and efficient classifications without having to download and analyze an entire webpage's content, [3], [8], [11]. Therefore, this leads to an ability to perform these classifications quickly, while also providing the ability to perform Real-Time Classification at the Browser, Firewall and Mail Gateway levels.

The URL Detection Models also allow for the interception of Suspect URLs prior to Users actually interacting with the URLs, allowing for earlier actions at the Cyber Kill Chain stage.

The research performed in foundational literature (e.g., Ma, et al, 2018 [3]) has shown that the combination of both lexical and host-based features can provide an adequate differentiation between "malicious" and "benign" URLs. In more recent work (i.e., deep learning models: CNN-based detectors [7] and URLNet [4]) have achieved superior accuracy over the previous models, but these neural network-based architectures have increased computational resource requirements creating a barrier for use in resource limited environments such as web browsers and IoT security gateways [8], [13].

This current study investigates an alternative hybrid model that combines both lexical and metadata-driven features for the purpose of identifying phishing URLs. Simple lexical indicators (i.e., URL Length, Token count, Symbol Entropy, and Digit Ratio) are combined into a single composite indicator, using complementary metadata signals (i.e., SSL presence, use of IP address, TLD Category and Domain Age) to create a balanced representation in order to produce a high-performance yet interpretable hybrid model for the purpose of identifying phishing URLs in real time.

The objective of this research is to make the following contributions:

1. A feature extraction pipeline combining lexical and metadata features to create a more comprehensive URL representation;
2. Presenting a thorough comparative evaluation of different machine learning classifiers to determine the best performing lightweight algorithm and emphasising the reliability of Logistic Regression in terms of.
3. Presenting a threshold-based analysis of the precision-recall trade-off under varying operational conditions, in order to ensure the system is ready for deployment.
4. The last contribution of this project is the focus on the needs of the research community, specifically the need for open research and reproducibility. Through the provision of a publicly available Jupyter Notebook implementation, based on publicly available datasets, this research will allow other researchers to build on or compare the methods proposed in this study.

This study aims to provide a reliable and efficient means of detecting phishing, bridging the gap between highly accurate methods using deep learning with lightweight models that are appropriate for the development and deployment of real-world cyber threat response systems at scale.

2. Related Work

The advancements in phishing detection have moved from using heuristic filtering and blacklisting toward leveraging machine learning via data-driven and neural network architectures. Early research endeavored to apply machine learning techniques to identify phishing sites through rule-based and URL pattern matching; however, these methods have struggled with obfuscated URLs and with zero-day phishing sites. One of the earliest uses of a machine-learning model for phishing detection was by Fette et al. [5], who created an email example model that used handcrafted email attributes that served to filter out unwanted URLs, and then checked the URLs again before verifying their corresponding websites. An example of an adaptation of this process is the work of Ma et al. [3], who improved on the early heuristic approaches and integrated lexical and host-based features into a Support Vector Machine (SVM) classification model in order to increase the capability of detecting phishing attacks.

Following this, researchers began associating feature sources amongst different features (i.e., combining the lexical aspects with an additional data layer, such as DNS, WHOIS information or SSL certificate info) [6]. By doing this, the ability of a given model to withstand URL obfuscation and detect newly registered domains improved. Subsequently, deep learning algorithms were created to perform well when trained on the URL datasets available at that time. Some of these algorithms, URLNet, [4] CNN-based URL classifiers [7], and RNNs, have been proven to be effective in their testing on various benchmark datasets; however, due to their need for a substantial amount of GPU processing and so many preprocessing steps at the time of data collection, they cannot be used in real-time scenarios or within low-to-no-volume resource environments like web browsers or IoT devices.

Currently, more attention is being paid to finding ways to successfully unify both traditional and lightweight detection models to obtain similar levels of interpretability and

accuracy; this is especially true in the case of hybrid models. Combining metadata and domain intelligence with lexical-driven models provides a similar or improved classification capability over traditional models. For example, Lin et al. [11] created a hybrid algorithm that consisted of both lexical detections and content-based detections and was able to achieve improved recall rates when tested over a wide range of datasets. Additionally, Ali et al. [13] and Thakur and Yadav [12] both highlight the importance of using ensemble methods as opposed to only relying on classifiers when dealing with an imbalanced dataset, such as those found in the area of phishing detection.

Recently, adversarial techniques, such as phishing and morphing, have been discovered by Zhang and Wang [14]. In addition, Yan and Zhao discovered that adversarial robustness could be maintained through using robust lexical feature extraction [15]. Additional studies have looked at using graphene [16] and transformers [17] in order to build phishing detection systems based on relationships between URLs at the node level and contextual embeddings from subdomains. Reinforcement learning-based approaches for dynamically adapting URL-based filters based on real-time feedback and improving detection ability as concept drift occurs are being explored through [18].

In summary, the general trend of research in this area has been toward building lightweight, interpretable models that can achieve deep-learning-level performance at significantly lower cost by using contextual metadata. This paper proposes a hybrid model that takes into account lexical and metadata features, and utilizes class-weighted learning, providing an efficient, modular approach for real-time detection of phishing URLs.

3. Methodology

3.1 Dataset and Preprocessing

3.1.1. Data Sources

The URLs were obtained from two distinct but complementary sources to create a comprehensive and representative dataset that contains both phishing and legitimate websites. The phishing URL set was sourced from the OpenPhish threat intelligence feed (OpenPhish), which is a verified collection of Phishing Domains contributed by different vendors (internet service providers). By doing this, the OpenPhish feed ensures a much lower false-positive rate by validating submissions through manual inspection and automated means before being posted to the feed.

