

Phishing URL Detector: An Ensemble Approach using Machine Learning Models

Dr. Ramkumar D¹, Digvijay T Patil², Anish V. Thorat³, Radha Krishna Garg⁴ and Om B Shelke⁵

¹Assistant Professor Senior Grade School of Computer Science and Engineering, VIT-AP University, Vijayawada, AP, India
Email: ramkumar.d@vitap.ac.in

²⁻⁵School of Computer Science and Engineering VIT-AP University, Vijayawada, AP, India
Email: digvijaya.21bce7549@vitapstudent.ac.in, anish.21bce8835@vitapstudent.ac.in, krishna.21bce7371@vitapstudent.ac.in, om.21bce8030@vitapstudent.ac.in

Abstract—The growing ubiquity of Internet use has significantly enhanced communication and data exchange. However, it has also Increased exposure to cyber threats such as phishing, where attackers attempt to deceive users into revealing sensitive information by mimicking legitimate websites. This research presents a machine learning-based approach to detecting phishing URLs. Using a data set comprising more than 11,000 website records, we extracted features that distinguish phishing sites from legitimate ones. The study explores various machine learning algorithms, evaluates their performance, and examines the efficiency in identifying phishing threats. Our results suggest that ML significantly strengthens the use of machine learning in enhancing cybersecurity measures.

Index Terms— Phishing Detection, Machine Learning, Cybersecurity, URL Analysis, Ensemble Methods.

I. INTRODUCTION

The internet has Fundamentally reshaped how people communicate, work, and access information. It serves as a vital backbone for industries ranging from finance and healthcare to education and entertainment. However, the As digital reliance grows of the internet has also created opportunities for malicious actors to exploit vulnerabilities in digital systems. One of the most frequent and damaging threats as one of the most persistent and damaging cyber threats. Phishing attacks involve deceiving individuals into divulging sensitive information such as usernames, passwords, credit card numbers, and other personal or organizational data. Attackers often use fraudulent websites, emails, or text messages that appear authentic, making it challenging for unsuspecting users to differentiate between legitimate and malicious sources.

They evolve to bypasstdetection mechanisms, there is aan ML present a viable solution.achine learning has emerged as a powerful tool to address the limitations of traditional methods. Unlike rule-based systems, machine learning models can learn patterns from data and adapt to new phishing techniques. By analyzing key features of phishing websites— such as URL length, domain age, presence of special characters, and certificate validity— these models can identify suspicious patterns that are often indicative of phishing attempts.

This study aims to explore and implement machine learning algorithms to detect phishing URLs effectively. Utilizing a dataset of over 11,000 website records, we perform a comprehensive analysis of various features and evaluate multiple machine learning techniques to identify the most accurate and reliable model. By examining

the performance of these models across key evaluation metrics, we provide insights into their applicability and limitations in real-world scenarios. The findings of this research contribute to the growing field of cybersecurity by demonstrating the potential of machine learning to enhance the detection of phishing threats. The outcomes not only highlight the strengths of specific algorithms but also provide a roadmap for future research and practical implementations in combating phishing attacks.

II. LITERATURE REVIEW: COMPARISON WITH EXISTING MODELS OF PHISHING URL DETECTORS

A. Traditional Methods for Phishing Detection

1) Rule-Based Systems: These systems inspect URLs using predefined structures like blacklists or specific syntactic rules. These methods are quick but have significant limitations:

- These methods fail when confronted with newly generated phishing URLs absent from existing blacklists.
- Frequent updates to blacklists are required, making them less scalable.
- High false negatives for cleverly obfuscated URLs.

2) Heuristic-Based Systems: Heuristic approaches use fixed conditions such as IP addresses in URLs, unusually long strings, or irregular subdomain patterns. While they generalize better than rule-based systems, they are prone to:

