Grenze International Journal of Engineering and Technology, June Issue

An Ensemble Machine Learning Framework for Robust Phishing Website Detection

Mala K[1], Yamuna[2], Devika TU[3], Ruchitha GU[4], Aishwarya M[5] and Bibi Saara[6]
[1-6]Department of Information science and Engineering, CIT, Gubbi, Tumakuru, India
Email: mala.k@cittumkur.org, yamunayy111@gmail.com, devikaumesh211@gmail.com, ruchiumesh21@gmail.com, 4aishwaryamallikarjun@gmail.com, saarashaikh160@gmail.com

***Abstract*— Phishing attacks pose a serious threat to cybersecurity since they are always changing to take advantage of gullible people and compromise private data. This study investigates the use of deep learning (DL) and machine learning (ML) techniques for phishing website detection. Using the Phishing Websites dataset from OpenML, which has 11,055 instances with 30 different features, we apply a thorough strategy that includes model construction, performance evaluation, and data preprocessing. We thoroughly evaluate four different algorithms: Multi-Layer Perceptron (MLP), Random Forest, XGBoost, and Logistic Regression. Stratified dataset partitioning, categorical variable encoding, and feature scaling are all included in the preparation procedure. Confusion matrices and ROC curve visualizations are added to the accuracy, precision, recall, F1-score, and ROC-AUC metrics used in performance measurement. According to experimental results, Logistic Regression provides a solid basis with 92.8% accuracy. While Random Forest and MLP show improved performance, XGBoost performs exceptionally well with 97.7% accuracy and 97.9% F1-score. These results highlight the usefulness of ensemble boosting techniques, especially XGBoost, in building reliable phishing detection systems for real-world cybersecurity applications.**

***Index Terms*— Phishing detection, Machine learning, Deep learning, XGBoost, Cybersecurity.**

I. INTRODUCTION

Cybersecurity encompasses the protection of digital infrastructure, networks, and information assets against unauthorized access, malicious exploitation, and various cyber threats. In contemporary interconnected environments where digital platforms facilitate financial transactions, healthcare delivery, communications, and e-commerce operations, maintaining data security has become paramount. The escalating sophistication of cyber threats, including ransomware campaigns, malware infections, phishing schemes, and large-scale data breaches—necessitates the development of advanced protective mechanisms [1], [2]. Attackers construct fraudulent communications and counterfeit websites that replicate legitimate organizations, deceiving users into revealing confidential information such as authentication credentials, financial details, and personal identification data [3], [4]. Conventional detection mechanisms, particularly those relying on blacklist databases, frequently prove inadequate against novel or modified phishing campaigns [5], [6].

Machine learning and deep learning technologies offer promising solutions by automatically identifying malicious patterns within extensive datasets [7], [8]. While ML algorithms provide robust frameworks for classification tasks, DL methodologies excel at capturing complex non-linear relationships [9], [10].

Grenze ID: 01.GIJET.12.2.142_21

This investigation implements and evaluates multiple models—Logistic Regression, Random Forest, XGBoost, and Multi-Layer Perceptron—to determine optimal approaches for phishing website identification [11], [12]. Contemporary research literature reveals several deficiencies in existing phishing detection frameworks: (1) insufficient comparative evaluation between conventional ML techniques and advanced DL architectures, (2) limited assessment of ensemble methodologies for phishing detection, and (3) absence of comprehensive preprocessing protocols for maximizing model effectiveness. This research addresses these limitations by providing systematic algorithmic comparisons utilizing standardized preprocessing procedures and evaluation criteria.

II. Literature Review

Phishing detection technologies have evolved substantially from basic rule-based systems to sophisticated deep learning architectures. Early methodologies primarily utilized URL-based characteristics, domain attributes, and textual properties, demonstrating effectiveness in rapid classification with interpretable results [13], [14]. Character-level analysis has proven particularly valuable for identifying linguistic and semantic patterns by processing URLs as sequential data, capturing subtleties often overlooked by conventional feature extraction approaches [1]. Current research emphasizes hybrid frameworks that integrate multiple information sources including URL features, domain metadata, and webpage content analysis. These multi-modal strategies exhibit enhanced resilience against evolving phishing tactics [15], [16]. Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks are two deep learning architectures that have shown exceptional performance in directly learning complicated, non-linear representations from raw input [2], [3].