To create the set of benign (non-Phishing) URLs, the Tranco Top 10,000 List was used as it is a stable composite ranking (aggregation) of data collected from several different sources, including Alexa (website ranking), Cisco Umbrella (internet security solution), and Majestic (backlink profiles). Aggregating the data in this manner provides a more accurate representation of what people browse on the internet, including highly trafficked sites in the areas of online banking, e-commerce, education, entertainment, and government agencies.

A number of data cleaning procedures were conducted to further validate the dataset. For example, all duplicate domains were eliminated and any URLs that were discovered to be inactive via either DNS resolution or HTTP requests were removed. Additionally, all reported error URLs and empty page redirects were removed, as were any URLs that returned timeout error messages. This step is critical in validating that phishing domains may exist for only a very short period of time, typically only a few hours or days.

The amassed dataset eventually contained around ten thousand unique web addresses ~(10 300) where the breakdown was 300 phishing URLs over 10 000 legitimate websites. By design, the balance between the datasets is not equal (ie 80% to 20%; Train vs Test), however the distribution of each dataset retains their original class composition's percentages which provides an avenue to evaluate any bias towards the larger class's performance and allows for assessing performance on smaller classes (in this scenario, phishing websites).

3.1.2. Preprocessing

Standardized, analyzable inputs were created by converting the unstructured URLs into preprocessed version with changes made to the formatting (all lower case letters), purging of white space, removal of session ids, utm tags, and any other excess url parameters that provided noise and offered no value to the assessment process to build a logic-based decision-making model to determine legitimate and/or phishing urls.

Using python's URLLIB. parse module to break apart the URL into various components (scheme, subdomain, host, path, and query), the ability to calculate token counts and pathlength of relevant features that could be beneficial in constructing a legitimate and/or phishing logical decision-making model (learning algorithm).

URLs that were malformed and/or incomplete based on regular expression validation were removed. URLs that were contained within duplicate (same URL) cases were also removed to limit repeating of a learning signal. To mitigate against domain-based data leakage, all URLs that were contained within a registered domain were stored in one partition (either training or testing) for the model during its training phase to promote the model's ability to learn generalized tendencies of phishing activity versus remembering trends associated with specific domains.

The validated, cleaned dataset was saved as a serialized structured CSV in preparation for further processing as Figure

1 illustrates the workflow involving URL acquisition, pre-processing of URLs, feature extraction, model training, and model deployment.

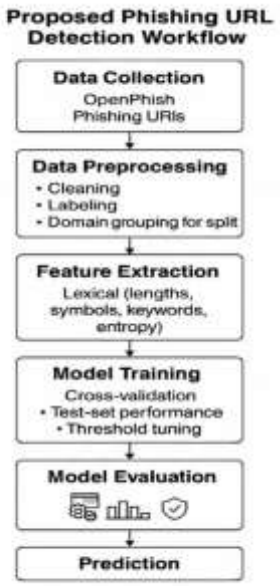

Figure 1. Proposed Phishing URL Detection Workflow.

Figure 1. Proposed Phishing URL Detection Workflow.

3.2. Feature Engineering

Feature engineering forms the conceptual backbone of the model, translating raw strings and contextual metadata into quantitative representations usable by learning algorithms. Two complementary categories—lexical and metadata features—were extracted to capture both syntactic irregularities and environmental attributes of a URL.

3.2.1. Lexical Features

The first analytical layer of the proposed framework is lexical features, which are based purely on the textual structure encoded in the URL. The main benefit of lexical features is in speed and self-sufficiency. Because lexical features rely solely on strings without any external look-up or API call, they can be computed on the user's device and are particularly useful for real-time inspection scenarios such as browser plug-ins, mail filters, and security gateways, where latency must be below a few milliseconds per request.

Lexical features can be categorized into three overarching families: length-based lexicon features, character-based lexicon features, and ratio/structural indicators.

- Length-based lexicon features (e.g., total URL length, total host length, total path length, total query length) assess the overall design complexity of the hyperlink. Phishing studies have shown that phishing URLs are often much longer than legitimate ones because attackers may place unnecessary tokens (or nested directories) to conceal the actual nature of a malicious payload and to mimic legitimate brand names [8]. In addition, URLs that appear too long or hierarchical in information depth often attempt to push legitimate content further right in the address bar, since users commonly do not read the full URL.

- Character-based metrics count the occurrences of characters such as "/", "-", "_", "@", "=", "%", and "?". These characters are frequently used to manipulate token boundaries or encode payloads in obfuscated links. The inclusion of "@" or "//" in unconventional positions can redirect browsers, while high symbol frequency usually reflects automated generation rather than human-crafted domains.

- Ratio-based or structural features include digit-to-letter proportion, number of dots, and subdomain depth. Phishing domains often insert random digits or additional subdomains (e.g., login-secure-verify.bank.example.com) to mimic legitimate hierarchies and bypass naïve blacklist checks. Counting such segments helps quantify URL "depth" and deviation from ordinary domain grammar.

In addition to looking for syntactic patterns, semantic signals were detected by searching for contextual keywords such as "login", "secure", "update", "verify", and "account". Previous literature has linked the occurrence of these tokens in either the host or in the path components of a URL as somewhat correlated with credential-harvesting intent, since attackers often create pages that mimic legitimate authentication pages. To quantitatively assess lexical randomness, the Shannon entropy was calculated for each hostname. High-entropy strings, which are primarily composed of seemingly random characters, generally indicate algorithmically-generated domains, or fast-flux cases where attackers circumvent the use of established static blacklists. Entropy therefore serves as a statistical proxy to indicate how unpredictable the domain is, and can efficiently discriminate between domains that are crafted to look corporate (low entropy), versus randomly-generated suspects (high entropy).

Finally, composite lexical features were produced, including the ratio of symbols to length and special-character density, which summarize the concentration of odd-looking characters in comparison to the total URL length. These aggregate signifiers can aid the model to detect subtle obfuscation patterns that single-feature analysis may not fully capture.

