Istanbul University-Cerrahpasa
Introduction to Machine Learning
Final Project Report

Group ID: 10
Mustafa Çirci 1306210018
Ahmet Oğuz 1306210064

January 27, 2026

1 Problem Justification and Importance

1.1 Why is this problem important?

Phishing is one of the most common and dangerous cyber threats today. The main goal of these attacks is to trick users into visiting fake websites to steal passwords, credit card numbers, or personal data. This problem affects everyone, from individual users to large companies and banks. The importance of solving this problem comes from three main realities:

- **Global Security Impact:** Phishing is not just a minor technical problem; it is a serious form of cybercrime. Global cybersecurity organizations such as the Anti-Phishing Working Group (APWG) report [1] that phishing attacks increase every year, especially after the widespread adoption of remote work. This trend shows that attackers continuously adapt and find new ways to exploit users.
- **Visual Similarity Tricks:** Phishing URLs are carefully designed to look almost identical to legitimate ones. As shown in the study by Prasad and Chandra [2], attackers use small visual changes, such as replacing characters or using look-alike symbols, which are very difficult for humans to notice. This makes manual detection unreliable and highlights the need for automated solutions.
- **Cost of Errors:** Wrong predictions have serious consequences. If a phishing URL is not detected, sensitive data may be stolen and financial losses can occur. Conversely, blocking a legitimate website can disrupt normal business activities and reduce user trust. Therefore, an effective detection system must carefully balance security and usability.

Because phishing targets human weaknesses rather than just software bugs, traditional security measures are often not enough.

1.2 Why is Machine Learning necessary?

We cannot solve this problem with simple lists or basic rules anymore. Here is why Machine Learning is the necessary solution:

- **Old Methods are Too Slow:** Traditional systems use "blocklists" to keep list of known bad sites. However, attackers create new phishing sites every minute. A static list cannot stop a website that was created just five minutes ago. ML models can analyze the website's structure and detect it, even if it is brand new.
- **Complexity of Attacks:** Attackers use many complex tricks, such as URL shortening or hiding malicious codes inside innocent-looking links. It is impossible for a human to write manual "if-else" rules for every possible trick. ML algorithms can learn these complex patterns automatically.
- **The Problem of Scale:** There are billions of websites on the internet. It is impossible for human experts to check every single URL one by one. Machine Learning is fast and can scan millions of links in seconds, providing real-time protection for everyone.

For these reasons, Machine Learning provides a dynamic and scalable solution that adapts to new threats much better than old manual methods.

1.3 Related Works

1.3.1 Urlnet: Learning a Url Representation with Deep Learning for Malicious Url Detection [3]

In this article, they proposed URLNet, a deep learning framework using Convolutional Neural Networks to detect malicious URLs without manual feature engineering. They trained their model on a massive dataset of 15 million URLs collected from VirusTotal. While they achieved a 99.29% AUC score, they noted that such models require substantial computational power to process millions of parameters. While this approach is practical for large scale industrial systems with sufficient hardware resources, in this assignment we considered hardware limitations and project constraints, and therefore aimed to obtain the most effective results within a restricted computational setting.

1.3.2 Malicious Url Detection Using Machine Learning: A Survey [4]

In this article, they reviewed over 100 different studies and techniques for malicious URL detection and analyzed various algorithms, including SVM and Online Learning, across datasets like DMOZ and PhishTank. Their analysis concluded that feature-based machine learning, which relies on extracting attributes like URL length and special characters, remains the most robust approach for real-time systems. This study supports our choice of using manual feature extraction instead of raw text processing within the scope of this project.

1.3.3 Web Phishing Net (WPN): A Scalable Machine Learning Approach for Real-Time Phishing Campaign Detection [5]

They addressed the emerging threat of AI-generated phishing URLs in a 2025 study. They used the PhishStorm dataset combined with phishing URLs generated by GPT-3 to test

a lightweight clustering model called Web Phishing Net. They achieved a 97.9% detection rate on AI-generated threats with a processing time of just 192 ms, outperforming complex alternatives. Considering the scope and constraints of this assignment, these findings are consistent with our choice of Random Forest due to its efficiency and competitive accuracy.

2 Dataset and Target Variable

2.1 Dataset Selection

For this project, we selected the PhiUSIIL Phishing URL Dataset [2] from the UCI Machine Learning Repository. This dataset is designed to detect advanced attacks like "combosquatting" and "typosquatting."

