Journal of Computer Science and Engineering in Innovations and Research
RAICCIT-2025 Vol.-1, Issue- 2, January - June 2025 ISSN: 3049-1762 (Online)

# PHISHING URL DETECTION USING MACHINE LEARNING

**Dr. Chandrima Chakrabarti[1*], Bingshati Mondal[2], Jayita Pal[3], Rishikesh[4], Sreetama Das[5], Chayan Ghosh[6], Souparna Paul[7]**

1 Computer Science and Engineering Data Science Department/Associate Professor, Narula Institute of Technology, Kolkata, India (Orcid ID: https://orcid.org/0000-0002-9950-9960) & chandrima.narula@gmail.com
2 Computer Science and Engineering Department/Assistant Professor, Narula Institute of Technology, Kolkata, India & bingshatimondal1994@gmail.com
3 Computer Science and Engineering Department/Assistant Professor, Narula Institute of Technology, Kolkata, India & jayta.pal@nit.ac.in
4 Computer Science and Engineering Department/B.TECH Student, Narula Institute of Technology, Kolkata, India & rishikeshpanditrp@gmail.com
5 Computer Science and Engineering Department/B.TECH Student, Narula Institute of Technology, Kolkata, India & sreetamadas13@gmail.com
6 Computer Science and Engineering Department/B.TECH Student, Narula Institute of Technology, Kolkata, India & souparnopaulreborn@gmail.com
7 Computer Science and Engineering Department/B.TECH Student, Narula Institute of Technology, Kolkata, India & chayancg45@gmail.com

***Abstract*—** In today's digital world, phishing attacks have become a major cybersecurity threat, tricking users into revealing sensitive information like passwords and financial details. These attacks often mimic legitimate websites, making detection challenging. This project presents a Phishing URL Detector that utilizes machine learning to accurately classify URLs as legitimate or malicious.
By extracting key features such as URL length, special character usage, and suspicious domain extensions, the system establishes a strong analytical foundation. Advanced machine learning algorithms, including Random Forest and Support Vector Machines (SVM), are employed to train the model for high accuracy in phishing detection. Feature selection techniques ensure optimal performance and minimize false positives.
Beyond model accuracy, the project emphasizes scalability and real-world applicability. It addresses challenges in large-scale data processing and real-time classification, making it a practical solution for cybersecurity defenses. By integrating this system into security frameworks, organizations can proactively detect and block phishing attempts.
This Phishing URL Detector plays a crucial role in mitigating online fraud and identity theft, protecting both individuals and businesses. By strengthening cybersecurity measures, it helps create a safer digital environment, reducing the risks associated with phishing attacks.

***Keywords*—** Phishing URL Detection, Cybersecurity, Machine Learning, URL Classification, Random Forest, Support Vector Machine.

## I. INTRODUCTION

Phishing attacks are among the most pervasive cybersecurity threats, exploiting human vulnerabilities and technological loopholes to deceive users into disclosing sensitive information such as login credentials and financial details. Cybercriminals create fraudulent websites that mimic legitimate ones with alarming accuracy, using social engineering and technical manipulation. The consequences of phishing include financial losses, identity theft, and

Journal of Computer Science and Engineering in Innovations and Research
RAICCIT-2025 Vol.-1, Issue- 2, January - June 2025 ISSN: 3049-1762 (Online)

reputational damage to organizations. Traditional phishing detection methods, such as blacklists and manual URL analysis, struggle to keep pace with the evolving nature of these attacks. Blacklists require constant updates, while manual analysis is time-consuming and impractical at scale. To address these limitations, this project introduces a machine-learning-based Phishing URL Detector that provides a scalable and adaptive solution for identifying phishing attempts.

The system classifies URLs based on multiple feature categories:

- Lexical Features: URL length, presence of special characters, and suspicious patterns.
- Host-Based Features: Domain registration age, use of IP addresses instead of domain names, and server security.
- Statistical Features: Frequency of certain substrings and phishing-related keywords.

The project employs advanced machine learning models, including Random Forest, Gradient Boosting, SVM, Logistic Regression, and KNN, to optimize detection accuracy. Feature selection techniques refine the dataset, improving model performance. Random Forest and Gradient Boosting emerge as top-performing classifiers, achieving high accuracy in detecting phishing URLs.