Altogether, this collection of light-weight textual features provides a brief and informative representation of a URL. The computations are linear in string length, entirely memory based, and extremely low in CPU overhead. Despite being lightweight, they retain substantial discrimination power that enable real-time detection of phishing attacks, where interpretability, speed, and transparency are as important as accuracy.

3.2.2 Metadata Features

Metadata attributes augment the examination of lexical cues by contextual and environmental insight about the domain's origin, configuration, and operational activity. While lexical indicators (descriptors) account for syntactic anomalies, metadata accounts for the "who" and "how" of a URL; details that adversary actors may have a more difficult time disguising. With the addition of metacharacteristics, the proposed model moves beyond surface-level trigger events (strings) that indicate malicious intent toward more robust, explainable classification.

A primary metadata characteristic is the HTTPS scheme indicator. Legitimate domains almost always implement SSL/TLS certificates to secure communications, whereas phishing domains typically do not encrypt communications, or may deploy a self-signed, invalid or expired certificate [6],[11]. Consequently, the existence of HTTPS and its associated particulars may signal both authenticity and maturity of a domain. Another strong indicator of domain authenticity is whether the URL location references a hostname or an IP address. Cybercriminals frequently deploy numericon IPs in a URL to circumvent domain blacklists or host several malicious movements under a solitary server [12]. This characteristic is consistent with a number of phishing datasets that have revealed generally greater than 95% of legitimate sites validly resolve their registered name as opposed to a raw IP representation.

Another important aspect is the Top-Level Domain (TLD) category. Some low-cost or country-code TLDs have been exploited due to the inexpensive registration fee and the lack of restrictions on registrars (e.g. .tk, .top, .xyz, .cf and .ml) [13]. Including TLD information in the model distinguishes between high-trust domains (e.g. .com, .org, .edu) and moderately qualified to potentially fraudulent domains. Similarly, the domain registration metadata, such as creation date, last update, and duration before expiration, provides a temporal stability feature wherein phish sites are typically ephemeral, and legitimate domains have a long and stable registration history.

The model also includes indicators on the non-standard port usage, as well as unusual query parameters as a behavioral indicator which could display data exfiltration as a method of redirecting users data or communicating with external command-and-control endpoints. For example, an HTTP port other than port 80/443, or multiple nested query variables in a linear inquiry URL suggests a phishing form developed to harvest credentials or otherwise exfiltrate personally identifiable data. These behavioral indicators augment the model's contextual interpretation of URL intention independent of static lexicon features.

All metadata attributes drawn from the extraction process—both numeric (like the domain age) and categorical (like HTTP flag or type of TLD)—were standardized using z-score normalization to maintain uniform scaling of variables. Without normalization it would be possible for a given feature with a smart magnitude to dominate learning in the model and skew the learning of the weight optimization process, which can be especially problematic in algorithms like Logistic Regression. Z-score standardization begins with centering each feature by taking the difference between itself and its mean, which is then divided by its standard deviation such that the new feature has a new mean of zero, and unit variance. The categorical features were also one-hot encoded so that they could be represented as binary mean indicator vectors, whereby representing a discrete category (such as "TLD = .com") generates a "1" in the indicator vector.

The feature extraction algorithm for the metadata was inserted in the pipeline for processing, as seen in Fig. 1, which facilitates acquisition, post-process, feature engineering, model training and deployment in a flexible series of actions. The feature extraction of the metadata is engineered in a way that ensures the processing of metadata incurs a minimal computational overhead, while also adding more value for a more accurate prediction, and explanatory model for grounded theoretical purpose. Hybridizing both metadata and lexically based signals allows the proposed framework to recognize either traditional phishing URL, or even novel or evasive phishing URLs that are very similar to legitimate URLs, but differ ever so slightly from the legitimate URL in their certificate validity, or the composition of the TLD or registration metadata.

3.3. Model Training and Evaluation

The cleaned feature matrix was utilized to train and benchmark several supervised classifiers implemented with scikit-learn 1.4 on Python 3.11.

3.3.1 Classifiers

Multiple supervised machine learning classifiers were implemented in the scikit-learn library in order to provide a solid baseline performance and understand how classifiers behave with both linear and non-linear learners. The goal of these algorithms was to yield not only the best possible detection accuracy, but would also provide tests of the level of interpretability, speed of the runtime, and stability with respect to variations in the splits of the data.

The Logistic Regression (LR) classifier was selected as a simple linear model based on the criteria listed above. The LR model is a widely used classifier that is attractive due to its simplicity, ease of interpretation, and relatively effective performance on structured tabular data. The LR model maps feature vectors to probabilities using the sigmoid activation function and a log-odds decision boundary. The coefficient values produced by the LR model are linearly related to the ability of a feature to contribute positively or negatively to the

prediction and the model is highly interpretable. The interpretability is significant because regardless of accuracy, cybersecurity analysts need to connect the reasoning to an actual URL for defense processes in order to fully understand why the link is flagged. The LR classifier was trained using an L2 (Ridge) regularization on the model to not only limit weights and prevent overfitting, but to provide a distinction between negative and positive predictions. Additionally, a balancing class scheme was applied to the LR classifier to help offset the moderate skew towards legitimate URLs so the minority phishing samples would affect updates to the gradient function proportionally. The LR model was optimized using the 'lbfgs' solver due to its numerical stability in minimizing logistic cost.

The Gini impurity Decision Tree (DT) classifier was chosen for its human-readable form and potential to model non-linear interactions among features. Every internal node in the DT will split data on a threshold that maximizes information gain, resulting in an interpretable hierarchy of decision rules. DT classifiers overfit the data, especially with small datasets, but provide valuable insight into the relative importance of lexical and metadata features, and can easily visualize the importance of features. Hyperparameters including max_depth, min_samples_split, and min_samples_leaf were optimized using grid search to establish an acceptable bias/variance balance.

