JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

A DATA DRIVEN MULTI-SIGNAL FRAMEWORK FOR THE DETECTION OF PHISHING URLS

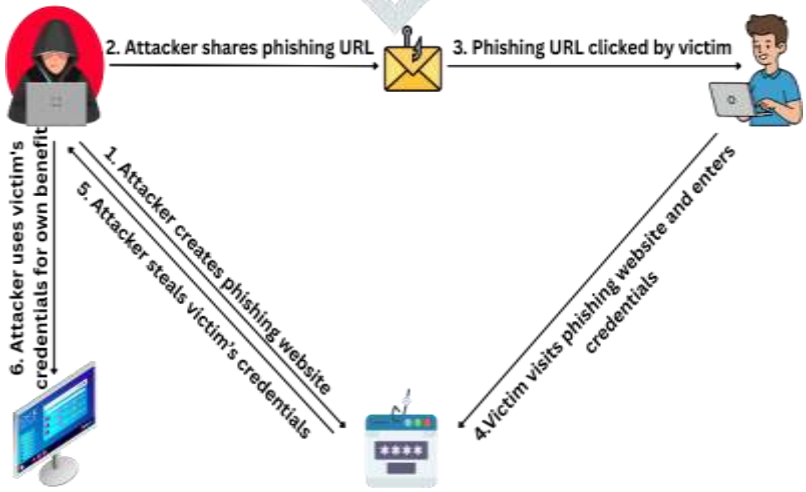
[1]Zainab, [2]Dr. Kumaraswamy S
[1]Student , [2] Assistant Professor
[1]Computer Science and Engineering
[1]University of Visvesvaraya College of Engineering, Bengaluru, India

Abstract : Phishing is one of the most effective ways to obtain passwords and perpetrate online fraud. In this paper, we present a real-time URL classification method that employs many signals to identify if a URL is phishing or secure. The classification process includes reputation checks from reliable threat intelligence sources (URLVoid, McAfee, Sucuri), confirming the SSL certificate and WHOIS information for the domain, computing topological features for each domain (domain age, IP blacklist), and finally classifying the URL using an RF classifier trained on a variety of labeled phishing datasets. To assess the system's efficacy, we collected a range of data from Phishtank and OpenPhish in addition to a few secure URLs. to evaluate the accuracy, recall, ROC-AUC, and F1 score of the system. We also looked at how obfuscation techniques impact the changing nature of phishing and how it relates to the overall resilience of our system. Based on our findings, we believe that integrating data from several sources and feature sets, rather than relying just on one source or feature set, greatly improves the ability to detect phishing. Consequently, this research develops a real-time phishing detection system that is proven, useful, and effective. Additionally, it provides a framework for identifying which features are essential for identifying possible threats and improving the capabilities of a phishing detection system, as well as supporting the need to combine several signals.

IndexTerms **- Phishing Detection, URL Classification, Threat Intelligence Aggregation, Machine Learning, Real-Time Security, LightGBM, Random Forest.**

I. INTRODUCTION

The "phishing" strategy involves hackers employing fraudulent means to get victims' email, social media, and banking credentials. Phishing enables criminals to inadvertently send users to a fraudulent website with cleverly placed links sent via various forms of communication (such as fake emails, Short Message Service (SMS), Quick Response (QR) codes, instant messaging, etc.). The characteristics of the fake websites are typically designed to mimic those of the real ones. Typically, a range of phishing technologies are used to create these fake websites [1]. Since criminals use phishing techniques to take advantage of people's trust in trustworthy businesses, consumers should be alert for signs of phishing. By identifying common phishing activities, people can strengthen their



ability to safeguard private information from potential compromise. Phishing fraudsters' sole objective is to Trick victims into divulging their login credentials so they may access their online accounts. The attackers may utilize the stolen credentials to accomplish their objectives, which might include espionage, impersonation, slander, and other financial benefits. Phishing is the most frequent cause of major cyberattacks, including ransomware and virus assaults. Fig. 1 illustrates the several improvements associated with a conventional phishing attempt.

Fig 1. Life cycle of a phishing attack.

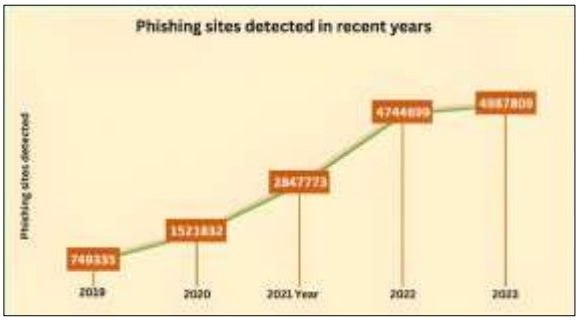


Fig 2. Phishing websites detected in last few years.

Fig 3. Phishing sites detected in last few quarters.