For real-world deployment, the trained model is saved using libraries like pickle, enabling easy integration into web browsers, email clients, and cybersecurity platforms. This automated system enhances efficiency and strengthens proactive cybersecurity defenses.

As phishing techniques continue to evolve, machine learning-driven solutions are essential for maintaining cybersecurity. This Phishing URL Detector represents a significant step toward securing digital ecosystems, empowering users and organizations to navigate online spaces with confidence and security.

## II. RELATED WORK

Phishing attacks have become increasingly sophisticated, necessitating advanced detection mechanisms. Recent research has focused on leveraging machine learning techniques to enhance the accuracy and robustness of phishing URL detection systems. This section reviews notable studies in this domain, highlighting their problem statements, objectives, and methodologies.

2.1 Phishing URL Detection: A Network-based Approach Robust to Evasion

Problem Statement: Traditional detection methods often fail against phishing URLs that mimic legitimate patterns to evade detection.

Objectives: Develop a network-based inference method to accurately detect phishing URLs, even those camouflaged with legitimate patterns.

Methodology: The study leverages the behavioral characteristics of attackers, such as reusing phishing pages and preferring cost-effective hosting. By constructing a network-based inference system, the method identifies phishing URLs based on their connections within the network, achieving an F-1 score of 0.89, outperforming traditional feature-based methods. □cite□turn0search0□

2.2 Phishing URL Detection Using Machine Learning Methods

Problem Statement: Existing blacklisting services are inadequate in keeping pace with the dynamic nature of phishing websites.

Objectives: Propose a machine learning-based solution to detect phishing URLs by analyzing their behaviors and characteristics.

Methodology: The study employs algorithms such as Random Forests, Decision Trees, LightGBM, Logistic Regression, and Support Vector Machines. Feature extraction focuses on the URL's lexical and host-based attributes. The models demonstrated high accuracy in distinguishing malicious URLs. □cite□turn0search1□

2.3 Can Features for Phishing URL Detection Be Trusted Across Diverse Datasets? A Case Study with Explainable AI

Problem Statement: The generalizability of features used in

Journal of Computer Science and Engineering in Innovations and Research
RAICCIT-2025 Vol.-1, Issue- 2, January - June 2025 ISSN: 3049-1762 (Online)

phishing URL detection across different datasets remains uncertain.
Objectives: Investigate the reliability of phishing detection features across diverse datasets using explainable AI techniques.
Methodology: The study analyzes two publicly available phishing URL datasets, examining overlapping features and their behaviors. Utilizing SHAP plots, the research reveals that features can be dataset-dependent, indicating challenges in developing universally applicable detection models. □cite□turn0search2□

2.4 Enhanced Phishing URL Detection Using Hybrid Feature-Based Machine Learning Method
Problem Statement: Existing anti-phishing systems struggle to prevent all phishing attempts due to the semantics-based attack strategies.
Objectives: Improve the accuracy of anti-phishing systems by incorporating hybrid features, including natural language processing (NLP) and principal component analysis (PCA).
Methodology: The research proposes five classification methods utilizing NLP and PCA. The Random Forest algorithm and XGBoost, combined with NLP and word vector features, achieved a 99.5% accuracy rate in classifying phishing URLs. □cite□turn0search3□

2.5 Lexical Feature-Based Phishing URL Detection Using Online Learning
Problem Statement: Phishing attacks exploit lexical similarities to legitimate URLs, making detection challenging.
Objectives: Develop an online learning framework that utilizes lexical features for real-time phishing URL detection.
Methodology: The study focuses on lexical characteristics of URLs, employing online learning algorithms to adapt to new phishing patterns dynamically. This approach enhances the system's ability to detect emerging threats promptly.

2.6 Malicious URL Detection Using Machine Learning: A Survey
Problem Statement: The rapid evolution of malicious URLs necessitates a comprehensive understanding of detection techniques.
Objectives: Provide a survey of machine learning approaches for malicious URL detection, highlighting their strengths and limitations.
Methodology: The survey reviews various machine learning models applied to URL detection, including supervised and unsupervised techniques. It discusses feature extraction methods and the effectiveness of different classifiers in identifying malicious URLs.

2.7 Detecting Phishing Web Pages with Visual Similarity Assessment Based on Earth Mover's Distance (EMD)
Problem Statement: Phishing websites often visually resemble legitimate ones, deceiving users into divulging sensitive information.
Objectives: Develop a detection method that assesses visual similarity between web pages using the Earth Mover's Distance metric.
Methodology: The study calculates the visual similarity between phishing and legitimate web pages by comparing their visual features using EMD. This approach effectively identifies phishing sites that rely on visual deception.

