ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

A Phishing Detection System by Application of Multi-Classifiers Using E-Voting Method

Emmanuel Ugochukwu Ndibuisi, Wasiu Oladimeji Ismaila, Olufemi Olayanju Awodoye, Folasade Muibat Ismaila

Department of Computer Science Faculty of Computing and Informatics Ladoke Akintola University of Technology Ogbomoso.

DOI: https://doi.org/10.51583/IJLTEMAS.2025.140600083

Received: 04 May 2025; Accepted: 08 May 2025; Published: 19 July 2025

Abstract-Phishing remains a major global threat, exploiting human weaknesses to access sensitive data. Existing detection methods often struggle with high false positives and fail to adapt to evolving phishing tactics. This study proposes a phishing detection system that combines Support Vector Machine (SVM), Feed Forward Neural Network (FFNN), and Extreme Learning Machine (ELM) using a weighted voting approach. A cleaned and normalized dataset of URLs was used, with dimensionality reduction via PCA. The models were evaluated using metrics like accuracy, sensitivity, and detection time in MATLAB (R2023a).

Results show that SVM achieves the best performance, with the lowest false positive rate (1.79%), highest precision (97.98%), and accuracy (97.75%). FFNN offers balanced performance, while ELM is the fastest but less accurate. The weighted voting mechanism consistently identifies phishing as the dominant class, enhancing detection accuracy. Overall, combining the three models improves robustness, with SVM emerging as the most effective classifier.

Keywords: Phishing attacks, detection system, Support Vector Machine, Feed Forward Neural Network, Extreme Learning Machine, voting method.

I. Introduction

Phishing is a significant and growing threat in the digital world, targeting users through deceptive emails, websites, or messages that appear to be from legitimate sources. Cybercriminals often use phishing attacks to steal sensitive information such as passwords, credit card details, and personal identities. Despite advancements in cybersecurity, phishing continues to evolve, making it increasingly difficult to detect. This challenge has led to the development of sophisticated phishing detection systems that use machine learning, artificial intelligence, and pattern recognition techniques to identify and block phishing attempts. Understanding how phishing attacks operate and evolve is essential for building effective detection systems that can keep pace with these threats [1].

The importance of phishing detection systems lies in their ability to safeguard individuals and organizations from financial losses, data breaches, and reputational damage. Traditional methods like blacklists and spam filters are limited, as they rely on known attack signatures and can be bypassed by new, sophisticated phishing techniques. Modern phishing detection systems incorporate machine learning models that analyze the content, structure, and source of emails or websites to detect suspicious patterns. By using features such as URL length, domain reputation, and language cues, these systems can identify potential phishing attempts in real time. Research into these systems focuses on enhancing their accuracy and reducing false positives, a key challenge in cybersecurity [2].

Furthermore, phishing detection systems are becoming increasingly vital as remote work, e-commerce, and online banking grow, expanding the attack surface for cybercriminals. The integration of artificial intelligence (AI) and natural language processing (NLP) in phishing detection helps analyze complex and context-specific threats, improving detection rates. As phishing attacks become more personalized and harder to detect, these systems must adapt to new tactics, such as spear-phishing, where attackers target specific individuals or organizations. This ongoing need for innovation in phishing detection drives research into new techniques, ensuring that defenses remain robust against emerging threats [3].

Phishing attacks pose a serious threat to internet users by attempting to deceive them into revealing sensitive information, such as passwords and financial data. With the rise of increasingly sophisticated phishing attempts, traditional detection methods like blacklists and signature-based systems have become inadequate. Machine learning techniques have gained attention as effective tools for phishing detection due to their ability to analyze large amounts of data and identify hidden patterns. By using various classification algorithms, these techniques can detect phishing attempts based on features extracted from emails, URLs, and web pages. The development of a phishing detection system using machine learning classification methods is, therefore, a promising area of research to improve online security [4].

The core advantage of using machine learning in phishing detection is the ability to detect emails or websites as either phishing or legitimate based on patterns learned from data. Supervised learning techniques, such as Decision Trees, Support Vector Machines (SVM), and Random Forests, have proven effective in identifying phishing attacks. These models are trained on labeled datasets, where the input features might include URL structure, domain reputation, email metadata, and content analysis. By learning from

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

both legitimate and phishing examples, the system can generalize and detect new, previously unseen phishing attempts. The focus of this research lies in improving the accuracy, precision, and recall of these machine learning models to minimize false positives and false negatives [5].

Despite the potential of machine learning-based phishing detection systems, challenges remain in ensuring they are robust and adaptable to evolving attack strategies. Attackers are constantly changing their tactics, creating new phishing techniques that are harder to detect. Research into phishing detection systems using machine learning classification must address these challenges by developing adaptive models that can quickly learn from new data and respond to emerging threats. Additionally, the integration of deep learning and natural language processing (NLP) techniques can enhance the system's ability to understand and detect more complex phishing attempts, such as those involving personalized or highly targeted spear-phishing. This ongoing exploration of machine learning approaches is essential for building more reliable phishing detection systems [6].

In this study, the use of selected classifiers in phishing detection systems, combined with the E-voting method, will be explored, which is believed to offer a robust and adaptive approach to enhancing detection accuracy. By leveraging the strengths of multiple classifiers and aggregating their decisions through an E-voting system, this method improves the system's ability to identify phishing attacks while reducing false positives.