The Despite much research being conducted to solve this issue, the number of reported phishing websites has not decreased in recent years. The Anti-Phishing Working Group's (APWG) analysis on phishing activity trends indicates that since 2020, the number of registered phishing occurrences has grown in the post-COVID years. Fig. 2 displays phishing websites found since 2019. Fig. 3 displays several phishing websites that were reported on a quarterly basis throughout the preceding two years. 1,624,144 phishing attacks. Despite a little decline in recent quarters, the number of phishing sites discovered in each quarter is higher than the yearly records of the pre-COVID period. is ignorance. Whether conveyed by email, SMS, QR codes, or other media, the messages are intended to appear visually legitimate. Additionally, they employ misleading language designed to arouse feelings of fear, worry, tension, or haste in order to persuade users to disregard even the smallest errors in the phishing links and take swift action.

Phishing URL	Legitimate URL	Squatting Type
https://amaz0nn.asia	https://amazon.com	Typo-squatting
https://aeonbanick.com	https://aeonbank.com	Typo-squatting
https://bnl-bnpparibas.com	https://bnl-bnpparibas.com	Combo-squatting
https://steamcamunity.ru	https://steamcommunity.com	Sound-squatting
http://pankakeswape.com	http://pancakeswap.finance	Typo-squatting
https://faceb00k.homes	https://facebook.com/	Homograph-squatting

Table 1. Some examples of domain-squatting.

Phishing URL	Legitimate URL	Obfuscation Type
https://www.paypal.usid.me	https://www.paypal.com	Brand as subdomain
https://amazon.co.jp.ggbu14.com	https://amazon.co.jp	Brand as subdomain
http://pancakeswapfinance.li	http://pancakeswap.finance	Missing dot obfuscation
https://pancakeswap.services	https://pancakeswap.finance	Fake Top-level-Domain
https://bnpparibas.shop	https://bnpparibas.com	Fake Top-level-Domain
https://aw-store.ru/auth/netflix	https://netflix.com	Brand in path

Table 2. Some examples of URL obfuscation.

The attackers purchase domain names that mimic well-known businesses, a practice known as domain-squatting [3]. After that, they either give the victims their URL or wait for them to make a typo and go to the bogus website the attacker has set up. Spelling errors [4], bit flips [5], homophones [6], combining another phrase with the well-known brand [7], and using similar-looking characters [8] are just a few examples of the various ways that domain squatting may manifest. Table 1 displays some of the squatter phishing domains next to the domains of real organizations.

In addition to domain squatting, attackers frequently use URL obfuscation [9] to trick users into thinking harmful URLs are legitimate. One typical tactic is to keep the brand name in the URL while utilizing it as a subdomain of a phishing website. Attackers may change the top-level domain (TLD) or country code TLD (ccTLD) to misrepresent legitimacy. Another popular obfuscation approach that further misleads visitors by giving the impression of legitimacy while actually connecting to a harmful destination is inserting the brand name into the URL route or query string. Table 2 lists a few phishing domains that have been registered for URL obfuscation.

According to research [10], phishing attacks on companies have almost doubled in recent years, affecting 621 brands in July 2022. LinkedIn is the most often phished brand, and over 55% of phishing assaults leverage a credible brand to boost their legitimacy [11]. To avoid a phishing assault, customers need to be cautious and aware of the strategies employed by phishers. Because anybody can access the internet, regardless of their level of technological proficiency, this isn't always feasible. It is nearly impossible to instruct every internet user to avoid phishers. Software-based phishing detection solutions are therefore the best choices for effectively blocking phishing efforts.

II. LITERATURE SURVEY

This section reviews recent and widely adopted academic approaches for phishing website detection.

A variety of machine learning algorithms use a combination of many features (NLP based, word vector based, hybrid) to detect a zero-day phishing attack. The authors [12] produced a large dataset of 73575 URLs with an accuracy of 97.98% using the Random Forest approach and NLP features. Although this approach is quite precise, it is less dependable when examining URLs with minimal or no domain information.

This article describes a feature extraction technique employing hyperlinks to train a logistic regression classifier based on web page source code. A logistic regression classifier trained with this method obtained 98.4% classification accuracy [13]. However, because this method only uses source code analysis, it may be exploited in a number of ways, such as by changing embedded objects, favicons, and JavaScript references and connections.

A new approach to leveraging search engines to identify phishing-based websites was provided by reference number [14]. This technique searches the title, meta description, and content of a web site for any relevant identifying keywords. The weight of the relevant keywords is then calculated based on where they appear on the page using the TF-IDF values. A website may be deemed legitimate or phishing-based if its domain name appears in the top results of the selected third-party search engines. On a test sample of 200 legitimate and phishing websites, our approach yielded an accuracy of 89%. The method heavily relies on the results that third-party search engines provide.

Reference [15] proposed a lightweight technique that uses only nine URL-based criteria to quickly identify phishing websites on mobile and Internet of Things devices (a resource-constrained environment). The narrow feature set may limit the technique's practical utility, even if the authors report an average detection accuracy of 99.5%.