- **UCI Source Link:** `https://archive.ics.uci.edu/dataset/967/phiusiil+phishing+url+dataset`
- **Donation Date:** 3/3/2024.
- **Total Instances:** 235,795 samples. 134,850 legitimate and 100,945 phishing URLs.
- **Total Features:** 54 columns.
- **Data Type:** Tabular / Multivariate.
- **Target Variable:** Class 1 Legitimate, Class 0 Phishing.

2.2 Feature Description

The dataset consists of 54 features extracted from both the URL string and the webpage source code. We have categorized them into three functional groups:

2.2.1 URL-Based Features

These features are derived strictly from the URL string without accessing the page content.

- **Length Attributes:** Includes `URLLength`, `DomainLength`, and `TLDLength`. Phishing URLs are often intentionally long to hide the actual domain name from users on mobile devices.
- **Special Character Counts:** Attackers often use special characters to obfuscate the URL or pass malicious parameters. `NoOfQMarkInURL` (?), `NoOfEqualsInURL` (=), `NoOfAmpersandInURL` (&), and `NoOfOtherSpecialCharsInURL`.
- **Structural Ratios:** To detect machine-generated URLs, we analyze density metrics such as `LetterRatioInURL`, `DegitRatioInURL`, and `CharContinuationRate`.
- **Domain Identity:** `IsDomainIP` checks if IP is used instead of domain and `NoOfSubDomain` looks URL has any subdomain or not
- **Similarity and Obfuscation:** This category captures deceptive tactics, utilizing `URLSimilarityIndex` to measure visual resemblance to popular brands, while `HasObfuscation` and `ObfuscationRatio` quantify the use of hidden or encoded characters designed to bypass security filters.

2.2.2 Content-Based Features

These features are extracted by analyzing the HTML source code of the webpage.

- **Page Resources:** Legitimate sites usually have rich content. We count `NoOfImage`, `NoOfCSS`, `NoOfJS`, and `NoOfiFrame`. Phishing sites are often simple copies with low resource counts.
- **Code Structure:** Attackers often use messy or compressed code. We analyze `LineOfCode` and `LargestLineLength` to detect obfuscated scripts.
- **Metadata:** This category assesses page credibility by verifying the presence of standard identity elements like `Title` and `HasFavicon`, while utilizing the `DomainTitleMatchScore` to measure the consistency between the domain name and the webpage title.
- **Technical Properties:** `Robots` presence of robots.txt and `IsResponsive` checks mobile compatibility.

2.2.3 Security and Behavioral Features

These features analyze how the site interacts with users and sensitive data.

- **Sensitive Data Collection:** Phishing sites exist to steal data. Features include `HasSubmitButton`, `HasPasswordField`, and `HasHiddenFields`.
- **Redirection Behaviors:** Excessive redirections are used to hide the final destination. We track `NoOfURLRedirect`, `NoOfSelfRedirect`, and `NoOfEmptyRef`.
- **Target Keywords:** The presence of high-risk keywords in the HTML such as `Bank`, `Pay`, and `Crypto`, which indicate a financial target.
- **Security Protocol:** `IsHTTPS` checks for SSL encryption.

3 Methodology

3.1 Data Understanding and Preprocessing

- **Irrelevant Columns:** We removed the FILENAME, URL, and Domain columns from the dataset. Since the dataset already contains extracted features that capture the structural properties of these texts, keeping the raw versions was unnecessary.
- **Encoding Categorical Data:** We converted the Top Level Domain column into numerical values using the Label Encoding method. This step was necessary because machine learning algorithms require numerical input to process categorical data effectively.
- **Data Leakage:** We noticed that the URLSimilarityIndex feature gave away the answer too easily and caused data leakage. We decided to remove this feature manually to ensure our model actually learns to detect phishing patterns instead of just memorizing this score.

- **Feature Scaling:** We applied the Standard Scaler to normalize all numerical features. This makes sure that large numbers do not dominate the calculation, which is very important for algorithms that rely on distance like KNN.

- **Feature Selection:** We analyzed the relationships between features using a correlation matrix to find repetitive information. When we found two features with a correlation higher than 0.75, we removed one of them to make the model faster and more efficient. As a result of this analysis, we removed ObfuscationRatio, NoOfLettersInURL, NoOfDegitsInURL, NoOfEqualsInURL, NoOfAmpersandInURL, NoOfOtherSpecialCharsInURL, and URLTitleMatchScore features from the dataset.