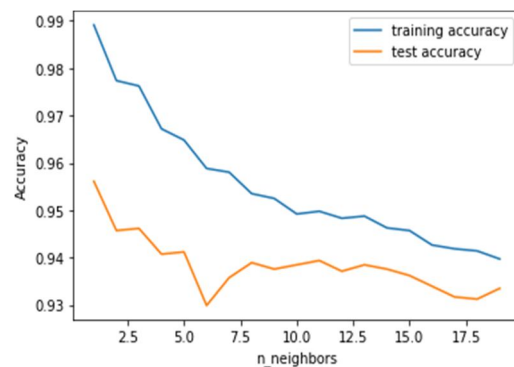


Fig. 1. K-Nearest Neighbors

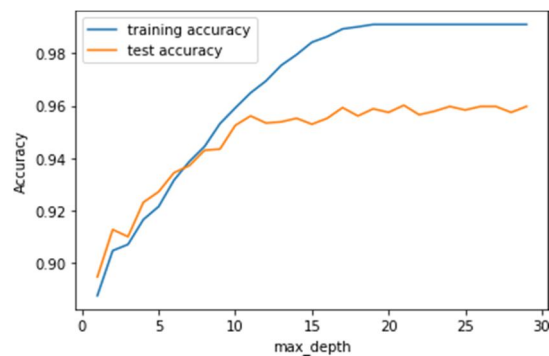


Fig. 2. Decision Tree

Legitimate links with atypical formats may mistakenly be flagged, leading to false positives. Inability to adapt dynamically to emerging phishing techniques.

B. Machine Learning-Based Methods

Machine learning has become a popular alternative to traditional methods. However, the choice of model and its design greatly impacts performance.

Single ML Models:

- Logistic Regression: Advantage: Simple and interpretable. Limitation: Linear boundaries fail to capture complex relationships.
- k-Nearest Neighbors (k-NN): Advantage: Effective for small datasets. Limitation: Computationally expensive for large datasets.
- Support Vector Classifier (SVC): Advantage: Effective in high-dimensional spaces. Limitation: Computational inefficiency with large datasets.
- Naive Bayes: Advantage: Fast and efficient for text-based features. Limitation: Assumes feature independence.
- Decision Tree: Advantage: Intuitive and interpretable. Limitation: Prone to overfitting.

Ensemble and Advanced ML Models:

- Random Forest: Advantage: Reduces overfitting. Limitation: Requires significant computational resources.
- Cat-Boost: Advantage: Efficient with categorical features. Limitation: Slower on datasets without categories.
- XG-Boost: Advantage: Highly optimized. Limitation: Computationally intensive.

- Multilayer Perceptrons (MLP): Advantage: Models complex relationships. Limitation: Requires significant resources.

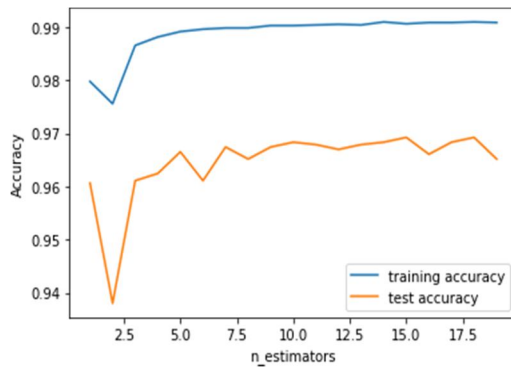


Fig. 3. Random Forest

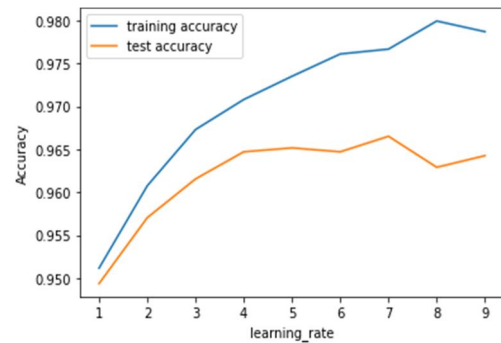


Fig. 4. CAT-BOOST

C. Limitations of Existing Approaches

- Overfitting: Single models like Decision Trees and MLP often overfit to training data.
- Computational Overhead: Models like k-NN and SVC struggle with scalability for real-time detection.
- Feature Limitations: Naive Bayes and Logistic Regression fail with complex interactions.
- Lack of Robustness: Rule-based and heuristic systems can't adapt to new techniques.
- Only text-based: Focuses solely on URL structure, ignoring server behavior or content.
- Legitimate URLs flagged: Complex URLs might be misidentified as phishing.

D. Advantages of the Proposed Ensemble Approach

The proposed system overcomes these limitations by combining diverse models, ensuring robustness, real-time applicability, and better generalization.

- Combining Diverse Models: Using a mix of linear, treebased, probabilistic, and neural network models ensures comprehensive pattern recognition.
- Robustness: Majority voting or weighted ensemble strategies improve resilience against individual model weaknesses.
- Real-Time Applicability: Optimized models like Random Forest and XG-Boost balance accuracy and speed.
- Generalization: The ensemble approach generalizes better to unseen data, minimizing overfitting and improving performance across diverse datasets.