II. Literature Review

Phishing attacks have become increasingly sophisticated and prevalent in recent years, posing significant threats to individuals and organizations alike. [7] conducted a comprehensive review of phishing attack techniques, highlighting the evolution from simple email-based attacks to more complex methods involving social engineering and targeted spear-phishing. The study by [8] emphasized the importance of user awareness and education in combating phishing threats, proposing a multi-layered approach to phishing defense. Few studies analyzed various machine learning techniques for phishing detection, comparing the effectiveness of different algorithms in identifying malicious websites and emails [9]. Lastly, a study explored the use of natural language processing techniques to improve the accuracy of phishing detection systems, particularly in analyzing email content and URL structures [10].

The effectiveness of various anti-phishing techniques has been a subject of extensive research in the cybersecurity community. The study proposed an associative classification approach for phishing detection, demonstrating improved accuracy compared to traditional classification methods [11].A study focused on the application of deep learning techniques in phishing detection, showcasing the potential of neural networks in identifying complex patterns in phishing attempts [12]. [13] developed a real-time phishing detection system using natural language processing and machine learning algorithms, achieving high accuracy in identifying malicious URLs. The work of [14] explored the use of hybrid feature selection techniques to enhance the performance of phishing detection systems, addressing the challenge of high-dimensional data in cybersecurity applications. Finally, [15] conducted a comparative analysis of various ensemble learning methods for phishing detection, highlighting the benefits of combining multiple classifiers to improve overall system performance.

Recent advancements in phishing detection have focused on leveraging emerging technologies and addressing evolving attack vectors .[16] introduced VisualPhishNet, a novel approach using computer vision techniques to detect zero-day phishing websites based on visual similarity to legitimate sites. The study by [17] explored the use of deep learning models for email phishing detection, demonstrating superior performance compared to traditional machine learning approaches. [18] proposed a blockchain-based framework for phishing detection and prevention, leveraging the decentralized nature of blockchain technology to enhance security and trust in online transactions. Researchers like [19] have investigated the application of federated learning in phishing detection, addressing privacy concerns while enabling collaborative model training across multiple organizations. Lastly, the work of [20] focused on developing phishing detection systems for emerging platforms such as mobile devices and Internet of Things (IoT) environments, highlighting the need for adaptive and context-aware solutions in the rapidly evolving digital landscape.

G. Related Works

[9] introduced a new feature selection framework for machine learning-based phishing detection systems, named the Hybrid Ensemble Feature Selection (HEFS). In the initial phase, the HEFS employs a novel Cumulative Distribution Function gradient (CDF-g) algorithm to generate primary feature subsets, which are then passed through a data perturbation ensemble to create secondary feature subsets. In the second phase, a function perturbation ensemble is used to derive a baseline feature set from the secondary subsets. Experimental results indicate that HEFS performs optimally when combined with a Random Forest classifier, achieving a 94.6% accuracy in distinguishing phishing from legitimate websites using only 20.8% of the original features. Additionally, HEFS demonstrated strong results when tested on a popular phishing dataset from the University of California Irvine (UCI) repository, making it a highly effective feature selection method for phishing detection systems.

[21] introduced a URL-based phishing detection system that uses both the URL of the web page and the URL of its source code as features, with Levenshtein Distance applied to calculate string similarity. The approach was enhanced by incorporating a machine learning framework. Given the high accuracy in small sample sizes and binary classification, Support Vector Machine

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

(SVM) was selected as the machine learning model for the system. The system is designed to achieve high accuracy and a low false positive rate when detecting unknown phishing websites.

[4] introduced a straightforward method for developing an effective phishing page detection system. This system utilizes a voting mechanism with multiple models, where each model is trained by introducing controlled noise into a randomly selected subset of features. Comprehensive experiments were conducted using real datasets to assess the system's resilience against various evasive strategies. The findings indicate that the proposed system performs comparably to the traditional machine learning model in the absence of adversarial attacks. However, it demonstrates greater robustness against evasion attacks compared to the native model.

[5] Mughaid et al. (2022) explored machine learning algorithms and technical solutions for mitigating phishing attacks, along with discussing essential user awareness knowledge. They proposed a detection model using machine learning techniques, training and validating it on three different datasets to detected emails as phishing or non-phishing based on text characteristics and other features. The study compared the performance of various algorithms across these datasets, finding that models using a higher number of features generally achieved more accurate and efficient results. Among the tested algorithms, the boosted decision tree demonstrated the best performance, achieving accuracies of 0.88, 1.00, and 0.97 on the respective datasets.

[1] developed a hybrid classification model called the Intelligent Phishing Detection System (IPDS) using a combination of the convolutional neural network (CNN) and long short-term memory (LSTM) algorithm. The model was trained on 1 million URLs and over 10,000 images. The sensitivity of the model was assessed by considering various factors, including the type of feature, number of misclassifications, and data split challenges. Extensive experimental analysis was performed to evaluate the IPDS's effectiveness in detecting phishing websites and attacks on large datasets. The results demonstrated an accuracy rate of 93.28% with an average detection time of 25 seconds.