- **Removing Duplicates:** After removing specific features, some rows became identical. We removed these duplicate samples to prevent data leakage and to ensure that the training and testing sets remain distinct.

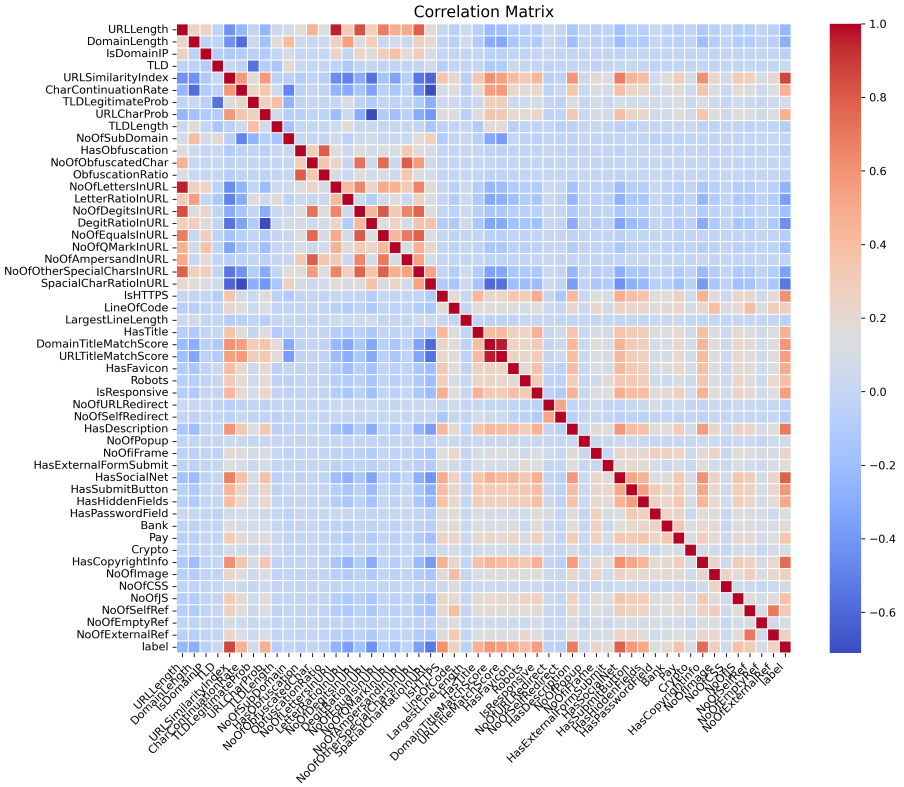

Figure 1: Correlation Matrix of Features

3.2 Modeling Strategy

- **Data Splitting:** We split the dataset into 80% training and 20% testing sets. Also, we used stratified splitting to ensure that the ratio of phishing to legitimate URLs remained consistent in both sets.

- **Baseline Model:** To see balance of dataset and minimum performance of benchmark, we used a DummyClassifier.

- **Model Selection:** We implemented four distinct algorithms to cover different learning paradigms:
  - **Logistic Regression:** We chose this model to test if the problem is linear. It serves as a simple baseline to see if we really need complex models.
  - **K-Nearest Neighbors:** By selecting this algorithm, we aimed to observe if URLs with similar feature vectors belong to the same class.
  - **Gaussian Naive Bayes:** We included this model to test how the model performs if we assume features are independent.
  - **Random Forest:** We selected this model to capture complex patterns that simple models miss. It is also very resistant to overfitting.
- **Hyperparameter Tuning and Validation:** We employed GridSearchCV with 5-fold Stratified Cross-Validation to optimize the Random Forest model. We selected this strategy because it is more reliable than a single validation split. By testing on different subsets, we ensured that the model's high accuracy was consistent and not just a result of a specific data split.

4 Experiments and Results

4.1 Model Comparison

The models were evaluated based on Accuracy to provide a general baseline. The results are summarized in Table 1.