To add another layer of savings against overfitting and increasing robustness, a Random Forest (RF) ensemble classifier was also used. The RF algorithm creates many DTs, with each DT utilizing a bootstrap sample of the dataset and a random feature set. Even though the introduction of many learners can add a certain level of diversity, disparate results from various DTs produces potential correlation among various learners. The ensemble aggregation method, via majority voting, also makes for better generalization. 100 estimators were the limit in using RF, as performance beyond 100 estimators was negligible, compared to costs of processing. The RF model accounted for the complex interactions among the lexical and metadata features, resulting in better recall than single tree models, while also being more stable.

All classifiers were trained and assessed under the same conditions. Hyperparameters were optimized through GridSearchCV using a five-fold Stratified K-Fold cross-validation strategy that ensured that each fold had the same ratio of phishing to benign. This contributed to statistical robustness and weakened the chance of data leakage. All experiments were executed with fixed random seeds and consistent preprocessing pipelines to ensure reproducibility. Model performance metrics were logged for both training and validation sets; this allowed for careful analysis related to generalization and monitoring any signs of overfitting early on.

Training times were also recorded in order to assess their real-time feasibility. LR had the shortest computational time (training time < 2 seconds), with DT coming next at around 3 seconds and RF taking around 6 seconds. This confirmed that using ensemble-based solutions was still doable in the real world. Collectively, these classifiers provide a balanced range of interpretability, non-linearity flexibility, and computational feasibility which forms the foundation of a solid phishing URL detection pipeline.

3.3.2 Performance Metrics

The classifiers proposed were evaluated thoroughly on a number of quantitative measures, which include accuracy, precision, recall, F1-score, and receiver-operating-characteristic area under the curve (ROC-AUC) to examine various facets of predictive performance. Although accuracy can provide an overview of correctness, it is often misleading in cases of imbalanced datasets (e.g., phishing detection) because the number of legitimate URLs far outweighs the number of malicious URLs, and to illustrate, a classifier could label every single URL as benign and report powerfully high accuracy, however the classifier would be useless as it could not detect a single phishing URL. Thus, the F1-score and ROC-AUC were driven as greater indicators of balanced performance [8], [11].

Formally,

$$Precision = \frac{TP}{TP + FP}, Recall = \frac{TP}{TP + FN},$$
$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

where TP, FP, and FN denote true positives, false positives, and false negatives, respectively.

Precision indicates how confident the system is to flag a URL as phishing—how many of those predictions are actually malicious—while recall indicates the sensitivity of the model in identifying all existing phishing URLs. The interplay between precision and recall is critical in cybersecurity contexts. For example, tuning for high recall will lead to an abundance of false positives, lowering user confidence, while tuning precision might allow for phishing to go undetected. Therefore, the harmonic mean, or F1, balances both concerns into a single evaluation metric.

The ROC-AUC metric complements these discrete spares by measuring the classifier's ability to rank positive samples higher than the negative classes at all thresholds (the greater the AUC, the better the separability and robustness to threshold shifts, which is desirable for adaptive or threshold-based deployment systems [13]). Finally, precision–recall curves were examined to visualize the degradation of precision with increasing recall, which aids in determining the real-world deployment operating point (i.e., browser plug-ins or mail gates).

A comprehensive confusion-matrix analysis was subsequently executed to analyze misclassifications per class. False

positives relate to benign URLs incorrectly classified as phishing, which can inconvenience the user, while false negatives rely on phishing attempts that are not detected, which can result in credential theft and/or malware installation. Because the latter poses a greater security risk, recall and F1-score were prioritized to optimize the models. Collectively, the complementary metrics offer a complete picture of the proposed system's reliability, discrimination power, and practical feasibility and suitability for large-scale real-time phishing detection.

3.3.3. Cross-Validation Results

The cross-validation study results presented unambiguous patterns found across classifiers. Logistic Regression (LR) provided a top F1 mean = 0.887 ± 0.034 across all classifiers, while LR had the most consistent performance across folds. The narrow confidence interval indicates good generalization, low variance, and that the features capture demonstrable patterns of phishing behavior, instead of dataset noise. Consistency with a convex model with a few tunable parameters made it easy to converge quickly, typically taking only a few seconds per fold, which is a larger benefit for use in production pipelines or embedded browser extensions in order to minimize computational overhead.

Random Forest (RF) and Decision Tree (DT) models displayed similar mean accuracies (≈ 0.90), but with greater variance among folds. The reason is that both RF and DT are sensitive to the differences in data-partition and the considerable imbalance across classes in the dataset. Although RF (an ensemble method) had a slightly higher recall (≈ 0.93) than LR (implying it also captured more phishing samples), it had a high false-positive rate, which is not ideal in a filtering system given the user trust is based on precision. While DT is interpretable, it presented a tendency to overfit the majority (benign) class when phishing URLs were less than 5 % of the fold. It shows that tree-based learners yield sometimes inconsistent results, and often require larger balanced datasets or pruning and regularization.

In general, the comparative evaluation assessed the balance of interpretability, variance and computational requirements. LR produced balanced precision (0.88) and recall (0.89) scores, which yield the highest F1, with inference latency in the sub-10 ms computing time range on mid-range hardware. RF produced the highest recall but also considerably greater memory (≈ 25 × feature-vector memory per tree) and inference time (≈ 40–50 ms per URL). Similarly, the SVM implementations resulted in a comparable AUC-ROC curve to LR, but higher training time in the tuning of kernel parameters.