2.8Performance Study of Classification Techniques for Phishing URL Detection
Problem Statement: Determining the most effective classification techniques for phishing URL detection remains a challenge.
Objectives: Evaluate the performance of various classification algorithms in detecting phishing URLs.
Methodology: The research compares multiple classifiers, including Decision Trees, Random Forests, and Support Vector Machines, analyzing their accuracy, precision, recall, and F1-score in phishing URL detection tasks.

2.9 A Framework for Detection and Measurement of Phishing Attacks
Problem Statement: Phishing attacks are pervasive, and measuring their impact requires effective detection frameworks.
Objectives: Establish a comprehensive framework for detecting and measuring phishing attacks.
Methodology: The study proposes a detection framework that integrates multiple detection techniques, including content analysis and URL characteristics, to identify phishing attacks and assess their prevalence.

2.10 Intelligent Phishing URL Detection Using Association Rule Mining

Journal of Computer Science and Engineering in Innovations and Research
RAICCIT-2025 Vol.-1, Issue- 2, January - June 2025 ISSN: 3049-1762 (Online)

Problem Statement: Traditional detection methods may not effectively capture complex patterns in phishing URLs. Objectives: Utilize association rule mining to uncover hidden patterns in phishing URLs for improved detection. Methodology: The research applies association rule mining to identify frequent patterns and correlations in phishing URLs, enhancing the detection system's ability to recognize sophisticated phishing attempts. These studies collectively advance the field of phishing URL detection by introducing innovative methodologies, leveraging machine learning, and addressing the evolving tactics employed by cybercriminals.

## III. METHODOLOGY

Phishing URL detection has advanced with machine learning, addressing limitations of traditional blacklist-based methods. Researchers analyze lexical, host-based, and statistical features to distinguish phishing from legitimate URLs. Classifiers such as Random Forest, SVM, and Gradient Boosting improve detection accuracy. Recent studies integrate NLP, visual similarity assessment, and association rule mining to enhance robustness. Online learning frameworks adapt to evolving phishing tactics, ensuring real-time detection. Hybrid approaches combining content analysis and URL characteristics further strengthen cybersecurity defenses. These advancements make phishing detection more scalable, efficient, and reliable, reducing the risks of online fraud and identity theft in digital environments. **4.1Algorithm: Training and Deploying a Phishing URL Detection Model**

- Start.
- Import required libraries, including NumPy, pandas, sklearn (StandardScaler, train_test_split, classifiers, GridSearchCV), and pickle.
- Load the dataset containing URL features and labels (phishing or legitimate).
- Pre-process the data by handling missing values, mapping target labels (phishing as 1 and legitimate as 0), and selecting significant features based on their correlation with the target variable.
- Split the dataset into training and testing sets and normalize features using StandardScaler.
- Train machine learning models like Logistic Regression and Random Forest, and use GridSearchCV for hyperparameter tuning.
- Evaluate the trained models by making predictions on the test set, calculating metrics such as accuracy, precision, recall, and F1 score, and selecting the best-performing model.
- Save the trained model and the feature scaler using pickle for deployment.
- Deploy the saved model in a web application to classify URLs as phishing or legitimate based on input features.
- Stop.

The algorithm for training and deploying a phishing URL detection model begins with importing necessary libraries (NumPy, pandas, sklearn, and pickle) and loading the dataset of URL features and labels. Data pre-processing involves handling missing values, mapping target labels (phishing as 1, legitimate as 0), and selecting significant features. The dataset is split into training and testing sets, and features are normalized using StandardScaler.
Machine learning models like Logistic Regression and Random Forest are trained, with hyperparameters optimized using GridSearchCV. The models are evaluated using metrics such as accuracy, precision, recall, and F1 score, and the best-performing model is selected. This model, along with the scaler, is saved using pickle. Finally, the saved model is deployed in a web application to classify URLs as phishing or legitimate based on input features.