[22] conducted a study to evaluate the effectiveness of machine learning in identifying phishing domains. They created and analyzed four models using different techniques: artificial neural networks (ANNs), support vector machines (SVMs), decision trees (DTs), and random forest (RF). To assess these models, they utilized the UCI phishing domains dataset, which contains uniform resource locator (URL) information. The researchers then compared the performance of these models against each other and existing solutions in the literature. Their results indicated that the random forest-based model demonstrated superior accuracy compared to the other three techniques and surpassed the performance of previously published solutions in this field.

[23] introduced a system for effective phishing website detection, presenting three deep learning-based approaches. Their study compared the performance of long short-term memory (LSTM) and convolutional neural network (CNN) models, and also proposed a hybrid LSTM-CNN approach. The experimental results revealed varying levels of accuracy among the techniques: CNN achieved the highest accuracy at 99.2%, followed by the LSTM-CNN hybrid at 97.6%, and LSTM at 96.8%. Based on these findings, the researchers concluded that the CNN-based model demonstrated the most effective performance in detecting phishing websites among the proposed methods.

[24] conducted a study using a dataset of over 11,000 phishing and legitimate URL attributes from a renowned repository. They applied various machine learning algorithms, including decision trees, random forests, and support vector classifiers, to detect phishing URLs. The researchers proposed a hybrid model called LSD, combining logistic regression, support vector machine, and decision tree techniques with voting mechanisms. They employed feature selection, cross-validation, and hyperparameter optimization to enhance the LSD model's performance. Comparative analysis using multiple evaluation metrics demonstrated that the proposed LSD approach outperformed other models in phishing detection accuracy and efficiency.

[16] developed a hybrid approach for detecting alluring ads phishing attacks using machine learning. The study utilized a dataset of 20,000 website URLs, extracting 22 salient features from each URL, and prepared an additional dataset containing website text for NLP-based evaluation. To address the challenge of phishing websites containing text as images, the researchers extracted and classified text from images. The layered classification model employed various algorithms, including XGBoost, random forest, and multilayer perceptron. Performance evaluation revealed that XGBoost outperformed other models with a maximum accuracy of 94% in training and 91% in testing, while logistic regression and SVM algorithms achieved 87% and 88% accuracy, respectively, in text-based classification.

[25] introduced a novel Convolutional Neural Network (CNN)-based method for detecting phishing websites with high accuracy. The study utilized the PhishTank dataset and created an additional dataset by crawling 10,000 phishing URLs and 10,000 legitimate websites for evaluation. The proposed CNN model, comprising seven layers including pooling, convolutional, and linear layers, outperformed previous state-of-the-art models in distinguishing phishing from legitimate websites. Comparative analysis showed that the CNN-based approach achieved 98.77% accuracy, surpassing k-nearest neighbors (87%), Natural Language Processing (97.98%), Recurrent Neural Network (97.4%), and Random Forest (94.26%) models. The improved performance was attributed to the use of more layers, larger training sizes, and the extraction of additional features from the PhishTank dataset.

[3] reviewed various methods for detecting and preventing phishing attacks. They introduced the Phishcatch algorithm, which effectively identifies phishing emails and alerts users to potential fraud. By analyzing user behavior on websites, Phishcatch

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

restricts access when suspicious activities are detected, achieving an impressive accuracy and detection rate of 90%. The article also details the processes involved in developing, testing, and implementing effective anti-phishing algorithms.

[26] conducted an analysis of machine learning techniques for identifying phishing URLs using various algorithms, including Extreme Gradient Boosting, Decision Tree, Logistic Regression, Random Forest (RF), and Support Vector Machine. The research aimed to determine the most effective algorithm by comparing their accuracy rates across two datasets, PhishTank and UCI. Model performance was assessed through K-fold validation, feature selection, and hyperparameter tuning, resulting in RF achieving accuracies of 98.80% on the PhishTank dataset and 97.87% on the UCI dataset. The highest precision, recall, and F1-score recorded were all 99%, with an AUC-ROC value of 99.89% for the PhishTank dataset. Validation against other studies indicated that this methodology could effectively assist in identifying phishing websites.

[2] developed a framework that enhances random forest techniques by addressing its limitations in capturing high feature correlation and their dependencies on labels. By integrating random forest with k-means clustering, the framework effectively assesses feature relationships. Evaluated on a dataset of 5,000 samples for phishing detection, the proposed framework significantly outperformed both the random forest classifier and other conventional classification algorithms. It achieved an impressive accuracy of 98.64%, along with precision, recall, and F-measure values of 0.986, 0.987, and 0.986, respectively.

[27] introduced a model for detecting phishing attacks that focuses on the text content of suspicious web pages rather than URL addresses, utilizing Natural Language Processing (NLP) and deep learning algorithms. The model employs the Keras Embedding Layer with Global Vectors for Word Representation (GloVe) to leverage the semantic and syntactic features of web page content. An initial analysis was conducted using NLP and word embedding techniques, with the resulting data then fed into various deep learning algorithms. Four algorithms were evaluated: Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), Gated Recurrent Unit (GRU), and Bidirectional GRU (BiGRU). The findings indicated that the model is effective, achieving a minimum mean accuracy of 96.7%, with BiGRU showing the highest performance at 97.39%.

[28] utilized machine learning algorithms and natural language processing (NLP) to analyze various features of emails, websites, and digital interactions. This initiative employs AI-based techniques to enhance existing cybersecurity measures by providing a more dependable and efficient method for detecting and preventing phishing attacks, thereby safeguarding individuals and organizations from the detrimental effects of cyber threats.