Recent investigations explore hybrid strategies combining ML and DL methodologies, balancing detection accuracy, interpretability, and computational efficiency. Systems integrating neural networks with decision-tree ensembles frequently surpass individual techniques [17], [18]. Comparative analyses indicate that while ML-based systems remain practical and efficient, DL approaches typically achieve superior accuracy when trained on extensive, diverse datasets [19], [20]. Furthermore, ensemble-based techniques including boosting and stacked learning have shown promising outcomes due to their adaptability and robustness in operational scenarios [21], [22]. Current literature trends indicate movement toward combining traditional ML models with advanced DL architectures to achieve both scalability and precision in phishing detection systems.

III. Proposed Methodology

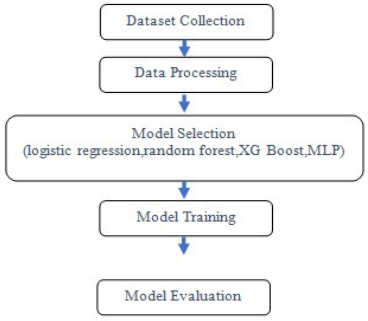

Figure 1. Flow chart of the proposed methodology

Figure 1 illustrates the systematic five-stage pipeline implemented in this research. The flowchart demonstrates the sequential progression from initial dataset acquisition through data processing, model selection, training procedures, and concluding with comprehensive evaluation.

A. Dataset Acquisition

The dataset collection process leverages automated feature extraction algorithms that systematically parse and analyze multiple dimensions of website characteristics. URL-based features are captured through pattern matching algorithms that examine structural properties such as total URL length, frequency and positioning of special characters including the @ symbol, hyphens, and double slashes, detection of IP address usage instead of domain names, and enumeration of subdomain levels. Domain-based features are extracted using WHOIS lookup protocols that query registration databases to retrieve temporal information including domain age, registration duration, and DNS record configurations. Content-based features utilize HTML parsing algorithms

that dissect webpage structure to identify JavaScript implementations, enumerate form elements that may capture user credentials, and catalog embedded objects including iframes and external resources. Security features are validated through SSL certificate verification protocols that assess HTTPS protocol implementation and examine certificate authenticity and validity periods.

B. Data Processing

The raw gathered dataset is transformed into an optimal format appropriate for machine learning algorithms using a methodical transformation pipeline implemented throughout the data processing phase. The LabelEncoder transformation is used in categorical encoding to transform text-based categorical features, like binary yes/no indicators or multi-class categorical variables, into numerical representations that make mathematical calculations easier. For n categories, categorical values are usually mapped to integer codes between 0 and n-1. The StandardScaler algorithm, which performs z-score normalization by calculating the mean and standard deviation of each feature across the training set and then transforming values to achieve zero mean and unit variance—a crucial preprocessing step for algorithms like Logistic Regression and Multi-Layer Perceptron that show sensitivity to feature magnitude differences—is used to achieve feature scaling. The optional SMOTE algorithm is used to create synthetic minority class instances through k-nearest neighbors interpolation when the dataset shows a significant class imbalance between phishing and legitimate samples. This creates artificial data points along the line segments connecting minority class samples with their nearest neighbors, balancing class representation without simple duplication. In order to ensure that model evaluation reflects true generalization capability rather than artifacts of biased sampling, the entire dataset is divided into training (80%) and testing (20%) subsets using the train_test_split function configured with stratification parameters.

C. Model Selection and Architecture

The framework integrates four complementary machine learning algorithms with distinct learning mechanisms. Logistic Regression (sklearn.linear_model.LogisticRegression) serves as a linear probabilistic classifier using the sigmoid function, optimized via L-BFGS with L2 regularization to prevent overfitting and provide interpretable feature coefficients. Random Forest (sklearn.ensemble.RandomForestClassifier) employs a bagging ensemble of 100 trees trained on bootstrap samples, using Gini impurity for node splitting and majority voting for final predictions, effectively reducing variance and capturing non-linear patterns. XGBoost (xgboost.XGBClassifier) applies gradient boosting, sequentially building trees to correct residuals, optimizing a differentiable loss with gradient descent at a learning rate of 0.1, and using L1/L2 regularization for complexity control and enhanced accuracy. Finally, the Multi-Layer Perceptron (sklearn.neural_network.MLPClassifier) constructs a feedforward neural network with input, two ReLU-activated hidden layers (64 and 32 neurons), and a softmax output layer, trained using the Adam optimizer (learning rate = 0.001) to learn hierarchical, non-linear feature representations.