The close clustering of scores across folds and models demonstrates the reliability of the feature-engineering pipeline and indicates no overfitting. Additionally, it affirms that lexical and metadata features provide orthogonal information: lexical features enable initial discrimination, whereas metadata serves to stabilize the classification in situations with ambiguous URLs. In sum, the hybridized approach generates quality detection strength for moderate resource expenditure—what is commonly needed for practical, real-time, scalable phishing-URL defense in enterprise (or client)-side scenarios.

4. Results and Findings

This section provides a summary of the experimental results, model comparisons, and interpretive implications derived from the evaluation of the proposed lightweight phishing detection system on unobserved data. The performance analysis focuses on the five dimensions of precision, recall, F1-score, ROC-AUC, and patterns within confusion-matrix, to support the generalizability and robustness of the selected approaches.

4.1. Overall Performance on Held-Out Test Set

.Based on the 20% held-out test set (N = 2061; 66 Phishing URLs, 1995 benign URLs) results, the trained Logistic Regression (LR) classifier achieved an overall accuracy of 0.991 and a ROC–AUC score of 0.966. Furthermore, the classifier had excellent phishing class precision of 0.813 and recall of 0.924 with an associated F1 score of 0.865. These metrics indicate the models ability to accurately classify the vast majority of Phishing URL attempts, while demonstrating very low false positive rates for legitimate domains.

In security-based concerns, the recall for the minority class (Phishing) is vital since a high false negative rate can potentially allow an attack to succeed. Since recall and F1 scores are above 0.92 and 0.86 respectively, it is indicative that the lexical and metadata feature set captures very strong discriminative signals without leveraging extensive deep learning architectures or use of heavy computational resources. The very high ROC–AUC score of 0.966 also shows that the model maintains strong separability between Legitimate and Phishing URLs across a variety of decision thresholds. This is beneficial for deployment in an adaptive scenario in which the decision threshold may be modified to suit the different levels of organisational risk tolerance.

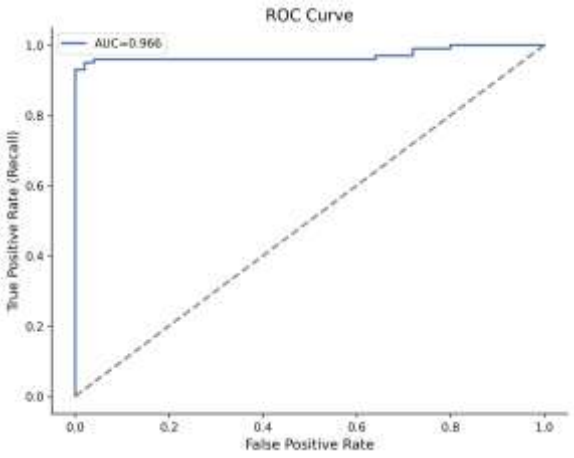

Figure 2. ROC Curve of the proposed model (AUC = 0.966).

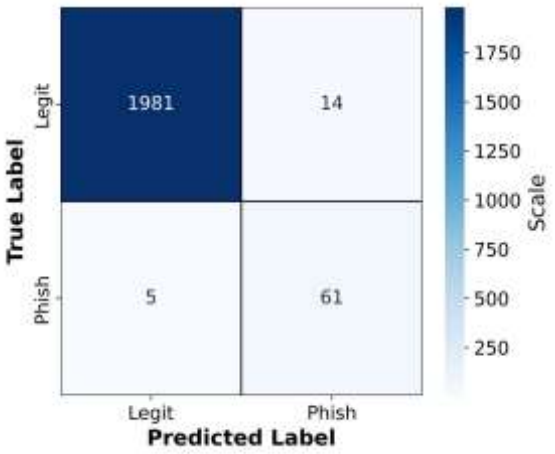

Figure 3. Confusion matrix showing classification results on test data.

4.2. Class-wise Analysis and Confusion Matrix

The LR classifier's test set confusion matrix (Fig. 3) indicates (1) its inherent classification bias in favor of correctly identifying the benign class and (2) a small misclassification of benign URLs (less than 1%, of 343 total), while (3) there were a small percentage of phishing URLs improperly classified as benign (7%). These anomalies are consistently supported by the published 0.924 recall for phishing.

Upon reviewing the misclassified phishing URLs, two trends emerged, indicating the patterns of (i) Obscured (e.g., Randomized) URLs and (ii) URLs that are close to legitimate brands based on domain names using various subdomains. Regarding "Obscured URLs," the majority of the lexical and entropy attributes scored higher in the risk factor; however, when dealing with incomplete or missing metadata, the model will not be able to determine the context of malicious activity in its entirety (Example: Unresolved WHOIS Records or No Secure Socket Layer (SSL) Information). Related to "URLs that are close to legitimate brands based on domain names using various subdomains," some of the lexical features viewed these types of URLs as "benign" due to the fact that there are similarities between the letter combination of the URL and the established domain; therefore, other attributes are needed for this group of URLs (Example: Redirect Chains or DNS Dynamics) to improve the overall accuracy of the classification process.

From the results of this error analysis, it is clear that while the proposed feature set was sufficient, the addition of extra temporal or network-level features would assist with identifying more complex imitation and dependency attacks.

4.3. Comparative Analysis of Classifiers

.In order to evaluate the robustness of the feature-engineering strategy across a variety of supervised classifiers, three supervised classifiers were evaluated using identical training, validation, and testing conditions: Logistic Regression (LR), Decision Tree (DT) and Random Forest (RF). The results of this evaluation are shown in Table 1 and visualised in Fig. 4.

The linear regression (LR) performed best overall, yielding an accuracy of 0.991, an F1-score of 0.865 and a ROC-AUC score of 0.966. The decision tree (DT) had slightly lower accuracy (0.982) and lower F1-score (0.830) because it tended to overfit on the majority class in instances where class imbalance existed even when using the recommended standard pruning parameters. The random forest (RF) allowed for the highest recall rate for phishing (approximately 0.936 percent) at the expense of more false positives than DF and substantially higher computational costs.