[29] developed a phishing detection system using a hybrid machine learning approach based on URL analysis. They employed various machine learning models, including Decision Trees, Random Forests, and Support Vector Machines, to protect users from fraudulent URLs. The researchers introduced a novel LSD model, combining Logistic Regression, Support Vector Machines, and Decision Trees with both soft and hard voting mechanisms. To optimize the LSD model, they utilized techniques such as canopy feature selection, cross-fold validation, and grid search hyperparameter optimization. The study evaluated model performance using multiple metrics, with comparative analysis demonstrating that their proposed approach outperformed other methods in phishing detection.

Despite significant advancements in phishing detection systems leveraging machine learning techniques like Support Vector Machines (SVM), Feed Forward Neural Networks (FFNN), and Extreme Learning Machines (ELM), certain challenges and gaps remain. Current research often focuses on individual models, but the comparative performance of these algorithms when used in combination or under different conditions, such as handling dynamic phishing patterns or large-scale datasets, is underexplored. Furthermore, while many studies emphasize accuracy, there is limited investigation into optimizing the trade-off between detection speed, computational cost, and false positive rates in real-time applications.

III. Methodology

Research Approach

The work flow of this research is presented in Figure 3.1 and it is made up of the following steps:

Collect a comprehensive dataset of URLs, or web pages, including both legitimate and phishing samples, from public datasets.

Select relevant features by deleting irrelevant features from the URLs dataset, such as domain information, URL length, presence of HTTPS, number of subdomains, content-based features, HTML attributes, and WHOIS information. Remove duplicates, handle missing values, and normalize features to ensure consistency in the dataset. Convert categorical features (e.g., domain registrant country) into numerical values using techniques like one-hot encoding.

Apply Principal Component Analysis (PCA) to reduce the feature set and eliminate redundant or irrelevant features.

Implement an SVM model with a Radial Basis Function (RBF) kernel, FFNN model with multiple hidden layers and ELM model which is known for its fast-training process to detected phishing and legitimate samples based on the extracted features. Train each of the individual models (SVM, FFNN, and ELM) on the training dataset.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

Combine the predictions of the three individual models (SVM, FFNN, ELM) using a Model Weighting Voting Mechanism. The Model Weighting will assign weights to each model's prediction based on its performance (accuracy, precision, sensitivity) during validation. Models with higher accuracy may be given higher weights in the voting process.

Evaluate the final voting-based phishing detection system on the testing dataset using various performance metrics such as, False Positive Rate, Specificity, Sensitivity, Precision, Accuracy and Detection Time.

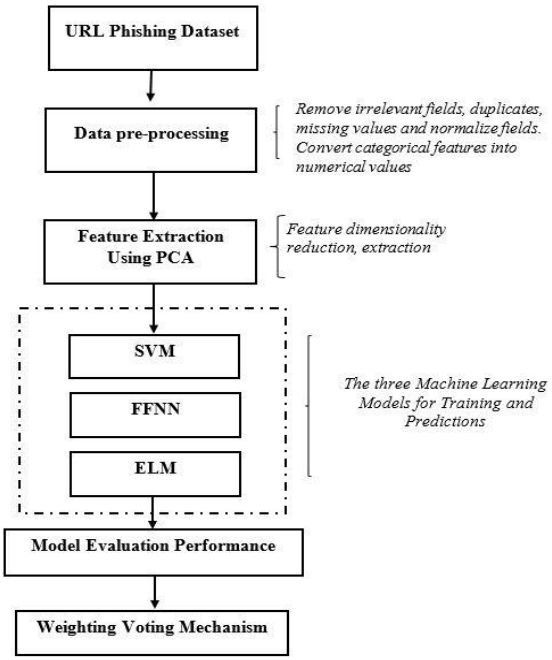


Figure 3.1: The Structure of the Phishing Detection System

Data Acquisition

The dataset will be sourced from www.kaggle.com and consists of over 2115 URLs, offering a comprehensive and diverse range of online domains. Approximately 52% of the domains are legitimate, representing trusted entities, while 47% are classified as phishing domains, indicating potential online threats. The dataset is organized into two main columns: "url," which contains the web addresses, and "status," where 0 denotes phishing domains and 1 indicates legitimate ones. The balanced distribution between phishing and legitimate domains helps prevent class imbalance issues, enhancing the dataset's reliability for analysis. This structure ensures that researchers can effectively study and address online phishing threats with greater accuracy.

Data Preprocessing

Some preprocessing method such as removal of unwanted features and selection of relevant features, removing duplicates, handling missing values and conversion of categorical features into numerical representation are described as follows:

Remove Unwanted Features and Selection of Relevant Features: Features such as URL length, number of subdomains, presence of HTTPS, special characters, IP addresses in the URL, and suspicious keywords (e.g., "login," "verify," "secure") will be selected and other features will be deleted. Web page content will be scraped to gather metadata like the presence of JavaScript, forms, and suspicious links or redirects within the HTML.

Removing Duplicates: Duplicate entries will be identified and removed in the dataset based on the "url" column or other key attributes, ensuring that each URL or domain is unique.

Handling Missing Values: The missing data will be checked in critical columns (e.g., "url," "status," or extracted features) and will impute the missing values by assigning a default value or using the mode of the column.