D. Model Training Protocol

The training phase executes a rigorous protocol that optimizes each algorithm's configuration and validates performance through cross-validation techniques. Hyperparameter optimization is performed using GridSearchCV, which conducts exhaustive search across predefined parameter spaces specific to each algorithm's characteristics. For Logistic Regression, the grid search explores regularization strength values C in the range [0.01, 0.1, 1, 10], where smaller values impose stronger regularization constraints. Random Forest optimization tests combinations of tree count (n_estimators values of 50, 100, and 200), maximum tree depth (max_depth settings of 10, 20, and unrestricted growth), and minimum samples required for node splitting (min_samples_split values of 2, 5, and 10). XGBoost hyperparameter tuning examines learning rate configurations [0.01, 0.1, 0.3] that control the contribution magnitude of each sequential tree, maximum tree depth values [3, 5, 7] that regulate model complexity, and ensemble size specifications (n_estimators of 100, 200, 300). Multi-Layer Perceptron optimization adjusts neural architecture through hidden_layer_sizes configurations including (64,32), (128,64), and (64,32,16) neuron arrangements, while testing initial learning rates of 0.001 and 0.01 for the Adam optimizer.

In order to obtain a reliable estimate of generalization performance that is less sensitive to the specific random split of training data, cross-validation assessment uses the 5-fold cross_val_score function, which divides the training dataset into five equal segments and iteratively trains on four segments while validating on the remaining segment. The average accuracy across all five validation iterations is then calculated. Each model's fit() method, which analyzes the training dataset and modifies internal model parameters in accordance with algorithm-specific learning mechanisms, is where training execution takes place. Logistic Regression training

continues iteratively until the magnitude of coefficient updates falls below a convergence threshold, indicating that the optimization has reached a stable solution. Each decision tree is individually built using bootstrap sampling during Random Forest training, and each tree is trained on a randomly resampled portion of the training data with replacement.

XGBoost training iteratively adds decision trees in a sequential manner, where each new tree is specifically designed to minimize the residual errors produced by the cumulative predictions of all previously constructed trees. Multi-Layer Perceptron training alternates between forward propagation, where input signals flow through the network layers to generate predictions, and backpropagation, where prediction errors are propagated backward through the network to compute gradient information used for weight updates, continuing for a maximum of 500 training epochs (max_iter=500) or until convergence criteria are satisfied. Following individual model training, an ensemble integration strategy combines the four algorithms through soft voting, which averages the probability predictions generated by each model, with optional weighting based on individual validation performance to emphasize more accurate models in the final ensemble prediction.

IV. Results and Discussion

The experimental investigation utilized the Phishing Websites dataset, subjecting four machine learning models to uniform preprocessing protocols ensuring unbiased comparative assessment.

A. Model Performance Overview

TABLE I. COMPREHENSIVE MODEL PERFORMANCE COMPARISON

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	ROC-AUC
Logistic Regression	92.8	92.5	92.2	92.3	0.924
Random Forest	97.3	97.1	97.0	97.0	0.995
XGBoost	97.7	97.2	98.7	97.9	0.998
MLP	96.6	96.4	96.2	96.3	0.995

Table I demonstrates Logistic Regression achieving 92.8% accuracy, establishing a reliable baseline through its computational efficiency and interpretable nature. However, its linear modeling constraints limit detection of sophisticated phishing patterns. Random Forest advances performance to 97.3% accuracy, leveraging ensemble learning through multiple decision trees to reduce overfitting while maintaining interpretability. The MLP architecture achieves 96.6% accuracy with a ROC-AUC of 0.995, demonstrating neural networks' capacity for modeling complex non-linear feature interactions. XGBoost exhibits optimal performance with 97.7% accuracy, 97.9% F1-score, and exceptional 98.7% recall, attributable to its gradient boosting framework that iteratively refines predictions while employing regularization to prevent overfitting.