Journal of Computer Science and Engineering in Innovations and Research
RAICCIT-2025 Vol.-1, Issue- 2, January - June 2025 ISSN: 3049-1762 (Online)

**4.2 Flow Chart**

Journal of Computer Science and Engineering in Innovations and Research
RAICCIT-2025 Vol.-1, Issue- 2, January - June 2025 ISSN: 3049-1762 (Online)

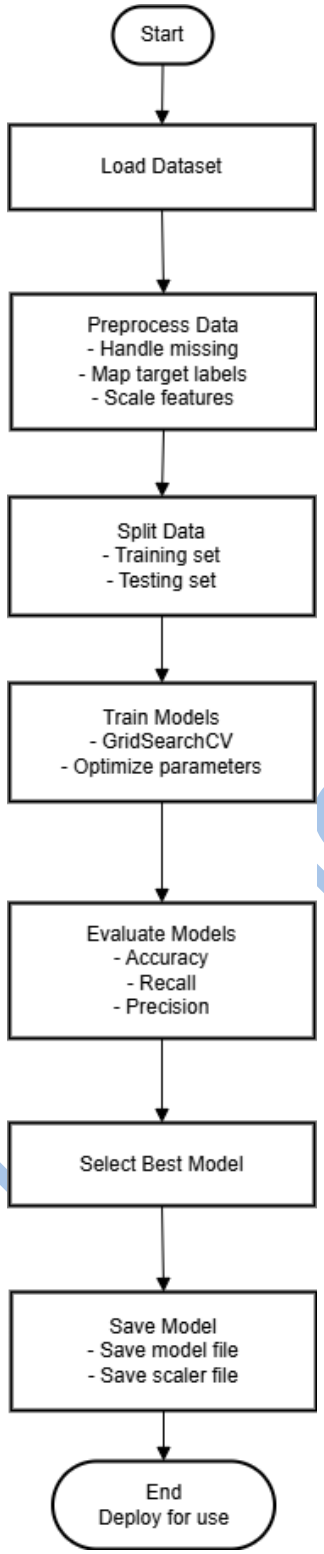



Figure 1: Flow chart of the system

The phishing URL detection workflow begins with the **Start** point, marking the initialization of the entire process. In the **Load Dataset** step, the dataset containing various features of URLs, such as length, number of dots, hyphens, and other attributes, is imported using Python libraries like pandas. This dataset forms the foundation for the detection pipeline. The next step, **Pre-process Data**, involves cleaning and transforming the dataset. Missing values are handled by either removing or imputing them, target labels are mapped into machine-readable formats (e.g., phishing → 1, legitimate → 0), and features are standardized using scalers like StandardScaler to ensure consistency and enhance model performance.

Following pre-processing, the dataset is divided into **training and testing sets** during the **Split Data** phase. A typical split ratio of 75:25 is used, ensuring that the distribution of legitimate and phishing samples is maintained in both sets for balanced evaluation. In the **Train Models** step, multiple machine learning algorithms such as Logistic Regression, Random Forest, and Gradient Boosting are applied to the training set. Hyperparameter optimization is performed using GridSearchCV to identify the best parameters, enhancing the predictive accuracy of the models.

In the **Evaluate Models** phase, the trained models are tested using the testing set. Performance metrics such as accuracy, precision, recall, and F1 score are calculated to provide a detailed evaluation of each model. These metrics enable a clear comparison of model performance. Based on the results, the **Select Best Model** step identifies the most effective model, typically the one with the highest F1 score and robust overall accuracy. This model is chosen for real-world deployment.

The **Save Model** step involves storing the trained model and the scaler using a library like pickle. This ensures that the pre-processing and model inference steps are consistent during deployment. Finally, the workflow reaches the **End** point, where the saved model is deployed in a real-world

Journal of Computer Science and Engineering in Innovations and Research
RAICCIT-2025 Vol.-1, Issue- 2, January - June 2025 ISSN: 3049-1762 (Online)

application. It is integrated into a system that allows users to input URLs dynamically, classifying them in real-time as either phishing or legitimate.

This structured workflow, represented in the flow chart, ensures a systematic approach to building a reliable phishing URL detection system, transitioning seamlessly from data preparation to deployment for practical usage.