The authors of this study used hybrid features, which included text features, URL-based and hyperlink-based approaches, and the XGBoost classification model, to reach an accuracy rate of 96.76% based on their own dataset (reference [16]). According to Reference [17], a combination of ensembles employing Canopy-based Feature Selection, Grid Search for parameter optimization, and Cross Validation approaches produced an accuracy rate of 98.12%.

The work in [18] proposes a hybrid phishing detection method that combines the benefits of textual and visual data sources. The visual module examines the positioning, coloring, and size of logos on websites, while the textual component employs n-grams extracted from a webpage's content. Using the information from both modules, a search query is developed that may ascertain whether or not any of the top-ranking websites generate results that match the query as legitimate websites. The success rate of this method has been demonstrated to be 98.60%. Nevertheless, this is a significant step toward hybridization. There hasn't been much focus on detecting domain-squatting attacks and URL obfuscation yet. Because they are limited to employing either a reliable source of trademarked names or a blacklist when combined with those URLs that have been identified alongside an accurate DNS (domain name system) record, current identification methods depend upon databases and are therefore failing to discern novel forms of domain-squatting attacks.

A multilingual transformer-based approach suggested in [19] can produce phonetic squatting variations and find them in SSL certificates with a success rate of about 15% of all SSL certificates issued. An variant to the multilingual transformer model, but exclusive to financial domains, is a context-free system that simulates the inconsistent domain names on the banking website in [20].

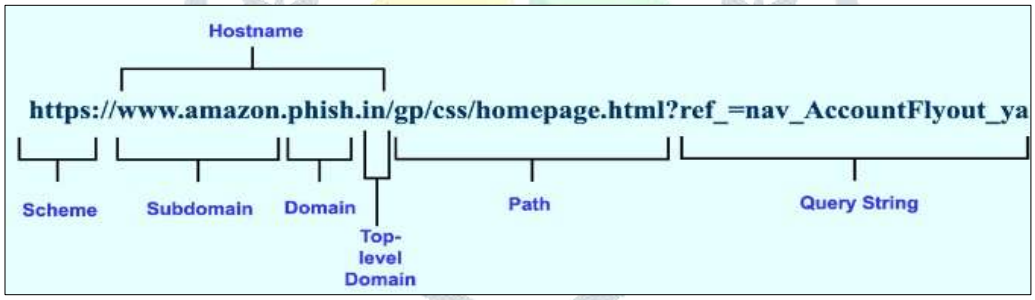


Fig 4. Components of a URL.

The study presented in [21] suggests a lightweight architecture with high computational efficiency for phishing detection on mobile devices. According to preliminary research, using an artificial neural network (ANN) model to do this task on both URL and HTML inputs yielded poor results when the URL was the only input method. For further testing, two transformer-based NLP models (BERT and ELECTRA) were trained directly on the URL input. These models were found to perform much better in terms of repeatability and generalization than previous studies using feature-based detection approaches.

III. PROPOSED MODEL

This section explains the preprocessing steps prior to inference, the extracted feature families, the FastAPI backbend's end-to-end scoring architecture, the provenance and handling of the project's URL datasets, and how the service deconstructs incoming URLs.

1) COMPOSITION OF A URL

Any A URL is the unique location associated with every web resource, including downloaded binaries, HTML pages, multimedia assets, and API endpoints. The detector divides each request into 4 canonical levels for analysis:

- The scheme encapsulates the transport protocol (http, https) and decides if TLS is expected. The parser isolates the prefix before ://; differences between scheme and certificate state immediately affect the heuristic score.
- The hostname (subdomain + registered domain + TLD) uniquely identifies the origin server. By eliminating common prefixes (www.) and standardizing case, the system can compare hosts to the Alexa Top 1M corpus, WHOIS data, temporary-domain lists, and typo-squatting variations.

- Path describes the host's directory structure. The part that comes after the hostname up to any? is used to calculate token counts, nesting depth, and suspicious folder names.
- String query Are key-value parameters added after? and in the query string, separated by &. The project counts and describes these optional arguments as lexical features, even though they often include payloads or tracking data.

To As seen in Fig. 4, the API records each URL component (scheme, hostname, path, and query) prior to feature extraction to provide auditability.