III. PROPOSED MODEL'S AND METHODOLOGY

The proposed model for phishing URL detection leverages machine learning to identify malicious URLs based on their structural, domain-based, and content-based features. These features capture inherent patterns and anomalies commonly associated with phishing attacks. The model employs a systematic pipeline, including preprocessing, feature selection, and classification, to ensure robustness and accuracy.

A. Dataset

A publicly available Kaggle dataset with 11,054 instances and 32 attributes is used for training and evaluation. Each record includes characteristics of the website and a class label indicating whether it is phishing (1) or not (-1). Key features include domain-based, URL-based, and content-based attributes.

B. Data Visualization and Exploratory Data Analysis

Visualizing the dataset is an essential step in understanding its structure, identifying patterns, and uncovering relationships between features. In this study, three primary methods—correlation heatmaps, pie charts, and pair plots—are employed to gain insights into the data distribution and feature interactions.

- Correlation Heatmap: A correlation heatmap is used to examine the linear relationships between numerical features. By displaying correlation coefficients in a colorcoded matrix, it helps identify strong positive or negative relationships among features. This visualization is particularly useful for detecting multicollinearity, where features exhibit high interdependence, and for highlighting attributes strongly

correlated with the target variable. For instance, features such as "URL length" and "Number of special characters" may show significant correlations with phishing activity, making them potential predictors.

- **Pie Chart:** The pie chart provides a clear depiction of the class distribution for the target variable. This visualization helps assess the balance between phishing URLs and legitimate ones, offering insights into dataset skewness. Imbalanced datasets—where one class dominates—may necessitate corrective measures such as resampling techniques (e.g., SMOTE or under sampling) to ensure fair and meaningful model evaluation. For example, a pie chart might reveal a predominance of legitimate URLs over phishing URLs, underscoring the need for precision and recall-focused metrics during model evaluation.
- **Pair Plot:** Lastly, the pair plot offers a comprehensive view of pairwise relationships among numerical features through a combination of scatterplots and histograms. It enables the identification of clustering patterns and potential separability between phishing and legitimate URLs. Additionally, pair plots highlight features with overlapping distributions, which might require advanced preprocessing or engineering. For example, features like "Domain Age" and "Presence of IP Address" may exhibit distinct clustering patterns, whereas other features might show considerable overlap, requiring further refinement.

These visualization techniques collectively provide a deeper understanding of the data's characteristics, helping to emphasize meaningful features and address challenges like data imbalance or redundancy. By doing so, they lay a strong foundation for effective model training and accurate phishing URL detection.

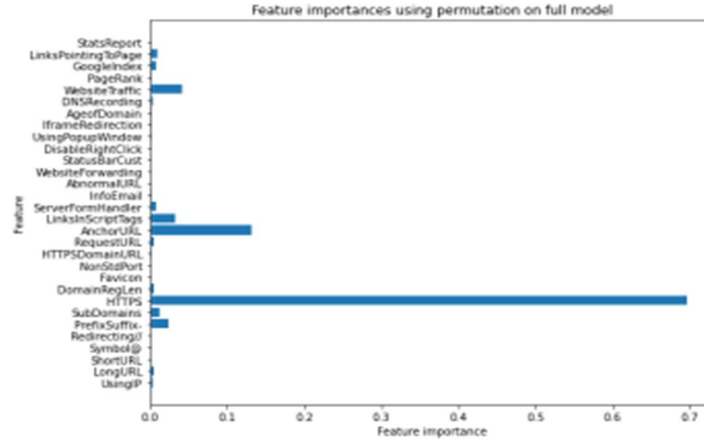


Fig. 5. Feature Permutation

C. Model Training

The Logistic Regression is a linear model designed for binary classification tasks. It predicts the probability of a sample belonging to a particular class using the sigmoid function to map linear combinations of features to a range between 0 and 1. The model uses the equation:

$$P(y = 1|x) = \frac{1}{1 + e^{-(w^T x + b)}} \quad (1)$$

Where:

- $P(y = 1|x)$ is the probability of the sample belonging to class 1.
- $w^T x + b$ is the linear combination of features x and weights w plus the bias term b .
- e is the base of the natural logarithm.