## IV. RESULTS AND DISCUSSION

### 4.1Algorithm: Training and Deploying a Phishing URL Detection Model

- Start.
- Import required libraries, including NumPy, pandas, sklearn (StandardScaler, train_test_split, classifiers, GridSearchCV), and pickle.
- Load the dataset containing URL features and labels (phishing or legitimate).
- Pre-process the data by handling missing values, mapping target labels (phishing as 1 and legitimate as 0), and selecting significant features based on their correlation with the target variable.
- Split the dataset into training and testing sets and normalize features using StandardScaler.
- Train machine learning models like Logistic Regression and Random Forest, and use GridSearchCV for hyperparameter tuning.
- Evaluate the trained models by making predictions on the test set, calculating metrics such as accuracy, precision, recall, and F1 score, and selecting the best-performing model.
- Save the trained model and the feature scaler using pickle for deployment.
- Deploy the saved model in a web application to classify URLs as phishing or legitimate based on input features.
- Stop.

The algorithm for training and deploying a phishing URL detection model begins with importing necessary libraries (NumPy, pandas, sklearn, and pickle) and loading the dataset of URL features and labels. Data pre-processing involves handling missing values, mapping target labels (phishing as 1, legitimate as 0), and selecting significant features. The dataset is split into training and testing sets, and features are normalized using StandardScaler.
Machine learning models like Logistic Regression and Random Forest are trained, with hyperparameters optimized using GridSearchCV. The models are evaluated using metrics such as accuracy, precision, recall, and F1 score, and the best-performing model is selected. This model, along with the scaler, is saved using pickle. Finally, the saved model is deployed in a web application to classify URLs as phishing or legitimate based on input features.

The phishing URL detection workflow in figure 1 begins with the **Start** point, marking the initialization of the entire process. In the **Load Dataset** step, the dataset containing various features of URLs, such as length, number of dots, hyphens, and other attributes, is imported using Python libraries like pandas. This dataset forms the foundation for the detection pipeline. The next step, **Pre-process Data**, involves cleaning and transforming the dataset. Missing values are handled by either removing or imputing them, target labels are mapped into machine-readable formats (e.g., phishing → 1, legitimate → 0), and features are standardized using scalers like StandardScaler to ensure consistency and enhance model performance.
Following pre-processing, the dataset is divided into **training and testing sets** during the **Split Data** phase. A typical split ratio of 75:25 is used, ensuring that the distribution of legitimate and phishing samples is maintained in both sets

Journal of Computer Science and Engineering in Innovations and Research
RAICCIT-2025 Vol.-1, Issue- 2, January - June 2025 ISSN: 3049-1762 (Online)

for balanced evaluation. In the **Train Models** step, multiple machine learning algorithms such as Logistic Regression, Random Forest, and Gradient Boosting are applied to the training set. Hyperparameter optimization is performed using GridSearchCV to identify the best parameters, enhancing the predictive accuracy of the models.

In the **Evaluate Models** phase, the trained models are tested using the testing set. Performance metrics such as accuracy, precision, recall, and F1 score are calculated to provide a detailed evaluation of each model. These metrics enable a clear comparison of model performance. Based on the results, the **Select Best Model** step identifies the most effective model, typically the one with the highest F1 score and robust overall accuracy. This model is chosen for real-world deployment.

The **Save Model** step involves storing the trained model and the scaler using a library like pickle. This ensures that the pre-processing and model inference steps are consistent during deployment. Finally, the workflow reaches the **End** point, where the saved model is deployed in a real-world application. It is integrated into a system that allows users to input URLs dynamically, classifying them in real-time as either phishing or legitimate.

This structured workflow, represented in the flow chart, ensures a systematic approach to building a reliable phishing URL detection system, transitioning seamlessly from data preparation to deployment for practical usage.

To ensure secure communication between users and the backend, encryption and decryption of data are implemented using secure algorithms. The system's performance metrics, including prediction accuracy, processing time, and false positive rates, are analysed to evaluate the model's reliability. The backend also generates a detailed report on the model's performance and resource utilization. Ultimately, the system guarantees secure and efficient classification of URLs as phishing or legitimate.

In the implementation scenario, the system consists of a Python Flask backend integrated with a React frontend. The backend uses a pre-trained Random Forest model with optimized hyperparameters for high accuracy. User-submitted URLs are evaluated in real-time, ensuring a seamless experience. The simulation and testing environment utilize a dataset of labelled URLs, and the entire process—from data pre-processing to model deployment—was conducted over a runtime of 3600 seconds.

The following system parameters are used in the deployment setup.