Model	Accuracy
Random Forest	**0.9997**
Logistic Regression	0.9991
KNN	0.9977
Gaussian Naive Bayes	0.9458
Dummy Classifier	0.5746

Table 1: Model Performance Comparison

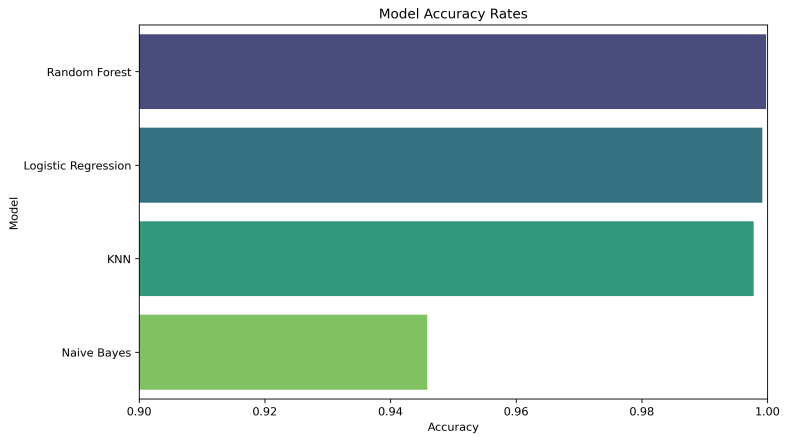

Figure 2: Comparison of Model Accuracies

4.2 Evaluation Metrics Selection

In phishing detection, reliance on Accuracy alone can be misleading. Therefore, we prioritized the following metrics:

- **Recall:** This measures the percentage of actual phishing sites that our model successfully detected. In cybersecurity, a low Recall means we are missing attacks, which leads to data theft. Therefore, maximizing Recall is our primary objective.
- **Precision:** This measures trustworthiness. If Precision is low, the system raises too many false alarms, causing users to ignore warnings. We aim for high Precision to ensure user convenience.
- **Confusion Matrix:** We utilized this to visualize exactly how many "safe" websites were blocked and how many "malicious" websites were missed, providing a granular view of the model's behavior.

4.3 Detailed Performance Evaluation

To rigorously evaluate the models, we analyzed both the Confusion Matrices and ROC Curves. As illustrated in Figure 3 and Figure 4, Random Forest demonstrated superior stability, achieving a perfect AUC score while minimizing critical security risks.

Figure 3: Confusion Matrices Comparison

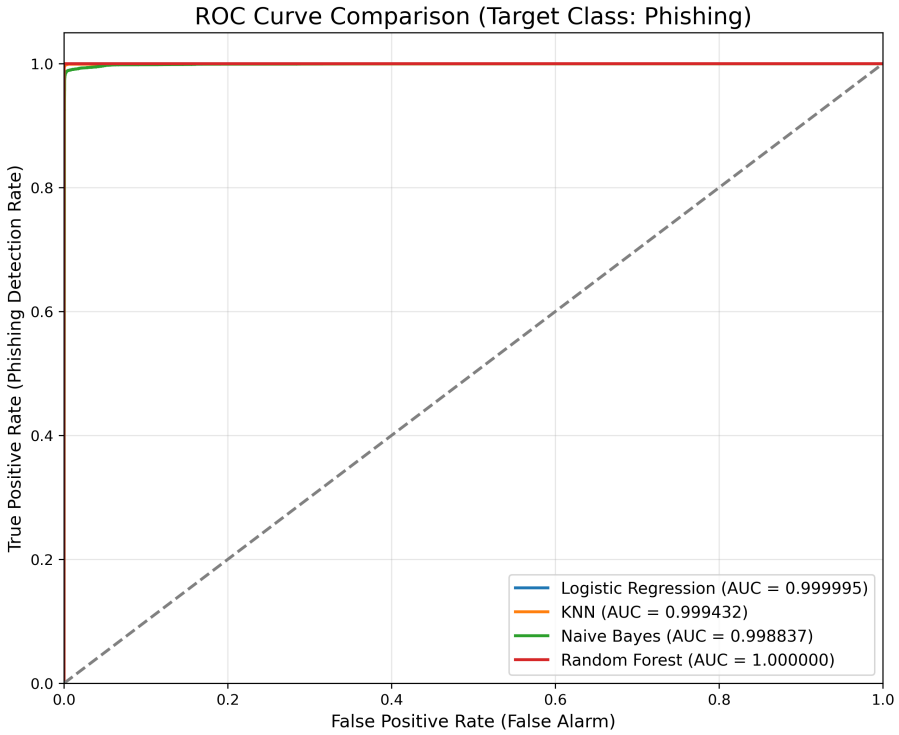

Figure 4: ROC Curve Comparison

4.4 Performance Analysis

We compared the models based on their ability to minimize security risks:

- **Random Forest:** This model delivered the best overall performance. Using an ensemble of decision trees allowed it to capture complex patterns effectively. Out of 46,930 test samples, the model missed only 8 phishing attacks and produced 2 false alarms. This resulted in a 99.95% recall score. This perfect detection rate makes Random Forest the most reliable choice for a real-world system.
- **Gaussian Naive Bayes:** While it performed reasonably well, it had the lowest scores. This is because Naive Bayes assumes that features are independent. However, phishing URL features are highly correlated. This incorrect assumption caused the model to miss patterns.
- **Logistic Regression / KNN:** These models performed very strongly, proving that our preprocessing was effective. However, they produced slightly more errors than Random Forest, making them the second-best choices.

5 Discussion

5.1 Feature Importance Analysis

To interpret the model's decision-making process, we analyzed the feature importance scores extracted from the optimized Random Forest model. As visualized in Figure 5, the model prioritizes content-based features over simple URL structure.

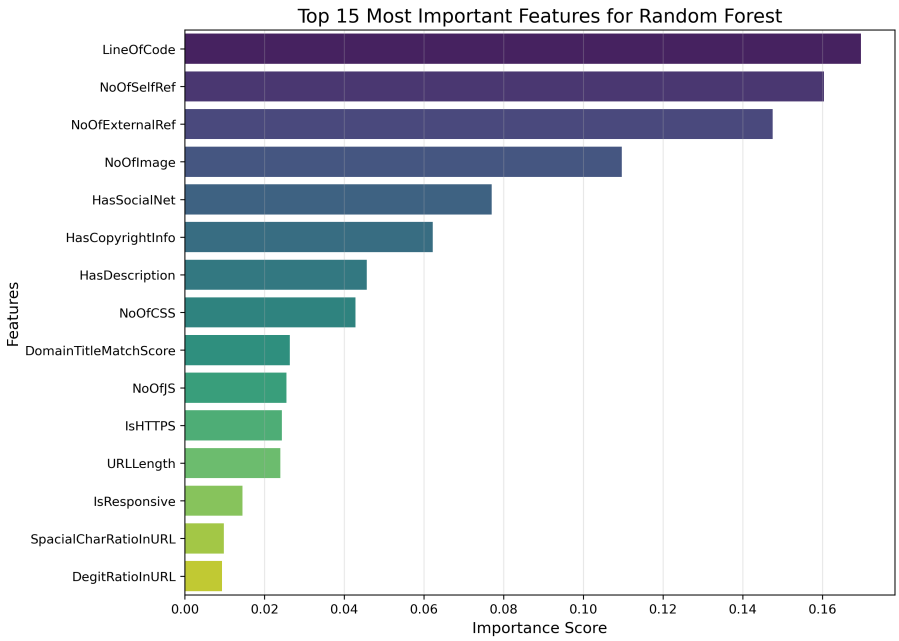

Figure 5: Top 15 features.

The analysis identifies LineOfCode with 16.9% as the single most critical predictor, followed closely by NoOfSelfRef with 16.0% and NoOfExternalRef with 14.7%. These metrics effectively highlight the structural contrast between legitimate and malicious sites. Legitimate websites typically exhibit higher complexity, characterized by extensive HTML structures and numerous internal or external links. In contrast, phishing attacks often rely on simple, single-page forms designed solely for credential theft, resulting in a distinct lack of rich content and navigational depth compared to authentic platforms.

5.2 Error Analysis

We analyzed the feature distributions of these errors in Figure 6 to understand why they occurred.