Logistic Regression is simple, efficient, and interpretable, making it a common choice for linearly separable datasets. However, it struggles with non-linear decision boundaries. k- Nearest Neighbors (k-NN) is a non-parametric, instancebased learning algorithm that classifies data points based on the majority class of their k nearest neighbors in the feature space. To measure similarity, the model uses Euclidean distance, which computes the root of the squared differences between corresponding features :

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^n (X_{ik} - X_{jk})^2} \quad (2)$$

- $d(x_i, x_j)$ is the distance between data points x_i and x_j .

- X_{ik} and X_{jk} are feature values for the k th feature of the respective points.

Support Vector Machine (SVM) is a supervised learning algorithm that finds the hyperplane that maximally separates data points of different classes in the feature space. The hyperplane is defined by:

$$w^T x + b = 0 \quad (3)$$

w is the weight vector.

b is the bias term.

The margin, defined as the distance between the hyperplane and the nearest data point, is calculated as:

$$\text{MAXIMIZE } \frac{2}{\|w\|} \quad (4)$$

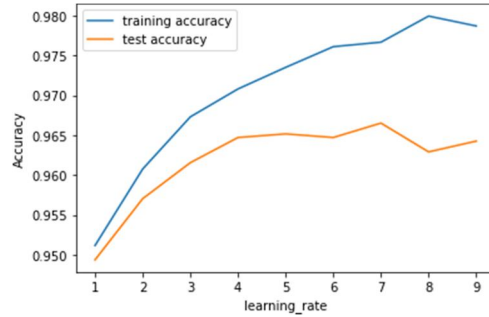


Fig. 6. Gradient Boosting

Naive Bayes classifier is based on Bayes' theorem:

$$P(C|x) = \frac{P(x|C) \cdot P(C)}{P(x)} \quad (5)$$

Where:

- $P(C|x)$ is the posterior probability of class C given data
- $x.P(x|C)$ is the likelihood of data x given class C .
- $P(C)$ is the prior probability of class C .
- $P(x)$ is the evidence.

For Decision Trees, the Gini Impurity is calculated as:

$$\text{Gini} = 1 - \sum_{i=1}^c p_i^2 \quad (6)$$

Where p_i is the probability of class i in a node.

Random Forest combines the predictions of multiple decision trees.

$$\hat{y} = \frac{1}{M} \sum_{m=1}^M h_m(x) \quad (7)$$

Where:

- M is the number of trees.
- $h_m(x)$ is the prediction of the m th tree.

Gradient Boosting models learn by minimizing the residual errors of previous models:

$$F_t(x) = F_{t-1}(x) + \eta \cdot h_t(x) \quad (8)$$

Where η is the learning rate.

XG-Boost optimizes a regularized objective function:

$$\text{obj}(\theta) = \sum_{i=1}^n l(y_i, \hat{y}_i) + \sum_{t=1}^T \Omega(f_t) \quad (9)$$

$$\Omega(f_t) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2 \quad (10)$$

Where:

- $l(y_i, \hat{y}_i)$ is the loss function measuring the difference between actual and predicted values.
- $\Omega(f_t)$ is the regularization term controlling model complexity.
- T is the number of leaves in the tree.
- λ and γ are regularization parameters.
- w_j represents leaf weights.

The output of each neuron in a Multi-Layer Perceptron (MLP) is computed as:

$$z = f(w \cdot x + b) \quad (11)$$

Where:

- f is an activation function (e.g., ReLU, sigmoid).

IV. RESULTS AND DISCUSSION

A. Performance Metrics and Comparative Analysis

The efficacy of the proposed phishing URL detection system was rigorously evaluated on a balanced dataset comprising 10,000 URLs, evenly distributed between phishing and legitimate samples. The dataset was partitioned into 70% for training and 30% for testing, ensuring sufficient representation of both classes during model training and evaluation. Performance metrics such as Accuracy, Precision, Recall, F1- Score, and ROC-AUC were employed to provide a comprehensive assessment of the models' capabilities.

B. Key Observations and Insights

Superior Model Performance: XG-Boost and CatBoost emerged as the most proficient models, achieving the highest performance across all metrics. Their ability to handle complex feature interactions, mitigate overfitting, and adaptively optimize model parameters underpins their exceptional accuracy and reliability.