2) SYSTEM WORKFLOW

Fig. 4 depicts the whole FastAPI-based prediction pipeline used in this investigation. The URL input is obtained via either carefully selected CSV files or API requests. When the URL is received, it is first normalized (lowercase, whitespace removed, schemes inferred, and the canonical network location obtained). After URL normalization, the URL will be compared to lists of recognized benign and hazardous sources, duplicate URIs will be removed, and class imbalance in the training notebooks will be corrected. After that, a bad domain analysis will be performed on each URL. The following methods are used in the bad domain analysis to find indicators of obfuscation and domain squatting: comparing hosts that show up on the Alexa Top 1 million list; generating typo-squatting permutation URLs from the typo-squatting module using Levenshtein similarity tests; locating recently registered domains and temporary registrars using WHOIS data; and comparing FireHOL IP sets with an IP literal URL. Next, a local blacklist and several threat intelligence sources, such as URLVoid, Norton, and McAfee, are used in the reputation verification process. If a URL is found to be high-confidence malicious, it may be terminated early. URLs are compared to lexical features after going through this sequence. Based on certain mathematical computations related to the structural/statistical characteristics of the URL (length, entropy, token counts, number of digits/hyphens, density of special characters, and existence of a malicious extension), these checks are utilized to create machine learning input vectors. The RF-trained model will next use binary classification to separate the URL into two categories: genuine and phishing. The results of each test will be used to develop a comprehensive weighted scoring model. Each check will result in a penalty, with a starting score of 180. By providing users with human-readable JSON processes, this will ensure consistency and repeatability between model building and real-time deployment.

3) FEATURES EXTRACTED

3.1 URL Normalization & Domain Analysis

An algorithm called URL normalization looks at the URL and does Risk Assessment at the Domain level to find possible high-risk domains early. When the URL is received, any unnecessary spaces are removed, it is transformed to lower case, and if the URL contains no scheme, a guess about its type is automatically established. For example, this will allow the URL to be parsed into its constituent pieces in order to retrieve all of the BAD Domain indicators .Are there any names that closely resemble well-known brands that may be identified using typo-squatting permutation analysis, or are the domains on the Alexa Top 1 billion list qualified for early allowlisting? Have any Domain Registrars been classified as Temporary using WHOIS_DATA, or have any domains been formed from scratch using WHOIS_DATA's estimated Domain Age? Additionally, any uttered URLs have been matched against the FireHOL Blacklists before to determining whether to move on with additional inquiry. Each of these Domain Level Feature Set items offers incredibly potent signals to detect Brandjackers, Squares, and Infrastructure Abusers prior to further investigation.

Algorithm 1: URL Normalization & Domain Analysis

1: **Step 1**: Normalize the URL
2: **function** NORMALIZE_URL (url)
3: url ← lowercase(trim(url))
4: if scheme missing then infer scheme as http
5: parse url into scheme, hostname, path, query
6: return normalized_url
7: **end function**
8: **Step 2:** Extract bad-domain features
9: **function** EXTRACT_DOMAIN_FEATURES (normalized_url)
10: **D1** ← check Alexa Top 1M membership
11: **D2** ← compute typo-squatting similarity with known brands
12: **D3** ← check temporary registrar using WHOIS
13: **D4** ← check domain age ≥ 3 months
14: **D5** ← detect IP-literal URL & check FireHOL IP sets
15: **return** {D1, D2, D3, D4, D5}
16: **end function**

3.2 Lexical URL Feature Extraction

A technique for extracting URL attributes using lexical characteristics has been developed in order to handle both the structural and statistical aspects of URLs. When speed and dependability are considered, all of the attributes derived from this method are computed locally. The second feature-extraction algorithm generates lexical features such as total URL length, Shannon entropy of the domain to measure randomness, IP address references, flags for malicious file extensions, counts of query parameters, counts of path tokens, counts of hyphens, counts of digits, and counts of special characters like (@, #, $, %, ^, &, *, _, and +). Combining these

traits can generalize the most popular methods for manipulating and obfuscating phishing URLs. which serves as the foundation for the input vector of the machine learning algorithm.

Algorithm 2: Lexical URL Feature Extraction

```
function EXTRACT_LEXICAL_FEATURES(normalized_url)
    L1 ← compute total URL length
    L2 ← compute domain Shannon entropy
    L3 ← detect IP address usage
    L4 ← detect malicious file extensions
    L5 ← count query parameters
    L6 ← count path tokens
    L7 ← count hyphens
    L8 ← count digits
    L9 ← detect special characters (@, !, #, $, %, ^, &, *, _, +)
  return {L1…L9}
end function
```

3.3 Reputation & Transport Analysis

To We combine domain indications with additional components like language, trust signals, and transport related checks to produce an understandable outcome of phishing or not. Using prior whitelist criteria, URLs classified as "safe" will be immediately assigned to a legal category, eliminating any unnecessary evaluations. The remaining URLs are then evaluated using a weighted scoring method that combines all three categories of indicators with a trained RF Model for lexical indicators. Ultimately, a risk value and a categorization option will be offered, and any inaccurate projections or bad judgments will be penalized. By employing a range of techniques, we can provide more precise results, create more reliable systems, and allow system users to understand the reasoning behind each URL's classification. to aggregate the data.