One-Hot Encoding: This will convert categorical features (such as, domain registrar, country) into numerical representations using one-hot encoding. This process creates binary columns for each category, where the presence of a category is marked by 1, and its absence by 0.

Normalizing Features: Continuous numerical features (such as URL length, domain age, number of subdomains) will be converted to a common scale, typically between 0 and 1, using techniques like Min-Max Scaling.

Feature Extraction

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

Principal Component Analysis (PCA) is a dimensionality reduction technique that transforms a large set of correlated features into a smaller set of uncorrelated components (principal components) while retaining the most important information in the dataset. For a phishing detection system, PCA helps reduce redundant features and enhances computational efficiency without losing significant information. In this study, the preprocessed feature will be introduced to PCA process as input. The feature matrix will be ensured to be purely numerical, as PCA cannot work directly with categorical data. One-hot encoding or label encoding will be used to convert categorical features to numerical ones. PCA will reduces the dimensionality of the dataset by converting it into a set of components. The study will decide how many principal components to retain. This decision is based on how much variance (information) you want to keep, typically 90% to 95% of the total variance [30].

After applying PCA, the explained variance ratio will be examined to ensure that the selected number of principal components retains enough variance (typically 90-95%). This indicates how much information each principal component contributes. The PCA-transformed feature set will be a linear combination of the original features. At the end, these principal components are now uncorrelated and contain the most relevant information in fewer dimensions. PCA will enables the phishing detection system to maintain the most critical information while reducing complexity, improving the efficiency of machine learning models that will be used in this study, and mitigating overfitting risks due to noisy or redundant features. Algorithm 3.1 expressed the PCA [31].

Algorithm 3.1: **Feature Extraction using Principal Component Analysis [31].**

Input: Preprocessed features

Output: Reduced feature set after applying PCA, ready for machine learning model training

Step 1: Load the phishing detection URL dataset, including relevant extracted features such as URL length, subdomains, HTTPS usage, Let the dataset be represented as `X` with `n` samples and `d` features: `X = [$x_1$, $x_2$, ..., $x_d$]`.

Step 2: Clean the dataset by removing missing values and duplicates. If any missing values exist in feature columns, impute them using mean or median for numerical features or a default value for categorical features.

Step 3: Normalize the feature values (scaling the data) using Standardization (Z-score normalization) to ensure that all features have a mean of 0 and a standard deviation of 1. Compute normalized features for each feature `i`:

$$x_i^{norm} = \frac{x_i - \mu_i}{\sigma_i}$$

where μ_i is the mean and σ_i is the standard deviation of feature `i`.

Step 4: Compute the covariance matrix `C` of the normalized feature set X_{norm}The covariance matrix `C` is defined as:

$$C = \frac{1}{n-1} X_{norm}^T X_{norm}$$

Here, `C` is a `d x d` matrix where each element represents the covariance between two features.

Step 5: Calculate the eigenvalues and corresponding eigenvectors of the covariance matrix `C`. Solve the equation:

$$Cv = \lambda v$$

where `v` is the eigenvector and λ is the eigenvalue.

Step 6: Sort the eigenvalues in descending order, and arrange the corresponding eigenvectors to form a matrix of principal components. Larger eigenvalues correspond to principal components that explain more variance in the data.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

Step 7: Determine the number of principal components `k` to retain based on the desired amount of explained variance (e.g., 95% of variance). Calculate the explained variance ratio for each eigenvalue:

$$expalined\ variance\ ratio_i = \frac{\lambda_i}{\sum_{i=1}^{d} \lambda_i}$$

Choose `k` such that the cumulative explained variance is greater than or equal to the desired threshold.

Step 8: Construct the projection matrix `W` from the top `k` eigenvectors (principal components). Form matrix `W = [v1, v2, ..., vk]`, where `vi` is the eigenvector corresponding to the `i`th largest eigenvalue.

Step 9: Project the normalized feature matrix X_{norm} onto the new lower-dimensional

$$X_{PCA} = X_{norm} W$$

where X_{PCA} is the reduced feature set with `n` samples and `k` components.

Step 10: The resulting matrix X_{PCA} is the reduced feature set, which will be used as input to the phishing detection machine learning models (SVM, FFNN, ELM).

Application of SVM, FFNN and ELM for Training and Detection using Voting System

This section described the application of Support Vector Machine (SVM), Feed Forward Neural Network (FFNN) and Extreme Learning Machine (ELM) for detecting phishing. Voting mechanism will be employed to determine the strength of the three algorithms.

Support Vector Machine (SVM)

The application of Support Vector Machine (SVM) in phishing detection involves a multi-step process that leverages the algorithm's ability to find optimal hyperplanes for classification in high-dimensional feature spaces. Once the dataset is pre-processed and the features are extracted, the next step is to train the SVM model. An appropriate SVM kernel based on the nature of the dataset will be choosen. Common kernels include: Linear Kernel which will be useful when the data is linearly separable, Radial Basis Function (RBF) Kernel which is often used when the data is not linearly separable, as phishing data might show non-linear relationships between features and Polynomial Kernel which will be used for non-linear data with complex interactions [32].