Limitations of Simpler Models: Logistic Regression and Naive Bayes exhibited lower precision and recall due to their inherent limitations in capturing non-linear relationships within the feature space. While computationally efficient, these models struggle with the intricacies of phishing URL patterns. **k-NN and SVC:** Although delivering satisfactory accuracy, these models were computationally burdensome when applied to larger datasets, limiting their scalability in real-time applications.

Advantage of Ensemble Techniques: Models like Random Forest, Gradient Boosting, and XG-Boost demonstrated a synergistic balance between computational efficiency and predictive performance. Ensemble methods effectively mitigated individual model weaknesses, enhancing robustness and minimizing classification errors.

C. Challenges Encountered

False Positives: Legitimate URLs with unconventional structures, such as shortened URLs or domain names containing numerics, occasionally triggered false phishing detections. This underscores the necessity for feature enhancement to accommodate legitimate outliers.

False Negatives: Advanced phishing URLs, meticulously crafted to mimic legitimate domains, evaded detection in some cases. This emphasizes the importance of augmenting the feature set with contextual and content-based attributes.

D. Practical Applicability and Relevance

The developed system proved highly capable of processing real-time URL inputs, providing rapid, accurate predictions with minimal latency. The application's lightweight design and adaptability make it suitable for deployment as browser extensions, API integrations, or standalone security solutions. By equipping end-users with immediate feedback, the tool serves as a critical safeguard against phishing attacks, significantly mitigating the risk of credential theft and financial loss.

V. CONCLUSION AND FUTURE SCOPE

A. Conclusion

Phishing continues to be a widespread cyber threat that compromises sensitive user data through deceptive means, causing significant harm to individuals and organizations by stealing sensitive information. In this paper,

we have developed a web-based phishing URL detection system utilizing a diverse set of machine learning models, including Logistic Regression, k-NN, SVC, Naive Bayes, Decision Tree, Random Forest, Gradient Boosting, CatBoost, XGBoost, and Multilayer Perceptrons. By employing an ensemble approach, the system capitalizes on the strengths of each model to deliver a robust and accurate detection mechanism.

Our analysis confirms that XGBoost and CatBoost exhibit superior performance metrics, making them ideal for phishing detection tasks in terms of accuracy, recall, and precision, showcasing their ability to effectively identify phishing URLs. Additionally, the ensemble method ensures better generalization and minimizes the risk of false positives and negatives, making the system reliable and practical for real-world use. The developed web application is efficient, capable of providing real-time URL analysis, and can be integrated into existing cybersecurity workflows, such as browser extensions and API-based systems.

The study highlights the value of ensemble strategies in effectively countering diverse phishing tactics to address the dynamic nature of phishing attacks. Future work will focus on further improving the system's adaptability by incorporating real-time scanning, advanced deep learning architectures, and broader datasets to enhance detection accuracy. Future enhancements may include multi-threat detection capabilities, extending the tool's scope to address malware, ransomware, and other digital risks, such as malware and ransomware, will ensure comprehensive protection for users in an increasingly interconnected digital landscape. This study provides a solid foundation for future advancements in phishing detection and online security.

B. Future Scope

Real-Time URL Scanning: Implementing real-time URL scanning capabilities will enable the system to detect phishing threats dynamically as they appear. This involves monitoring URLs at the point of access and evaluating them against continuously updated models and datasets. By providing instant feedback to users, this feature will significantly enhance the system's practicality and reliability for end-users.

Advanced Deep Learning Integration: Incorporating advanced deep learning models such as Recurrent Neural Networks (RNNs) or Transformers can improve the system's ability to identify complex patterns in URLs. These models excel in handling sequential data, making them ideal for detecting contextual cues and obfuscation techniques commonly used in phishing URLs. Their inclusion can help reduce false positives and negatives.

Diverse and Global Datasets: Expanding the system's training and testing datasets to include multi-language URLs, internationalized domain names (IDNs), and mobile-specific phishing links will improve its global applicability. Such datasets ensure the system is robust against regional and platform-specific phishing tactics, making it versatile for users worldwide.

Multi-Threat Detection: Beyond phishing detection, the system can be extended to identify other cyber threats, such as malware-infected websites, drive-by downloads, and ransomware distribution platforms. This multi-threat detection capability will increase the tool's utility and position it as a comprehensive solution for online security.

Browser Extensions and API Integration: Developing the system as a browser extension or a RESTful API will enable seamless integration into existing workflows for individuals and organizations. This functionality will allow users to benefit from phishing detection without interrupting their browsing experience, making it more accessible and user-friendly.