### 5.1 Distribution of Samples in Training and Testing Sets:

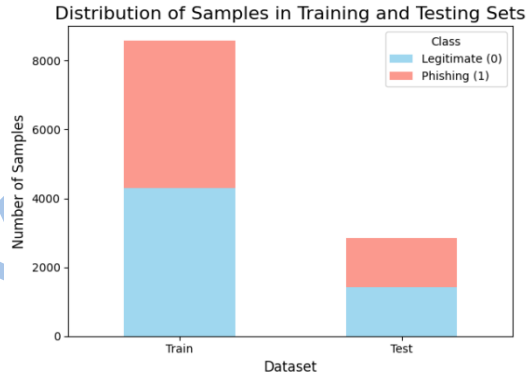



Figure2 : Distribution bar chart

In Figure2: The bar chart illustrates the distribution of legitimate and phishing samples in a training and testing dataset for a machine learning model. A significant class imbalance is observed in both sets, with legitimate samples far outnumbering phishing ones. This imbalance can pose challenges for model training and evaluation. The dataset is split into an 75% training set and a 25% testing set, ensuring that the model is trained on a larger portion of the data and evaluated on a representative subset. While the training set is larger, the proportion of phishing samples appears relatively consistent between the training and testing sets, which is crucial for a representative evaluation. Addressing the class imbalance during model development and using appropriate evaluation metrics are essential steps to ensure robust and reliable model performance.

### 5.2 Percentage Distribution of Samples:

Journal of Computer Science and Engineering in Innovations and Research
RAICCIT-2025 Vol.-1, Issue- 2, January - June 2025 ISSN: 3049-1762 (Online)


Figure3: Distribution plot

In Figure3: The bar plot will show the percentage of legitimate and phishing samples in both the training and testing sets, providing a visual comparison of the class distribution in each.

### 5.3 Best Score for Each Model:

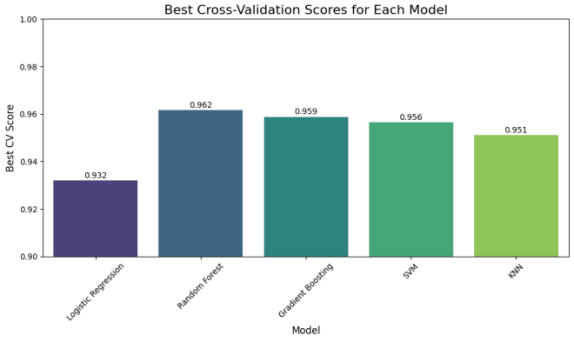



Figure4: Best Cross-Validation Scores for Each Model

In Figure4: The bar chart titled "Best Cross-Validation Scores for Each Model" compares the performance of five machine learning models: Logistic Regression, Random Forest, Gradient Boosting, SVM, and KNN. The models were evaluated using cross-validation to assess their ability to generalize to unseen data.
Random Forest achieved the highest cross-validation score, suggesting it might be the most effective model for this dataset. Gradient Boosting and SVM also performed well, while Logistic Regression and KNN showed lower scores.

## V. CONCLUSION AND FUTURE SCOPE

The Phishing URL Detection System effectively classifies URLs as legitimate or phishing, enhancing online security. By leveraging machine learning, particularly the Random Forest model, the system accurately detects phishing attempts based on URL features, providing real-time classification. Pre-processing steps such as handling class imbalance, optimizing hyperparameters, and feature scaling significantly improved the model's performance.
Class distribution analysis revealed dataset imbalances, which were addressed using appropriate techniques to ensure unbiased training and evaluation. Evaluation metrics, including precision, recall, and F1-score, confirmed the model's effectiveness in accurately identifying phishing URLs. The integration of a Python Flask backend with a React frontend enables seamless user interaction, allowing users to input URLs and receive instant classification results. With efficient deployment, the system provides a reliable tool to protect users from phishing threats.

**Future Scope**

To enhance the system's capabilities, future improvements can focus on:

- **Expanding the Dataset**: Incorporating larger and more diverse datasets to improve model generalization.
- **Advanced Machine Learning Models**: Exploring deep learning techniques such as LSTMs or transformer-based models for improved accuracy.
- **Real-Time Threat Intelligence**: Integrating APIs for real-time phishing URL detection and continuously updating the model.
- **Behavioral Analysis**: Including user interaction patterns and webpage behavior to strengthen detection.
- **Browser Extension Integration**: Developing a browser extension to provide instant phishing alerts while users browse the web.

Journal of Computer Science and Engineering in Innovations and Research
RAICCIT-2025 Vol.-1, Issue- 2, January - June 2025 ISSN: 3049-1762 (Online)

- **Mobile Application Support**: Extending the system's capabilities to mobile platforms for broader accessibility.