Algorithm 3: Reputation & Transport Analysis

```
function REPUTATION_AND_SSL_CHECK (url)
    R1 ← query URLVoid
    R2 ← query Google Safe Browsing
    R3 ← query Norton Safe Web
    R4 ← query McAfee SiteAdvisor
    R5 ← query Sucuri blacklist
    R6 ← check local blacklist
    T1 ← verify SSL certificate validity
    T2 ← check HTTPS usage
     return {R*, T*}
end function
```

3.4 Phishing Classification & Scoring

To We combine domain indications with additional components like language, trust signals, and transport related checks to produce an understandable outcome of phishing or not. Using prior whitelist criteria, URLs classified as "safe" will be immediately assigned to a legal category, eliminating any unnecessary evaluations. The remaining URLs are then evaluated using a weighted scoring method that combines all three categories of indicators with a trained RF Model for lexical indicators. Ultimately, a risk value and a categorization option will be offered, and any inaccurate projections or bad judgments will be penalized. By employing a range of techniques, we can provide more precise results, create more reliable systems, and allow system users to understand the reasoning behind each URL's classification. to aggregate the data.

Algorithm 4: Phishing Classification & Scoring

```
function CLASSIFY_URL (url)
  normalized_url ← NORMALIZE_URL (url)
  {D1…D5} ← EXTRACT_DOMAIN_FEATURES (normalized_url)
    if D1 = = TRUE then
          Note URL as Legitimate (early allowlist)
          return Legitimate
    end if
  {L1…L9} ← EXTRACT_LEXICAL_FEATURES (normalized_url)
  {R*, T*} ← REPUTATION_AND_SSL_CHECK (normalized_url)
    y_pred ← RF. predict ({L1…L9})
    score ← 180
    apply weighted penalties based on D*, R*, T*, and y_pred
    if score < threshold then
```

```
        label URL as Phishing
    else
        label URL as Legitimate
    end if
    return final verdict and score
end function
```

---

4) DATASET CONSTRUCTION & LABELLING

The open phishing corpus and benign URL lists with references to the Alexa Top 1 million Domains from several CSV files included in the data/directory will be combined to generate the dataset. All URLs were normalized before processing by deleting netlocs, inferencing schemes, and changing them to lowercase. After that, duplicates were eliminated and merged from every source. Only reputation-based signals are utilized to guarantee that the URLs stay the same during processing. Every URL's final state is classified as either benign (which the source maintains) or phishing based on the source's label. Preprocessing notebooks will divide the dataset into training and validation phases and employ either sampling or weighting procedures to correct the class imbalance in order to address the dataset's prevalent benign bias. The final artifact from the model produced by RF will be utilized for deployment after processing and labeling are complete.

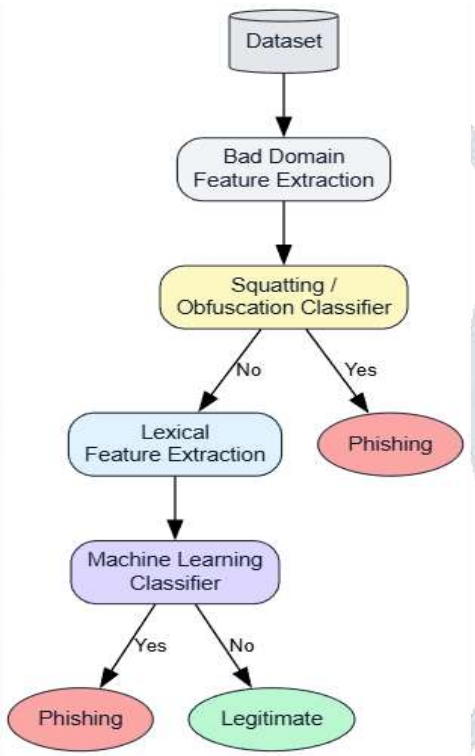


Fig 5. System workflow of the proposed approach

5) MODEL TRAINING AND HYPERPARAMETER SETUP

An RF model selected using ModelTrainin comparison tests serves as the final classifier, while a Random Forest model is maintained as a baseline. The model is trained using nine lexical URL features: URL length, entropy, IP address flag, malicious extension flag, query parameter count, path token count, hyphen count, digit count, and special-character presence. Hyperparameters are changed over learning rate, number of leaves, tree depth, minimum data in leaf, feature fraction, bagging fraction, and boosting iterations. Evaluation is carried out via held-out validation or cross-validation. RF was chosen due to its superior accuracy latency trade-off and resistance to feature interactions. The trained model is loaded for inference at runtime by the FastAPI backend.

6) COMPUTATIONAL COMPLEXITY ANALYSIS

The time required for lexical feature extraction and URL processing is proportional to the length of the URL. Domain lookups against the Alexa Top 1 million list now have linear-time complexity due to CSV scanning; however, indexed data structures can lower this to constant time. Predetermined constraints limit the production of typos and the evaluation of similarity, leading to helpful linear behavior for the number of candidates. For machine learning inference, RF's small trees and limited feature dimensionality lead to low processing costs. While fully local inference completes each URL in milliseconds, network-bound reputation inquiries account for most of the total request time.

7) THREAT MODEL AND ASSUMPTIONS

Using obfuscation techniques including IP-literal addressing, query stuffing, typo squatting, and temporary domain registration, it is anticipated that the attacker will be able to generate arbitrary URLs. The defense has access to IP blacklists, WHOIS-derived data, benign domain corpora, domain reputation services, and a trained lexical machine learning classifier.