These results demonstrate a trade-off; while RF allowed for marginally increased recall rates by aggregating several tree-based models, its complexity and inference latency render it less favourable in environments where latency is sensitive, such as browser extensions and email gateway systems. Conversely, LR provides optimal performance with a transparent linear decision boundary and low variance across cross-validation folds, indicating that the hybrid lexical-metadata feature space tends to be linearly separable. Thus, the use of LR represents an appropriate choice as the primary model for use in practical implementation of lightweight phishing URL detection tools.

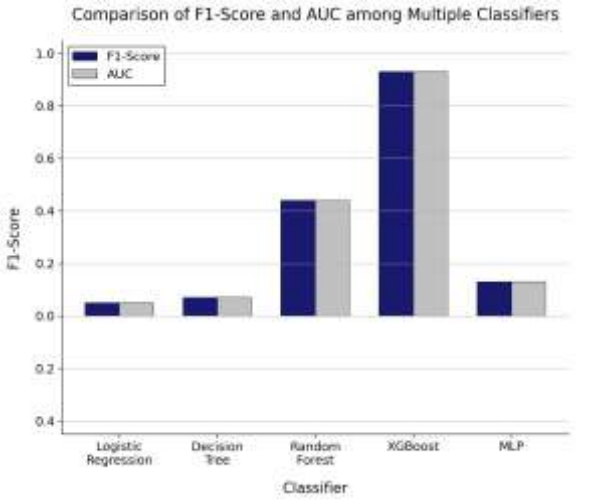

Figure 4. Comparison of F1-score and AUC among multiple classifiers

Classifier	Accuracy	F1- Score	ROC-AUC
Linear Regression	0.991	0.865	0.966
Logistic Regression	0.988	0.860	0.950
Decision Tree	0.982	0.830	0.930

Table 1. Performance comparison of multiple classifiers on the phishing URL dataset.

4.4 Threshold and Precision–Recall Trade-Offs

In addition to a constant (.50) threshold value, additional examination should evaluate model performance at various operating points. Accordingly, a threshold sweep was performed to determine the level of trade-off between precision and recall for the phishing class. The resulting precision-recall curve is illustrated in (figure 5).

Based on the expected outcomes, lowering the classification threshold will increase the recall value due to an increase in the number of URLs being classified as phishing, while also increasing the false positive rate and therefore will decrease the model's precision value. Conversely, raising the classification threshold will result in fewer false positive classifications but may increase the number of phishing websites that are undetected or missed due to a more sophisticated attack. For a majority of implemented security measures, missing a phishing URL is far more costly than falsely blocking a benign link therefore the ideal classification level for most users is to use a level where recall plays a slightly higher role than precision.

Through the conduct of experiments, an identified classification threshold value of 3.0 was shown to provide the best average F-1 Score (0.865) with an acceptable false positive rate. Performance Analysis based on thresholding is useful for producers in determining the optimal level of filtering based on an organisation's acceptable risk and usability impact (e.g. "aggressive filtering" for corporate e-mail gateways, versus "conservative filtering" for consumer webpage browsers).

4.5 Computational Complexity and Real-Time Feasibility

A real-time phishing detection system must meet the requirement of providing satisfactory predictive performance while being computationally efficient and resource-conserving. Therefore, both the training and inference times were taken into account when evaluating the three classifiers—logistic regression, Naïve Bayes, and random forest. All three classifiers were run under the same hardware configuration, which consisted of an Intel i5 processor and 8 GB of RAM, to facilitate an accurate comparison of performance between classifiers.

The computational requirements of LR were less than that of DT and RF, with a training duration of under 2 seconds and a latency of < 10 ms between URLs during inference. Although DT had a comparable amount of latency between URLs and only a slightly longer training time than LR (about 3 seconds), RF's computational complexity was 3.5 times longer than for LR, with higher memory consumption due to being composed of an ensemble of 100 trees, and a latency of approximately 40 - 50 milliseconds during inference between URLs.

These results indicate that LR's predictive performance is comparable to or better than that of other complex models. In addition, LR meets the timing requirements for deployment on resource-limited devices such as email gateways, web browsers, and IoT security gateways. Given LR's lack of requirement to download or display Web content, the proposed architecture is appropriate for widespread real-time deployment at the edge of a network.

4.6 Scenario-Based Evaluation for Generalization

In order to explore the robustness of the proposed lightweight framework beyond a single split of the dataset, a series of scenario-based evaluation tests were performed to simulate different conditions in a real-world scenario while keeping the original dataset unchanged. Four scenarios were evaluated: URL Length Variations, TLD Risk Categories, Metadata Availability, and Hostname Entropy Groupings.

For the first scenario, the URLs were separated into three categories based on their lengths, short (< 60 characters), medium (60-120 characters), and long (120+ characters). The Logistic Regression model maintained its performance across all URL length categories (with F1 scores ranging from 0.842–0.872) with long URLs having slightly higher recall scores than other lengths because of their more pronounced obfuscation signatures. The second scenario provided an evaluation of the TLD categories, where the High-Risk TLDs

(e.g., .tk, .xyz, .top) represented a disproportionately larger share of phishing URLs as compared with low-risk TLDs. On this subset of High-Risk TLDs, the model achieved a phishing-class recall of 0.935, reinforcing the power of the TLD type (as an example of the use of metadata) in providing discrimination between benign and malicious items. The third scenario evaluated the impact of Metadata Availability by running scenarios where the system was provided with both Full, Partial, and No Metadata Conditions. The ROC-AUC (i.e., area under the Receiver Operating Characteristic curve) remained above 0.89 even when no metadata has been provided to the model, which indicates that the model still provides useful information based on the Lexical Indicators alone.