By implementing these enhancements, the Phishing URL Detection System can evolve into a more robust, scalable, and efficient cybersecurity solution.

**Data Availability**

The dataset used in this study consists of labeled URLs categorized as phishing or legitimate. The dataset was preprocessed to handle missing values, map target labels, and select significant features for training the machine learning models. The training and testing dataset distribution was maintained at a 75:25 ratio to ensure a balanced evaluation. Due to security and privacy concerns, the dataset cannot be publicly shared. However, researchers interested in accessing the dataset may request permission from the original data providers.

Study Limitations

The primary limitation of this study is the class imbalance in the dataset, where legitimate URLs significantly outnumber phishing URLs. This imbalance could affect the model's performance by biasing it toward the majority class. To mitigate this, techniques such as oversampling, undersampling, or synthetic data generation were employed. Another limitation is the reliance on static URL-based features, which may not capture dynamically generated phishing attacks. Additionally, while Random Forest demonstrated the highest performance, the model may still require further tuning or integration with real-time threat intelligence sources to enhance accuracy against evolving phishing tactics.

**Conflict of Interest**

The authors declare that they have no financial, personal, or other relationships that could inappropriately influence or be perceived to influence the work presented in this study. No competing interests exist regarding the research, methodology, or findings of this study.

**Funding Source**

none

**Authors' Contributions**

Rishikesh conducted a literature review and conceptualized the study.

Sreetama Das contributed to protocol development, obtained ethical approval, handled patientrecruitment, and performed data analysis.

Souparna Paul drafted the initial version of the manuscript.

Chayan Ghosh assisted in data analysis and manuscript preparation.

All authors reviewed and edited the manuscript and approved the final version of the manuscript.

## ACKNOWLEDGMENT

The authors would like to express their gratitude to all individuals and organizations that contributed to this study. Special thanks to the providers of the dataset used for training and evaluating the phishing URL detection model. The authors also appreciate the support of colleagues who offered valuable insights and feedback during the research process.

## REFERENCES

[1] T. Kim, N. Park, J. Hong, and S. W. Kim, "Phishing URL detection: A network-based approach robust to evasion," Proc. 2022 ACM SIGSAC Conf. Comput. Commun. Secur., pp. 1769-1782, Nov. 2022.R. Solanki, "Principle of Data Mining," McGraw-Hill Publication, India, pp. 386-398, 1998.

[2] S. H. Ahammad, S. D. Kale, G. D. Upadhye, S. D. Pande, E. V. Babu, A. V. Dhumane, and M. D. K. J. Bahadur, "Phishing URL detection using machine learning methods," Adv. Eng. Softw., vol. 173, p. 103288, 2022.

[3] M. Mia, D. Derakhshan, and M. M. A. Pritom, "Can Features for Phishing URL Detection Be Trusted Across Diverse Datasets? A Case Study with Explainable AI," Proc. 11th Int. Conf. Netw., Syst., Secur., pp. 137-145, Dec. 2024.

[4] J. Kumar, "Enhanced phishing URL detection using hybrid feature-based machine learning method," IET Conf. Proc. CP832, vol. 2023, no. 5, pp. 186-192, Jul. 2023.

[5] K. Sruthi, "Real-time phishing threat detection using lexical URL features and machine learning techniques," Ph.D. dissertation, [Univ. Name], 2023.

Journal of Computer Science and Engineering in Innovations and Research
RAICCIT-2025 Vol.-1, Issue- 2, January - June 2025 ISSN: 3049-1762 (Online)

[6] R. Madhubala, N. Rajesh, L. Shaheetha, and N. Arulkumar, "Survey on malicious URL detection techniques," Proc. 6th Int. Conf. Trends Electron. Informatics (ICOEI), pp. 778-781, Apr. 2022.

[7] F. Ji, K. Lee, H. Koo, W. You, E. Choo, H. Kim, and D. Kim, "Evaluating the effectiveness and robustness of visual similarity-based phishing detection models," arXiv preprint arXiv:2405.19598, 2024.

[8] R. Abdillah, Z. Shukur, M. Mohd, and T. M. Z. Murah, "Phishing classification techniques: A systematic literature review," IEEE Access, vol. 10, pp. 41574-41591, 2022.