The method assumes that WHOIS and reputation services are sufficiently accessible and uses the Alexa Top 1 million membership as a strong benign prior. Content-based attacks, compromised legitimate domains, and client-side exploitation are considered beyond the purview. Identification depends only on less trustworthy machine learning predictions and linguistic heuristics in the absence of reputation cues.

IV. RESULTS AND DISCUSSION

1) Experimental Setup

Using a range of Python machine learning libraries, such as Scikit-learn and RF, experiments with a recently developed phishing detection system based on a large publicly available dataset comprising phishing sites from multiple sources, including Kaggle, PhishTank, and OpenPhish, as well as lists of supplied benign sites, were completed. The final dataset contained about 1,541,538 unique labeled URL samples after data pre-processing, normalization, and de-duplication; of these, 0 were benign URLs and 1 were phishing URLs. To minimize class imbalances, the tagged URLs are separated into training and testing sets using standard methods, and the proper sampling and weighting strategies are used. The models were evaluated in reference to the cyber security imbalanced datasets using the following standard metrics: accuracy, precision, recall, F1 score, and ROC-AUC.

2) Dataset Statistics

The s table (Table 3) displays the dataset distribution for each experiment. The dataset offers enough amount and variation for thorough testing, including 574,556 phishing URLs and 966,982 benign URLs. As a result, it makes sense to believe that the applicability of the proposed approach will translate well to real-world phishing situations.

Category	Phishing URLs	Benign URLs	Total
Available in dataset	574,556	966,982	1,541,538

Table 3. Dataset Distribution Used for Experiments

3) Performance Comparison of ML Models

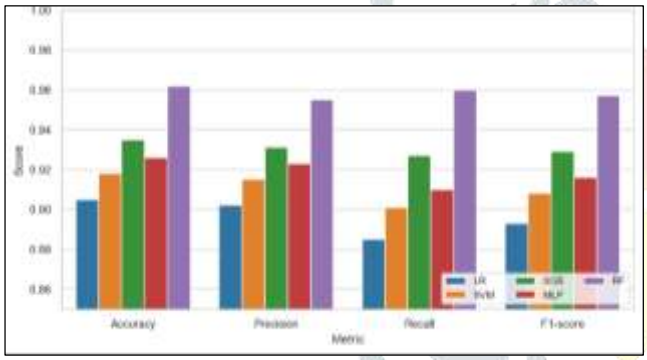


Fig 6. Before Feature Engineering.

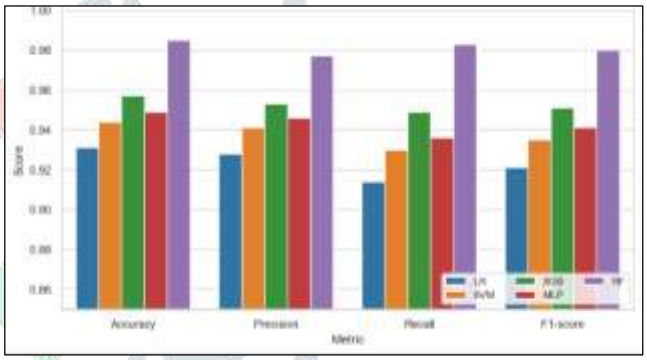


Fig 7. After Feature Engineering.

Two charts, Figs. 6 and 7, compare the various machine learning techniques used in this study to identify phishing websites. Fig. 6 compares the machine learning methods without further feature engineering or data enrichment (baseline value). The baseline values of the LR and SVM algorithms are much lower than those of the other approaches, as seen in Figure 6. The poor recall and F1 scores of both algorithms indicate that their ability to recognize complex phishing tactics was limited. However, the absence of further enriched characteristics still limited the performance of models constructed using ensemble and neural components.

As Fig. 7 shows, applying the feature engineering methods from this study consistently improved all assessment metrics on all kinds of classifiers. It is the classifier type that benefited the most from the extra features as Random Forest (RF) was the most successful classifier type and the Ensemble Based model had the highest overall performance as measured by Recall and F1 Scores. These characteristics are essential for phishing detection since there should be as few false negatives as feasible.

The results of the study demonstrate that using a multi-signal feature set enhances the model's performance and that using ensembles during real-time phish detection is the most effective way to deploy phish detection models.

4) Discussion of Results

In all evaluation metrics, ensemble-based classifiers perform better than linear models (Figures 6 and 7). The Logistic Regression and Support Vector Machine classifiers had lower Recall and F1-scores than the other classifiers, suggesting that these linear classifiers were unable to adequately capture the nonlinear and complex nature of patterns commonly associated with phishing URLs. The Random Forest Classifier outperformed all other classifiers in terms of F1-score and Recall. This is important since lowering false negatives in security applications depends on both measurements. The better scores obtained by feature engineering confirm the usefulness of the multi-signal feature sets recommended to enhance class separation.