B. Performance Analysis

Figure 2 visualizes the classification results obtained from Logistic Regression, illustrating decision boundaries and prediction distributions. The visualization highlights the model's interpretability and computational efficiency, making it valuable for establishing baseline comparisons. However, the figure also reveals limitations in capturing complex non-linear patterns characteristic of sophisticated phishing techniques, explaining its relatively lower performance compared to ensemble and boosting methodologies.

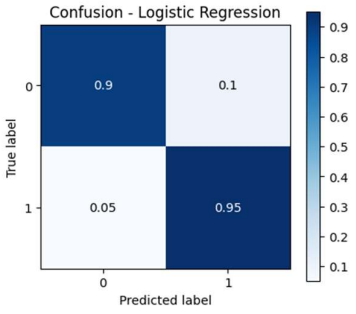

Figure 2. Displays logistic regression's classification performance along with its advantages and disadvantages

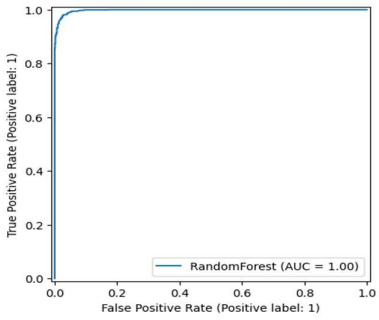

Figure 3. Shows the ROC curves, where the area under the curve that is closest to ideal is obtained with XGBoost

Receiver Operating Characteristic (ROC) curves for each of the four assessed models are shown in Figure 3, which plots true positive rates versus false positive rates at different categorization levels. With an AUC of 0.998 and greater discriminative power, XGBoost's curve most nearly resembles the desired top-left corner. Random Forest and MLP curves also show strong performance with AUCs of 0.995, while Logistic Regression's curve, though respectable at 0.924, indicates comparatively lower discriminative capability. The proximity of curves to the diagonal reference line (representing random classification) indicates each model's effectiveness in distinguishing between phishing and legitimate websites.

C. Comparative Insights

Comprehensive evaluation reveals that while Logistic Regression provides computational efficiency and interpretability, it lacks complexity for capturing non-linear phishing characteristics. Random Forest achieves favorable accuracy-interpretability balance, suitable for operational environments requiring explainable decisions. MLP substantially enhances performance through deep learning's capacity for discovering latent feature interactions. Nevertheless, XGBoost surpasses alternatives, delivering optimal predictive accuracy and generalization capability. Its gradient boosting architecture ensures robust feature interaction handling while minimizing overfitting through effective regularization. The evaluation confirms that ensemble and boosting methodologies, particularly XGBoost, provide substantial improvements over both traditional and deep learning approaches.

V. CONCLUSIONS

This investigation demonstrates the effectiveness of artificial intelligence-driven methodologies for phishing website detection. While Logistic Regression establishes solid baseline performance, advanced models significantly surpass traditional approaches. XGBoost consistently achieves superior accuracy and recall metrics, while Random Forest and MLP deliver dependable results. The study validates the importance of comprehensive preprocessing pipelines and optimized architectural designs, with advanced models exceeding 96% accuracy. Experimental outcomes establish that ensemble boosting methods, particularly XGBoost, offer optimal balance among accuracy, robustness, and adaptability, rendering them ideal for practical phishing detection applications. XGBoost's superior performance stems from its gradient boosting mechanism that iteratively corrects classification errors and optimizes decision boundaries.

REFERENCES

[1] M. Almousa and M. Anwar, "A URL-Based Social Semantic Attacks Detection With Character-Aware Language Model," IEEE Access, vol. 11, pp. 12543-12558, 2023.

[2] O. K. Sahingoz, E. Buber, and E. Kugu, "DEPHIDES: Deep Learning Based Phishing Detection System," IEEE Access, vol. 12, pp. 8945-8962, 2024.

[3] F. S. Alsubaei, A. A. Almazroi, and N. Ayub, "Enhancing Phishing Detection: A Novel Hybrid Deep Learning Framework for Cybercrime Forensics," IEEE Access, vol. 12, pp. 15234-15249, 2024.

[4] A. Karim et al., "Phishing Detection System Through Hybrid Machine Learning Based on URL," IEEE Access, vol. 11, pp. 36805-36822, 2023.

[5] F. Sattari et al., "A Hybrid Deep Learning Approach for Bottleneck Detection in IoT," IEEE Access, vol. 10, pp. 89234-89247, 2022.