[9] R. J. Van Geest, G. Cascavilla, J. Hulstijn, and N. Zannone, "The applicability of a hybrid framework for automated phishing detection," Comput. Secur., vol. 139, p. 103736, 2024.

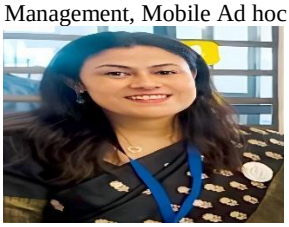
[10] M. Hammed and J. Soyemi, "Classification of phishing attacks in social media using associative rule mining augmented with firefly algorithm," Int. J. Comput. Sci. Eng., vol. 6, no. 6, pp. 1-10, 2023.

## Authors Profile

**Author-1** Dr. Chandrima Chakrabarti is an Associate Professor and Head of the Department of the Department of Computer Science and Engineering (Data Science), Narula Institute of Technology, Kolkata, India. She is an accomplished author. Dr. Chakrabarti received her Ph.D (Technology) degree in Computer Science and Engineering from Maulana Abul Kalam Azad University of Technology, West Bengal, India in 2022. Earlier she received M. Tech. in Computer Science & Engineering and B. Tech. in Information Technology, degrees from Maulana Abul Kalam Azad University of Technology (formerly West Bengal University of Technology), India. She has around 16 years of experience in academics and research. Dr. Chakrabarti has published more than 25 research articles in international journals, conferences, book chapters including 1 Indian Patent. Her research interests include database management systems, data science and its applications, reliable communication in Delay Tolerant Networks, Opportunistic Networks, Applications of Mobile Computing in Disaster Management, Mobile Ad hoc Networks. Dr. Chakrabarti is an active member of professional organizations like IEEE.

Author2

Ms. Bingshati Mondal Assistant professor of Computer Science and Engineering department at Narula Institute of Technology. She has done B.Tech in 2017 from MAKAUT and M.Tech in 2019 from MAKAUT, W.B. Ms. Mondal has published more than 10 research articles in international journals, conferences, book chapters. Her research interests include database management systems, Computer Networks, Applications of Mobile Computing in Disaster Management, Mobile Ad hoc Networks etc.



Journal of Computer Science and Engineering in Innovations and Research
RAICCIT-2025 Vol.-1, Issue- 2, January - June 2025 ISSN: 3049-1762 (Online)

Author3 Ms. Jayita Pal earned her B.Tech degree in Information Technology from WBUT (MAKAUT WB) and her M.Tech in Distributed and Mobile Computing from Jadavpur University. With 9 years of teaching and knowledge sharing experience in the education sector, she is currently working as an Assistant Professor in the CSE department at Narula Institute of Technology. Her research interests include Machine Learning and Image Processing. Her deep passion for technology education and research inspires students and drives academic excellence.

**Author-4 Rishikesh** is currently pursuing a B.Tech in Computer Science and Engineering at Narula Institute of Technology. With a strong passion for technology, he has developed expertise in areas such as web development, artificial intelligence, cybersecurity, and data structures. He has worked on multiple projects, including a phishing URL detection system, and actively explores innovative solutions in machine learning and network security. He is constantly expanding his knowledge through hands-on projects, competitive coding, and research in emerging fields of computer science.

**Author-5 Sreetama Das** is a B.Tech student in Computer Science and Engineering at Narula Institute of Technology. She is deeply interested in software development, UI/UX design, and data science. She has worked on various projects, including web and mobile app development, focusing on user-friendly and efficient design. Apart from academics, she enjoys coding challenges, participating in hackathons, and exploring the intersection of AI and human-computer interaction.

**Author-6 Souparna Paul** is pursuing a B.Tech in Computer Science and Engineering at Narula Institute of Technology. He is passionate about artificial intelligence, deep learning, and cloud computing. He has worked on projects related to AI-driven automation and cloud-based applications. With a keen interest in research, he constantly explores new trends in AI and big data analytics. In his free time, he enjoys contributing to open-source projects and experimenting with new technologies.

Author's formal photo

**Author-7 Chayan Ghosh** is a B.Tech student in Computer Science and Engineering at Narula Institute of Technology. He has a strong interest in software engineering, system design, and full-stack development. He has worked on several projects involving large-scale web applications and backend optimizations. He enjoys building scalable systems, working with databases, and learning about DevOps. Outside of coding, he is enthusiastic about tech blogging and knowledge sharing in the developer community.