The results suggest that integrating ensemble learning with robust feature representations results in a high-performance approach for real-time phishing URL recognition across extremely imbalanced datasets.

5) Practical Implications

The evidence presented indicates that its recommended Phishing Detection Framework will operate with real-world systems in a Real-Time Deployment Context, including email gateways, browser extensions, and network security solutions. This approach's major focus on URL-Based features and Computable at a Local Level, which increases trust in the integration of Reputation Signals into its Detection Capabilities, enables Low Inference Latency. As demonstrated by the results, the recommended approach may be used as a practical means of reducing the effect of phishing threats globally.

V. CONCLUSION

Attackers have been refining their obfuscation strategies over time. These kinds of hackers thrive in an atmosphere where people are increasingly trusting of their online experiences. In this study, we present a multi-signal system that uses lexical features, reputation-based intelligence, and domain-level analysis via domain names to identify phishing websites. We discovered that feature engineering had a significant impact on the performance of every machine learning model we examined in this research when we conducted a control experiment with around 1,500,000 phishing sites.

The Random Forest Model had the greatest recall and F1 score of all the models examined, making it the most effective method for identifying phishing attempts when reducing false negatives is crucial. These findings show that the capacity to identify fraud is significantly higher when several complimentary signals are combined than when any one feature set is used alone. The framework can achieve its goals of being a strong, understandable, and reliable product in the current market because of its suggested computational efficiency, interpretability, and ability to be implemented for real-time, practical security solutions (such as browser extensions, email gateways, and network protection systems). Future work will entail creating extensions to enable application in the areas of identifying and addressing compromised legitimate domains, in addition to improving existing adaptive learning systems to accommodate for evolving phishing strategies.

REFERENCES

[1] B. "Leaky kits: The increased risk of data exposure from phishing kits," B. Tejaswi, N. Samarasinghe, S. Pourali, M. Mannan, and A. Youssef, Proceedings of the APWG Symp. Electron. Crime Res. (eCrime), November 2022, pp. 1–13.

[2] A. Prasad and S. Chandra, ‘‘BotDefender: A collaborative defense framework against botnet attacks using network traffic analysis and machine learning,’’ Arabian J. for Sci. Eng., vol. 49, no. 3, pp. 3313–3329, Mar. 2024, doi: 10.1007/s13369-023-08016-z.

[3] K. Barik, S. Misra, and R. Mohan, ‘‘Web-based phishing URL detection model using deep learning optimization techniques,’’ Int. J. Data Sci. Anal., vol. 20, no. 5, pp. 4449–4471, Feb. 2025, doi: 10.1007/s41060-025 00728-9.

[4] APWG (2024). Report on Phishing Activity Trends. Accessed in February of 2024.

[5] G. Brezeanu, A. Archip, and C.-G. Artene, ‘‘Phish fighter: Self updat ing machine learning shield against phishing kits based on HTML code analysis,’’ IEEE Access, vol. 13, pp. 4460–4486, 2025, doi: 10.1109/ACCESS.2025.3525998.

[6] N. Kumar, S. Ghewari, H. Tupsamudre, M. Shukla, and S. Lodha, "When diversity meets hostility: A study of domain squatting abuse in online banking," in Proceedings of the APWG Symp. Electron. Crime Res. (eCrime), December 2021, pp. 1–15.

[7] Y.-M. Wang, D. Beck, J. Wang, C. Verbowski, and B. Daniels, "Strider typo-patrol: Discovery and analysis of systematic typo-squatting," SRUTI, vol. 6, pp. 31–36, March 2006.

[8] N. Nikiforakis, S. Van Acker, W. Meert, L. Desmet, F. Piessens, and W. Joosen, "Bitsquatting: Exploiting bit-flips for fun, or profit?" in Proceedings of the 22nd International Conference on the World Wide Web, May 2013

[9] Sound-skwatter (did you mean: Sound-squatter?) AI-powered generator for phishing prevention," by R. Valentim, I. Drago, M. Mellia, and F. Cerutti 2023, arXiv:2310.07005.

[10] P. Kintis, N. Miramirkhani, C. Lever, Y. Chen, R. Romero-Gómez, N. Pitropakis, N. Nikiforakis, and M. Antonakakis, "Hiding in plain sight: A longitudinal study of combosquatting abuse," in Proceedings of the ACM SIGSAC Conf. Comput. Commun. Secur., October 2017, pp. 569–586.

[11] G. S. Nayak, B. Muniyal, and M. C. Belavagi, ‘‘Enhancing phishing detection: A machine learning approach with feature selection and deep learning models,’’ IEEE Access, vol. 13, pp. 33308–33320, 2025,

[12] S. Remya, M. J. Pillai, B. S. Aparna, S. Rama Subbareddy, and Y. Y. Cho, ‘‘BGL-PhishNet: Phishing website detection using hybrid model-BERT, GNN, and LightGBM,’’ IEEE Access, vol. 13, pp. 2025, doi: 10.1109/ACCESS.2025.3551542.