The dataset will be split into training and testing sets, with a ratio of 70/30 using random subsampling cross-validation method. The training set will be used to train the SVM model, and the testing set will be used for evaluating the model's performance. The key SVM parameters such as C (regularization parameter) and gamma (for non-linear kernels like RBF) will be tuned to optimize model performance. The regularization parameter C balances the trade-off between maximizing the margin and minimizing classification errors. Higher values of C may lead to overfitting, so careful tuning is necessary. Cross-validation can be used to select optimal hyperparameters. The SVM model will be trained on the labeled dataset, where each URL's feature vector is used to determine whether the URL is phishing or legitimate. During training, SVM will identifies the optimal hyperplane (or decision boundary) that separates phishing websites from legitimate ones. It will also maximize the margin between these two classes, using support vectors (key data points closest to the decision boundary) [33].

Given a training set of feature extracted data $T = \{(x_1 y_1), (x_2 y_2), \ldots, (x_i y_i)\}$ by PCA, $x \in \Re^n$is the sample feature extracted data while $y \in \{0,\ 1\}$ is the corresponding tag of the label. The extracted feature will be introduced into the Support Vector Machine (SVM) for classification.

The decision function of SVM will be expressed as

$$f(x) = sign \sum_{i=1}^{l} s\alpha_i y_i Kcso(\boldsymbol{x}_i, \boldsymbol{x}) + b \qquad (3.1)$$

the function $Kcso(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is a kernel function defined as

$$Kcso(\boldsymbol{x}_i, \boldsymbol{x}_j) \equiv \Phi^T(\boldsymbol{x}_i)\ \Phi(\boldsymbol{x}_j) \qquad (3.2)$$

Where $s\alpha_i \geq 0$, are Lagrange multipliers, $i = 1, 2, \ldots, l$, b is an offset scalar, sign of $f(x)$ gives the membership class of x and y_i in the input selected features. Equation 3.4, which was employed in this work, provides the kernel for the radial basis function (RBF):

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

$$Kpoa(\boldsymbol{x}_i\,,\,\boldsymbol{x}_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right) \quad (3.3)$$

where σ is a constant that defines the kernel width. σ is likewise the penalty or regularization parameter, and it controls the trade-off between the size of the margin and the slack variable penalty. Setting these parameters in an attempt to find a better "approximating function" called selection procedure. The selected parameters are fed into the kernel and SVM will be applied to selected feature data sets.

Feed Forward Neural Network (FFNN)

The input layer accepts the feature vectors extracted from the URLs. Each feature is passed as an input neuron, so if the dataset has n features, the input layer will have n neurons. Typically, an FFNN consists of multiple hidden layers. Each hidden layer will contain a varying number of neurons, and this configuration will be adjusted during model tuning. Apply activation functions to introduce non-linearity into the model. The commonly used activation function is ReLU (Rectified Linear Unit), defined as [37]:

$$f(x) = \max(0, x) \quad (3.4)$$

ReLU is chosen because it helps mitigate the vanishing gradient problem during backpropagation and accelerates convergence.

Since the task is binary classification (phishing or legitimate), the output layer consists of a single neuron with a sigmoid activation function:

$$f(x) = \frac{1}{1+e^{-x}} \quad (3.5)$$

The output of this layer represents the probability that a given URL is phishing (values range from 0 to 1). A threshold of 0.5 is used to detected the output: if the output probability is greater than or equal to 0.5, the URL is classified as phishing; otherwise, it is classified as legitimate.

For binary classification, use binary cross-entropy as the loss function to measure the difference between predicted and actual labels:

$$Loss = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p_i) + (1 + y_i)\log(1 - p_i)] \quad (3.6)$$

where N is the total number of samples, y_i is the actual label (0 or 1), and p_i is the predicted probability for phishing.

Use stochastic gradient descent (SGD) or its variants like Adam optimizer to minimize the loss function and update the weights of the neural network during backpropagation:

Adam is typically preferred for its adaptive learning rate:

$$\theta = \theta - \eta \frac{m_t}{\sqrt{v_t}+\epsilon} \quad (3.7)$$

where θ represents the weights, η is the learning rate, and m_t and v_t are the first and second moments (mean and variance) of the gradients, respectively. Algorithm 3.2 expressed the application of FFNN for Phishing Detection [38] [39].

Algorithm 3.2: Phishing Detection Using Feed Forward Neural Network (FFNN)

Step 1: Initialize the Feed Forward Neural Network**

Input Layer: The input layer has `m` neurons, where `m` is the number of input features extracted from the URLs.

Hidden Layers: Use one or more hidden layers. The number of neurons in these layers is chosen based on model complexity and dataset size. Apply an activation function such as ReLU (Rectified Linear Unit) in each hidden layer:

$$\boldsymbol{ReLU(z) = \max(0, z)}\,;$$

where `z = w \cdot x + b`, with `w` being the weights, `x` the input, and `b` the bias.

Output Layer: The output layer consists of a single neuron with a **sigmoid activation function since phishing detection is a binary classification problem:**

$$\boldsymbol{\sigma(z) = \frac{1}{1 + e^{-z}}}$$

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

The output represents the probability that a URL is phishing. If the output is ≥ 0.5, detected the URL as phishing (1); otherwise, detected it as legitimate (0).

Step 2: Forward Propagation: For each training sample, perform the following steps:

Calculate Input to Hidden Layer:

$$\boldsymbol{z^{(1)} = W^{(1)}.X + b^{(1)}}$$

where $W^{(1)}$ is the weight matrix between the input layer and the first hidden layer, $b^{(1)}$ is the bias vector for the hidden layer.