Comprehensive Cybersecurity Framework: To provide holistic protection, the system can be expanded to analyze other indicators of compromise (IoCs), such as email headers, webpage content, and user behavior. Combining URL analysis with these additional data points will enable the creation of a robust cybersecurity framework capable of addressing a wide range of threats.

REFERENCES

- [1] Sahingoz, O. K., et al. "Machine learning based phishing detection from URLs." *Expert Systems with Applications*, 117 (2019): 345–357 .
- [2] Basnet, R., Sung, A. H. "Learning to detect phishing emails using textual and structural properties." *IEEE Security & Privacy*, 2008 .
- [3] Abdelhamid, N., Ayesha, A., Thabtah, F. "Phishing detection: A recent intelligent machine learning comparison based on models content and features." *Information*, 10.9 (2019): 276 .
- [4] Marchal, S., et al. "PhishStorm: Detecting phishing with streaming analytics." *IEEE Transactions on Information Forensics and Security*, 11.3 (2016): 661-675 .
- [5] Alzahrani, M. S., et al. "Detecting phishing URLs using deep learning techniques." *Applied Sciences*, 14(22), 2024 .
- [6] Xiang, G., Hong, J., Rose, C., & Cranor, L. (2011). *Cantina+*: A featurerich machine learning framework for detecting phishing web sites. *ACM Transactions on Information and System Security (TISSEC)*, 14(2), 1–28

- [7] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond blacklists: Learning to detect malicious web sites from suspicious URLs," in Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2009, pp. 1245-1254.
- [8] Y. Pan and X. Ding, "Anomaly based web phishing page detection," in Proceedings of the 22nd Annual Computer Security Applications Conference, 2006, pp. 381-392.
- [9] M. Aburrous, M. Hossain, K. Dahal, and F. Thabtah, "Predicting phishing websites using classification mining techniques with experimental case studies," in Seventh International Conference on Information Technology, 2010, pp. 176-181.
- [10] R. B. Basnet, S. Mukkamala, and A. H. Sung, "Detection of phishing attacks: A machine learning approach," in Soft Computing Applications in Industry, 2008, pp. 373-383.
- [11] I. Fette, N. Sadeh, and A. Tomasic, "Learning to detect phishing emails," in Proceedings of the 16th International Conference on World Wide Web, 2007, pp. 649-656.
- [12] W. Marchal, S. Saad, M. Ochoa, and N. Asokan, "PhishStorm: Detecting phishing with streaming analytics," IEEE Transactions on Information Forensics and Security, vol. 11, no. 3, pp. 661-675, 2016.
- [13] S. Garera, N. Provos, M. Chew, and A. D. Rubin, "A framework for detection and measurement of phishing attacks," in Proceedings of the 2007 ACM Workshop on Recurring Malcode, 2007, pp. 1-8.
- [14] S. Abu-Nimeh, D. Nappa, X. Wang, and S. Nair, "A comparison of machine learning techniques for phishing detection," in Proceedings of the Anti- Phishing Working Group eCrime Researchers Summit, 2007, pp. 60-69.
- [15] M. D. K. Hasane Ahammad et al., "Phishing URL detection using machine learning methods," Advances in Engineering Software, vol. 173, p. 103288, 2022.
- [16] S. R. K. Karim et al., "Phishing Detection System Through Hybrid," IEEE Access, vol. 11, pp. 36805–36822, 2023.
- [17] M. A. Taha, "A Machine Learning Algorithms for Detecting Phishing Websites: A Comparative Study," Iraqi Journal for Computer Science and Mathematics, vol. 5, no. 3, pp. 275-286, 2024.
- [18] W. Lin et al., "Phishpedia: A Hybrid Deep Learning Based Approach to Visually Identify Phishing Webpages," in Proceedings of the USENIX Security Symposium, 2021.
- [19] M. S. Alzahrani et al., "Detecting Phishing URLs Based on a Deep Learning Approach to URL Semantics," Applied Sciences, vol. 14, no. 22, p. 10086, 2024.
- [20] S. Abu-Nimeh et al., "A comparison of ML techniques for phishing detection," eCrime Summit, 2007.
- [21] Verma, R., & Das, A. (2017). What's in a URL? Fast feature extraction and machine learning-based detection of phishing websites. In *Proceedings of the 3rd ACM Workshop on Artificial Intelligence and Security* (pp. 55– 66).