[13] F. Quinkert, T. Lauinger, W. Robertson, E. Kirda, and T. Holz, "It's not what it looks like: Measuring attacks and defensive registrations of homograph domains," in IEEE Conf. Commun. Netw. Secur. (CNS), June 2019, pp. 259–267.

[14] H. Tupsamudre, A. K. Singh, and S. Lodha, "Everything is in the name—AURL based approach for phishing detection," in Proceedings of the International Symposium on Cybersecurity, Cryptography, and Machine Learning, 2019, pp. 231–248.

[15] Statista (2009). Phishing attacks targeted legitimate brands between January 2009 and October 2022. February 2024. of brands-hijacked-by-phishing-attacks/

[16] StationX (2024). Top Phishing Statistics for 2024: Current Data and Patterns. [Online] Accessed: February 2024.

[17] R. Goenka, M. Chawla, and N. Tiwari, "A comprehensive survey of phishing: Mediums, intended targets, attack and defense techniques and a novel taxonomy," Int. J. Inf. Secur., vol. 23, no. 2, pp. 819–848, April 2024

[18] O.K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from URLs," Expert Syst. Appl., vol. 117, pp. 345–357, March 2019.

[19] B. B. Gupta, K. Yadav, I. Razzak, K. Psannis, A. Castiglione, and X. Chang, ‘‘A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment,’’ Comput. Commun., vol. 175, Jul. 2021.

[20] A. K. Jain and B. B. Gupta, ‘‘A machine learning based approach for phishing detection using hyperlinks information,’’ J. Ambient Intell. Humanized Comput., vol. 10, no. 5, pp. 2015–2028, May 2019.

[21] A.Aljofey,Q.Jiang,A.Rasool,H.Chen,W.Liu,Q.Qu,andY.Wang,‘‘An effective detection approach for phishing websites using URL and HTML features,’’ Sci. Rep., vol. 12, no. 1, May 2022, Art. no. 8842.

[22] A. K. Jain, S. Parashar, P. Katare, and I. Sharma, ‘‘PhishSKaPe: A content-based approach to escape phishing attacks,’’ Proc. Comput. Sci., vol. 171, pp. 1102–1109, Jan. 2020.

[23] C. C. L. Tan, K. L. Chiew, K. S. C. Yong, Y. Sebastian, J. C. M. Then, and W.K.Tiong, ‘‘Hybrid phishing detection using joint visual and textual identity,’’ Expert Syst. Appl., vol. 220, Jun. 2023, Art. no. 119723.

[24] R. V. Valentim, I. Drago, M. Mellia, and F. Cerutti, ‘‘X-squatter: AI multilingual generation of cross-language sound-squatting,’’ ACM Trans. Privacy Secur., vol. 27, no. 3, pp. 1–27, Aug. 2024, doi: 10.1145/3663569.

[25] A. Karim, M. Shahroz, K. Mustofa, S. B. Belhaouari, and S. R. K. Joga, ‘‘Phishing detection system through hybrid machine learning based on URL,’’ IEEE Access, vol. 11, pp. 36805–36822, 2023.

[26] N. Kumar, S. Ghewari, H. Tupsamudre, M. Shukla, and S. Lodha, ‘‘When diversity meets hostility: A study of domain squatting abuse in online banking,’’ in Proc. APWG Symp. Electron. Crime Res. (eCrime), Dec. 2021, pp. 1–15.

[27] A. Costello. (2003). Punycode: A Bootstring Encoding of Unicode for Internationalized Domain NamesinApplications(IDNA).IETF.Accessed: Jan. 20, 2024. [Online]. Available: https://tools.ietf.org/html/rfc3492

[28] T. Koide, N. Fukushi, H. Nakano, and D. Chiba, ‘‘PhishReplicant: A language model-based approach to detect generated squatting domain names,’’ in Proc. Annu. Comput. Secur. Appl. Conf., Dec. 2025, pp. 1–13.

[29] K. Haynes, H. Shirazi, and I. Ray, ‘‘Lightweight URL-based phishing detection using natural language processing transformers for mobile devices,’’ Proc. Comput. Sci., vol. 191, pp. 127–134, Jan. 2021.

[30] A. Prasad, S. Chandra, M. Uddin, T. Al-Shehari, N. A. Alsad han, and S. Sajid Ullah, ‘‘PermGuard: A scalable framework for Android malware detection using permission-to-exploitation mapping,’’ IEEE Access, vol. 13, pp. 507–528, 2025, doi: 10.1109/ACCESS.2024.3523629.A. Prasad, S. Chandra, M. Uddin, T. Al-Shehari, N. A. Alsad han, and S. Sajid Ullah, ‘‘PermGuard: A scalable framework for Android malware detection using permission-to-exploitation mapping,’’ IEEE Access, vol. 13, pp. 507–528, 2025, doi: 10.1109/ACCESS.2024.3523629.