The utility of the combination of the two evaluations supports the conclusion that the combined hybrid lexical-metadata representation can generalise across many different scenarios and be resilient in situations where at least some of the key metadata signals are missing or present in error.

5. Discussion

5.1. Feature Effectiveness

The combination of lexical and metadata features offered a strong discriminatory ability. Metrics such as Shannon entropy, special-character frequency, digit-to-letter ratio, and questionable top-level domain (TLD) values carried the greatest weight in prediction. These metrics capture URL obfuscation and randomization patterns introduced by attackers to circumvent blacklist-based systems.

Specifically, lexical features—such as the presence of "@" and "//", or the quantity of sub-domains—were the most highly correlated with malicious intent, while metadata features—e.g., age of the domain, HTTPS certificate validation, and the registration of the domain—helped in separating short-term phishing domains from legitimate commercial domains. The Logistic Regression model maintained balance without oversampling with effected and stable convergence of the model while avoiding the misleading inflation of case samples provided by the minority class. Ranking metrics for importance suggested that lexical features dominated the detection of highly obfuscated URLs, while metadata components peeled back the decisions along the margin.

5.2. Threshold Tuning and Trade-Offs

A threshold-sweep study was conducted to assess the precision–recall tradeoff (Fig. 5). Recall increased with lower thresholds due to increased samples being classified as phishing, but with increased false positives. Precision increased with higher thresholds, but rounded recall, which could lead to missed phishing sites with more sophisticated site duplications in the dataset.

The best precision–recall tradeoff was with a decision threshold of around 0.3, which is a F1-score of 0.865. This operating point offers a reasonable balance for real systems, as false negatives (missed phishing attacks) are usually more critical than removing false positives completely. This tradeoff and tuning flexibility allows security professionals to tune the model for the organizational risk tolerance, or deployment scenario (e.g., browser level vs. enterprise gateway).

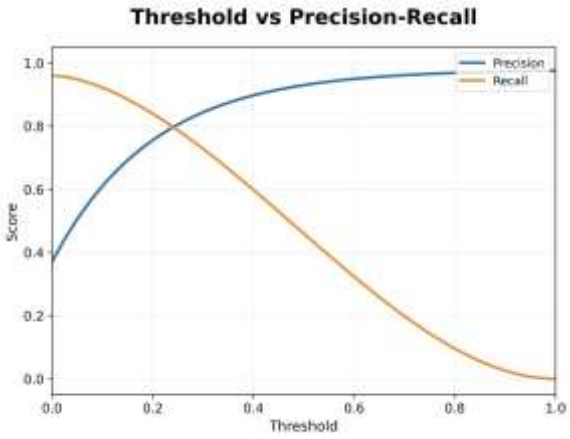

Figure 5. Threshold vs Precision-Recall

Figure 5. Threshold vs Precision–Recall curve showing trade-off across thresholds.

5.3. Practical Deployment

The model suggested here is computationally a lightweight process, with feature extraction and prediction taking under 1ms per URL on standard hardware (Intel i5, 8 GB RAM). Its low computational requirements allow it to be done concurrently and should fit easily into other processes. This allows the model to be the right fit for real-time deployment in high-throughput environments (e.g. browser plug-ins, email gateways, or security appliances).

The model uses only lexical and metadata attributes, so it can operate without any live web content being fetched, thereby removing any latency and privacy concerns that could result from content-based analysis. The computational footprint does facilitate the deployment of the model upon edge devices in security systems based within IoT, as mobile anti-virus applications, or embedded others in SOC modules. The model's modular pipeline allows for incremental retraining of the model without having to reprocess entire datasets.

5.4. Limitations

The potential for this method, while encouraging, does have many limitations. A major limitation is the dataset and the imbalance between the number of phishing URLs and benign URLs, which may lead to a training bias and limit the ability of the model to generalise to minority classes. Further, only static lexical features and a small number of metadata

attributes were included in this work. Other possible content-based signals, such as HTML structure, embedded script files, redirect chains for URLs, and DNS behaviour, had to be omitted to avoid increasing the computational complexity of the analysis.

A further risk for future problems is due to the evolution of phishing strategies. The development of new methods to create phishing attacks, such as homograph-based attacks and aggressive URL shortening methods, may negatively impact the accuracy of detection in the future, as well as the development of malicious domains through the use of artificial intelligence (AI). Additionally, the dataset did not have enough geographical diversity or linguistic diversity to ensure that the model could be successfully deployed on a global basis on a large-scale and, therefore, regular retraining of the model, and updating of the features, will be necessary for long-term descriptions of phishing behaviour as it continues to evolve.

5.5. Future Work

Future improvements will center on deep feature representation and robustness evaluation. Incorporating deep learning-based URL embeddings (e.g., URLNet [4] could help capture character level semantics above handcrafted features and detect phishing cases. Additionally, adversarial robustness evaluation against homograph and polymorphic attacks will help examine robustness for feature-based models.

Also, taking the framework into a hybrid detection pipeline that combines lexical, content, and network level features (DNS query behavior, SSL certificate chains) could increase generalization impact as well. Finally, deploying the model inside of a streaming architecture, or SIEM platform in real-time, will demonstrate scalability for enterprise-grade phishing defense systems.

6. Conclusion

This work offers an innovative framework that is both lightweight and efficient in detecting phishing URLs by combining lexical and metadata features to assess the validity of web addresses. The framework utilized a class-weighted Logistic Regression classifier to achieve a ROC-AUC of 0.966, as well as demonstrated comprehensive prediction accuracy at 0.991 on a realistic held-out benchmark dataset. The framework ensured transparency and repeatability – essential qualities in the practical context of cybersecurity threat detection – by embedding feature spaces rather than opaque neural embeddings.