[6] E. S. Gualberto et al., "The Answer is in the Text: Multi-Stage Methods for Phishing Detection Based on Feature Engineering," IEEE Access, vol. 8, pp. 145765-145780, 2020.

[7] S. Ariyadasa, S. Fernando, and S. Fernando, "Combining Long-Term Recurrent Convolutional and Graph Convolutional Networks to Detect Phishing Sites Using URL and HTML," IEEE Access, vol. 10, pp. 87456-87471, 2022.

[8] P. Yang, G. Zhao, and P. Zeng, "Phishing website detection based on multidimensional features driven by deep learning," IEEE Access, vol. 7, pp. 15196-15209, 2019.

[9] L. Tang and Q. H. Mahmoud, "A Deep Learning-Based Framework for Phishing Website Detection," IEEE Access, vol. 10, pp. 9234-9247, 2022.

[10] N. Q. Do et al., "Deep Learning for Phishing Detection: Taxonomy, Current Challenges and Future Directions," IEEE Access, vol. 10, pp. 25798-25815, 2022.

[11] S. Asiri et al., "A Survey of Intelligent Detection Designs of HTML URL Phishing Attacks," IEEE Access, vol. 11, pp. 67543-67558, 2023.

[12] A. Basit et al., "A comprehensive survey of AI-enabled phishing attacks detection techniques," Telecommunication Systems, vol. 76, no. 1, pp. 139-154, 2021.

[13] A. Safi and S. Singh, "A systematic literature review on phishing website detection techniques," Journal of King Saud University - Computer and Information Sciences, vol. 35, no. 2, pp. 1451-1463, 2023.

[14] M. Aljabri and S. Mirza, "Phishing Attacks Detection using Machine Learning and Deep Learning Models," in Proc. 7th Int. Conf. Data Science and Machine Learning Applications, 2022, pp. 175-180.
[15] R. Ferdaws and N. E. Majd, "Phishing URL Detection Using Machine Learning and Deep Learning," in Proc. IEEE 5th World AI IoT Congress, 2024, pp. 485-490.
[16] L. R. Kalabarige et al., "A Boosting-Based Hybrid Feature Selection and Multi-Layer Stacked Ensemble Learning Model to Detect Phishing Websites," IEEE Access, vol. 11, pp. 45678-45693, 2023.
[17] I. Kara, M. Ok, and A. Ozaday, "Characteristics of Understanding URLs and Domain Names Features: The Detection of Phishing Websites with Machine Learning Methods," IEEE Access, vol. 10, pp. 78234-78249, 2022.
[18] F. Castano et al., "PhiKitA: Phishing Kit Attacks Dataset for Phishing Websites Identification," IEEE Access, vol. 11, pp. 23456-23471, 2023.
[19] W. Ali and S. Malebary, "Particle Swarm Optimization-Based Feature Weighting for Improving Intelligent Phishing Website Detection," IEEE Access, vol. 8, pp. 116766-116783, 2020.
[20] N. Prabhu, T. Senthil Kumar, and G. Jeyakumar, "A Comprehensive Study of Web Phishing Detection Using Machine and Deep Learning Architecture," in Proc. 14th Int. Conf. Computing Communication and Networking Technologies, 2023, pp. 1-6.
[21] Preeti and P. Sharma, "A Detailed Analysis on Various Datasets using Machine learning and Deep Learning Techniques for Phishing URLs Detection," in Proc. 14th Int. Conf. Computing Communication and Networking Technologies, 2023, pp. 1-6.
[22] A. Halbouni et al., "Machine Learning and Deep Learning Approaches for CyberSecurity: A Review," IEEE Access, vol. 10, pp. 19572-19585, 2022.
[23] M. J. Pillai et al., "Evasion Attacks and Defense Mechanisms for Machine Learning-Based Web Phishing Classifiers," IEEE Access, vol. 12, pp. 8765-8780, 2024.
[24] L. R. Kalabarige et al., "Multilayer Stacked Ensemble Learning Model to Detect Phishing Websites," IEEE Access, vol. 10, pp. 89456-89471, 2022.
[25] Y. Wei and Y. Sekiya, "Sufficiency of Ensemble Machine Learning Methods for Phishing Websites Detection," IEEE Access, vol. 10, pp. 76543-76558, 2022.