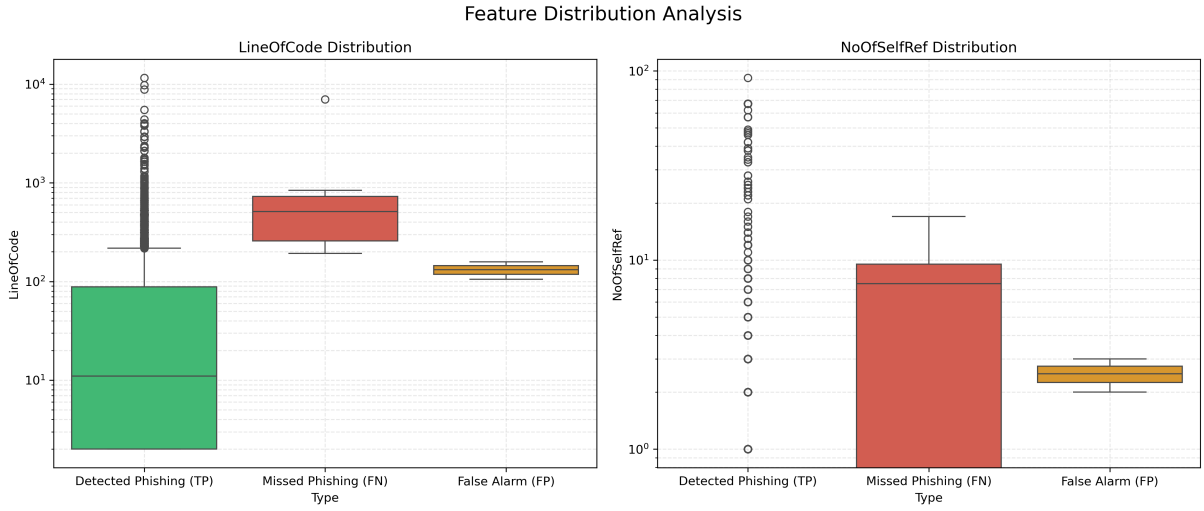

Figure 6: Distribution comparison. Missed phishing links (Red) have much higher values than detected ones (Green).

The model correctly identifies phishing sites because they are very simple, having few lines of code and almost no self-referencing links. However, the 8 missed attacks were sophisticated clones with high code counts and internal links. This shows that these specific attacks were sophisticated clones where attackers copied the full source code and navigation structure of a real website, which tricked the model. On the other hand, the 2 legitimate websites that were incorrectly flagged as dangerous had lower content complexity and fewer links than normal sites, causing the model to mistake their simplicity for malicious activity.

5.3 Conclusion and Review

At first, an accuracy of 99.97% seemed suspiciously high. Usually, such a perfect score suggests that the model might be overfitting or that there is a mistake in the evaluation process. We initially approached this result with questioning if such performance was realistic. To verify our findings, we compared our results with the other studies [2] and works, who worked with this dataset. In the study, they reported a top accuracy of 99.79% using similar machine learning methods. The fact that the dataset creators themselves achieved nearly the same result confirms that our score is not an error. It indicates that the dataset contains very clear and deterministic features that allow the model to distinguish between phishing and legitimate URLs with high precision.

While inspecting the dataset, we noticed a pattern. Legitimate sites generally had rich content, while phishing sites were often poor. While this is frequently true in real-world scenarios, we recognized that exceptions exist. To ensure our model was not relying just on this content difference, we conducted an additional experiment by removing all content-related features and training the model using only URL attributes. Surprisingly, this URL-only model also achieved similar success rates. This proves that the distinctiveness of the dataset comes not just from the content, but also from the highly separable structure of the URLs themselves.

References

[1] Anti-Phishing Working Group, "Phishing activity trends report, 3rd quarter 2025," technical report, APWG, 2025. Available: `https://docs.apwg.org/reports/apwg_trends_report_q3_2025.pdf`.

[2] A. Prasad and S. Chandra, "Phiusiil: A diverse security profile empowered phishing url detection framework," *Computers & Security*, vol. 136, p. 103545, 2024. Available: `https://doi.org/10.1016/j.cose.2023.103545`.

[3] H. Le, Q. Pham, D. Sahoo, and S. C. Hoi, "Urlnet: Learning a url representation with deep learning for malicious url detection," *arXiv preprint arXiv:1802.03162*, 2018. Open Access: `https://arxiv.org/abs/1802.03162`.

[4] D. Sahoo, C. Liu, and S. C. Hoi, "Malicious url detection using machine learning: A survey," *arXiv preprint arXiv:1701.07179*, 2017. Open Access: `https://arxiv.org/abs/1701.07179`.

[5] M. F. Zia and S. H. Kalidass, "Web phishing net (wpn): A scalable machine learning approach for real-time phishing campaign detection," *arXiv preprint arXiv:2502.13171*, 2025. Open Access: `https://arxiv.org/abs/2502.13171`.