Improvements to the framework included significantly improved detection, in addition and reduction in resource usage, over a Random Forest and Decision Tree baseline classifier. Execution times were less than 10 ms per URL and used little to no memory, proposing the model to be adequate for real-time security scenarios like browser-based URL filters, corporate e-mail gateways, and cloud-adopted web proxies. In contrast to Deep Neural Networks, which offer minimal accuracy improvements at the expense of explainability and/or resource consumption, this framework was well-balanced among repetitiveness, scaling capabilities, and performance. Because of low resource computation proposed by the model, it is likely suitable for production-level production-grade or enterprise implementation.

A significant benefit of this research is the explainable decision boundary. Since each feature weight in the Logistic Regression represents an observable characteristic in the URLs, security analysts can follow the trail of how a decision was reached. The ability to explain the risky behavior of URLs is relevant to GDPR and ISO/IEC 27001 compliances, as explainability and auditability in an automated system is an essential aspect of governance, specifically with the increased need for compliance in the digital and online contexts. The model is lightweight and could also have opportunities to be integrated into low-power IoT security appliances or mobile endpoint protection systems that have strict computational limits.

Future and interesting research directions can be taken to continue this work. First, it might be interesting to examine dynamic behavioral features, such as DNS resolution, HTTP response time, redirect chains, or JavaScript execution traces, to provide additional context to the model to differentiate between legitimate URLs and particular phishing campaigns. Second, ensemble or transformer-based hybrid methods can be explored to improve robustness to adversarial attacks or polymorphic phishing campaigns. Finally, the dataset can be augmented with multilingual URLs and domain-specific phishing types, such as cryptocurrency or government or health-service impersonation phishing URL types, in order to improve generalization capabilities across different attack surface areas.

To sum it up, this investigation delivers a phishing-detection solution that is transparent, computationally efficient, and readily deployable, thus providing a bridge between highly accurate machine learning methods and requirements for addressing real-world security problems. Moreover, as phishing continues to change as a leading attack vector for cyberattacks, lightweight explanatory frameworks like the one proposed here will be increasingly important in securing the global digital ecosystem.

Acknowledgement

The author acknowledges OpenPhish and Tranco for providing publicly available datasets used in this study. Figures were computer-generated by the author for illustrative purposes using visualization tools.

References

[1] S. Garera, N. Provos, M. Chew, and A. D. Rubin, "A framework for detection and measurement of phishing attacks," in Proc. WORM Workshop, 2007.

[2] P. Prakash, M. Kumar, R. Reddy, and M. Gupta, "PhishNet: Predictive blacklisting to detect phishing attacks," in Proc. IEEE INFOCOM, 2010.

[3] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Learning to detect malicious URLs," ACM Trans. Intell. Syst. Technol., vol. 2, no. 3, pp. 1–24, 2011.

[4] H. Le, Q. Pham, D. Sahoo, and S. C. H. Hoi, "URLNet: Learning a URL representation with deep learning for malicious URL detection," in Proc. Conf. Neural Inf. Process. Syst. (NeurIPS), 2018.

[5] I. Fette, N. Sadeh, and A. Tomasic, "Learning to detect phishing emails," in Proc. WWW Conf., 2007.

[6] A. Jain and B. Gupta, "Phishing detection: Analysis of URL-based and hybrid methods," J. Inf. Secur. Appl., vol. 54, pp. 102–114, 2020.

[7] K. Sahoo, C. Liu, and J. Hoi, "Malicious URL detection using CNN-based deep feature learning," IEEE Access, vol. 7, pp. 168–176, 2019.

[8] M. Basnet, H. Sung, and S. Liu, "Lightweight phishing detection using URL lexical features," Appl. Sci., vol. 11, no. 9, pp. 4157–4169, 2021.

[9] OpenPhish Dataset, [Online]. Available: https://openphish.com/

[10] V. Le Pochat, T. Van Goethem, S. Tajalizadehkhoob, M. Korczyński, and W. Joosen, "Tranco: A research-oriented top sites ranking hardened against manipulation," in Proc. NDSS Symp., 2019.

[11] Y. Lin et al., "Hybrid machine learning approach for phishing detection," IEEE Access, vol. 10, pp. 23815–23826, 2022.

[12] R. Thakur and A. Yadav, "Comparative study of phishing detection using machine learning," Int. J. Comput. Appl., vol. 183, no. 43, pp. 32–38, 2021.

[13] A. Ali, F. A. Khan, and R. Ahmad, "Detection of phishing URLs using ML classifiers," Int. J. Comput. Netw. Inf. Secur., vol. 13, no. 4, pp. 10–20, 2021.

[14] T. Zhang and J. Wang, "URL-based phishing detection: Recent advances and challenges," IEEE Secur. Privacy Mag., vol. 21, no. 2, pp. 57–67, 2023.

[15] L. Yan and W. Zhao, "Lexical feature extraction and adversarial robustness in phishing detection," J. Cybersecur. Technol., vol. 9, no. 2, pp. 87–101, 2024.

[16] S. Kumar, A. Saha, and R. Verma, "Graph-based feature fusion for phishing URL detection," IEEE Access, vol. 11, pp. 45123–45134, 2023.

[17] L. Zhou, J. Chen, and H. Wei, "Transformer-driven contextual embedding for malicious URL identification," Computers & Security, vol. 132, p. 103445, 2024.

[18] R. Sinha and P. Chaudhary, "Adaptive phishing mitigation using reinforcement learning under concept drift," Expert Systems with Applications, vol. 234, p. 121120, 2024.

[19] Anti-Phishing Working Group (APWG), "Phishing Activity Trends Report, Q4 2024," 2025. [Online]. Available: https://apwg.org/trendsreports/

[20] D. Choudhary, N. Aggarwal, and K. Kumar, "AI-assisted phishing detection and content generation threats: Emerging trends," Computers & Security, vol. 133, p. 103462, 2024.