Apply Activation Function to Hidden Layer:

$$\boldsymbol{A^{(1)} = ReLU(z^{(1)})}$$

where $A^{(1)}$ is the output of the hidden layer.

Calculate Input to Output Layer:

$$\boldsymbol{z^{(2)} = W^{(2)}.A^{(1)} + b^{(2)}}$$

Where $W^{(2)}$ is the weight matrix between the hidden layer and output layer, $b^{(2)}$ is the bias vector for the output layer.

Apply Activation Function to Output Layer:

$$\boldsymbol{\hat{y} = \sigma\left(z^{(2)}\right) = \frac{1}{1 + e^{-z^{(2)}}}}$$

where $\hat{y}$ is the predicted probability that the URL is phishing.

Step 3: Loss Function: Use Binary Cross-Entropy Loss to measure the error between predicted and actual labels:

$$\boldsymbol{Loss = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 + y_i)\log(1 - \hat{y}_i)]}$$

where y_i is the true label, and $\hat{y}_i$ is the predicted probability for each sample.

Step 4: Backpropagation and Weight Update:

Compute Gradients: For each layer, calculate the gradients of the loss function with respect to weights and biases using the chain rule of calculus.

Weight and Bias Update: Update the weights `W` and biases `b` using gradient descent:

$$\boldsymbol{W \leftarrow W - \eta.\frac{\partial Loss}{\partial W}}$$

$$\boldsymbol{b \leftarrow b - \eta.\frac{\partial Loss}{\partial b}}$$

where η is the learning rate.

Step 6: Model Training: Train the FFNN by iterating through the training data for multiple epochs (iterations). In each epoch, perform forward propagation, calculate loss, backpropagate the gradients, and update the weights and biases.

Extreme Learning Machine (ELM)

Extreme Learning Machine (ELM) is a fast and efficient learning algorithm for single-layer feedforward neural networks (SLFNs). Its key advantage lies in its speed and ability to generalize well with a random initialization of hidden node parameters, eliminating the need for iterative tuning like in traditional neural networks. The input weights `$W_{in}$` and biases `b` for the hidden layer are randomly initialized and fixed throughout training as defined in Algorithm 3.3. Algorithm 3.3 expressed the Phishing Detection Using Extreme Learning Machine (ELM) [34][5].

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

Weighted Voting Mechanism

The goal of this study is to enhance phishing detection accuracy by leveraging a model weighting voting mechanism that combines the strengths of three different algorithms: Support Vector Machine (SVM), Feed Forward Neural Network (FFNN), and Extreme Learning Machine (ELM). This approach integrates diverse models to capitalize on their unique advantages and improve overall classification performance. The following steps explain how to apply Model Weighting and Voting Mechanism

Calculate performance metrics such as accuracy as discussed in section 3.7 for each model using the validation set or cross-validation.

Based on the performance metrics, assign weights to each model. In this study, higher weights will be assigned to models with better performance.

Obtain Predictions: Collect predictions from each model. SVM Predictions P_{SVM}, FFNN Predictions P_{FFNN}, ELM Predictions P_{ELM}. Calculate Weighted Sum as

$$Weighted_{sum} = Weight_{SVM} \times P_{SVM} + Weight_{FFNN} \times P_{FFNN} + Weight_{ELM} \times P_{ELM} \quad (3.8)$$

Apply Voting Mechanism: For each test sample, determine the final prediction by applying a threshold to the weighted sum.

$$Final\ Prediction = \begin{cases} 1 & if\ Weighted_{sum} \geq Threshold \\ 0 & if\ Weighted_{sum} < Threshold \end{cases} \quad (3.9)$$

Algorithm 3.3: Phishing Detection using ELM
Step 1: Initialize ELM Parameters: Number of Hidden Neurons (`L`), Choose the number of hidden neurons `L`. This hyperparameter can be tuned experimentally. Random Initialization of Input Weights and Biases: **- Initialize input weight matrix and bias vector b randomly.** **- These parameters remain fixed during the entire training process.** **Step 2: Forward Propagation (Hidden Layer Output Calculation)** **Calculate Hidden Layer Input:** **- For each training sample, compute the input to the hidden layer:** $\boldsymbol{z = W_{in} \cdot X + b}$ **Where X is the input feature vector, W_{in} is the randomly initialized input weight matrix, b is the bias vector.** **Apply Activation Function:** **- Apply the activation function `g(.)` to the hidden layer input. The commonly used activation functions are sigmoid or ReLU:** **- Sigmoid activation function:** $\boldsymbol{g(z) = \frac{1}{1+e^{-z}}}$ **- Compute the hidden layer output matrix `H`:** $\boldsymbol{H = g(W_{in} \cdot X + b)}$ **Step 3: Calculate Output Weights** **Output Weights Calculation:** **- Compute the output weight matrix using the Moore-Penrose pseudoinverse of `H`:** $\boldsymbol{\beta = H^{\uparrow} \cdot Y}$ **where $H^{\uparrow}$ is theMoore-Penrose pseudoinverse of the hidden layer output matrix `H`. Y is the target output (phishing or legitimate labels). This step replaces traditional backpropagation in neural networks, allowing the output weights to be calculated in a single step.**

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

Step 4: Testing Phase

Forward Propagation for Testing:

- For each test sample, compute the hidden layer output H_{test} using the same weights W_{in} and biases `b` initialized during the training phase:

$$H_{test} = g(W_{in} \cdot X_{best} + b)$$

Predict Output:

- Compute the predicted labels `Y_{pred}` for the test set:

$$Y_{pred} = H_{test} \cdot \beta$$

- Apply a threshold to detected URLs as phishing (`1`) or legitimate (`0`):

$$\hat{y} = \begin{cases} 1 & if\ Y_{pred} \geq 0.5 \\ 0 & if\ Y_{pred} < 0.5 \end{cases}$$

Implementation of SVM, FFNN and ELM

The application of SVM, FFNN and ELM will be implemented using MATLAB (R2023a) software on Windows 10 Enterprise 64-bit operating system, Intel®Core ™ i7-250M CPU @2.60GHZ Central Processing Unit, 4GB RAM and 500 Gigabytes hard disk drive. An interactive Graphic User Interface (GUI) was developed with URL phishing datasets.

Evaluation Metrics

The performance of SVM, FFNN and ELM will be evaluated using metrics such as false positive rate, sensitivity, specificity, precision and recognition accuracy. Confusion matrix will be used to determine the value of the performance metrics. It contains True Positive (TP), False Positive (FP), False Negative (FN) and True Negative (TN).

TP (True positive): This denote correctly detected as phishing. FP (False positive): This denote incorrectly detected as phishing. TN (True negative): This denote correctly detected as legitimate. FN (False negative): This denote incorrectly detected as legitimate.

$$False\ Positive\ Rate = \frac{FP}{FP+TN} * 100 \quad (3.10)$$

Sensitivity is defined as the ability of a classification algorithm to detect phishing. It also denotes ability to identify presence of phishing.

$$Sensitivity = \frac{TP}{TP+FN} * 100 \quad (3.11)$$

Specificity is defined as the ability of a classification algorithm to detect legitimate. It also denotes ability to identify absence of phishing.

$$Specificity = \frac{TN}{TN+FP} * 100 \quad (3.12)$$

Precision is defined as the ability of a classification algorithm to detect phishing.

$$Precision = \frac{TP}{TP+FP} * 100 \quad (3.13)$$

Recognition accuracy is defined as the ability of a classification algorithm to detect all phishing. It also denotes ability to identify the overall presence of phishing.

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} * 100 \quad (3.14)$$

Expected Contribution to Knowledge

In a Phishing Detection System, the integration of Support Vector Machine (SVM), Feed Forward Neural Network (FFNN), and Extreme Learning Machine (ELM) using a voting method will significantly enhance detection accuracy. SVM is known for its ability to handle high-dimensional data and separate classes with optimal margins, which will make it effective in identifying phishing patterns. FFNN will contributes by learning complex, non-linear relationships in data, which will further improve the system's predictive power. ELM, with its fast-learning speed and generalization capabilities, will adds efficiency and robustness to the detection process. By employing a voting method that combines these three algorithms, the system will leverage their individual strengths to provide a more reliable and comprehensive phishing detection mechanism.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

VI. Results and Discussion

Results

The phishing detection system was tested using three machine learning classifiers: Support Vector Machine (SVM), Feed Forward Neural Network (FFNN), and Extreme Learning Machine (ELM) on a dataset consisting of 2115 URLs. This dataset included 1115 phishing URLs and 1000 legitimate URLs, which were divided into training and testing sets using three different ratios: 60%-40%, 70%-30%, and 80%-20%. The performance of each classifier was evaluated based on accuracy, precision, recall, and F1-score. The random sub-sampling cross-validation method was employed to ensure a fair evaluation by averaging results over multiple iterations. Each classifier demonstrated different strengths and weaknesses across the dataset splits.

The training process for these classifiers is illustrated in Figure 4.1. During training, each model learned the distinguishing patterns between phishing and legitimate URLs based on extracted features such as domain length, presence of special characters, and HTTPS usage. The SVM classifier aimed to find an optimal hyperplane that best separates phishing from legitimate URLs. FFNN, a multi-layer perceptron, used backpropagation to adjust weights and minimize classification error over several epochs. ELM, on the other hand, leveraged its single hidden-layer structure to rapidly assign random weights and analytically determine output weights, making it computationally efficient.

The testing/detection process, as demonstrated in Figure 4.2, involved classifying new URLs as phishing or legitimate based on learned patterns. When a URL was entered, the system extracted relevant features and fed them into the trained models for classification.

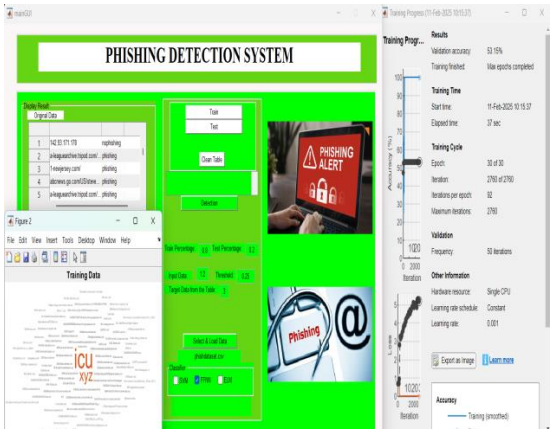


Figure 4.1: Graphical User Interface describing the Training Process of the Techniques

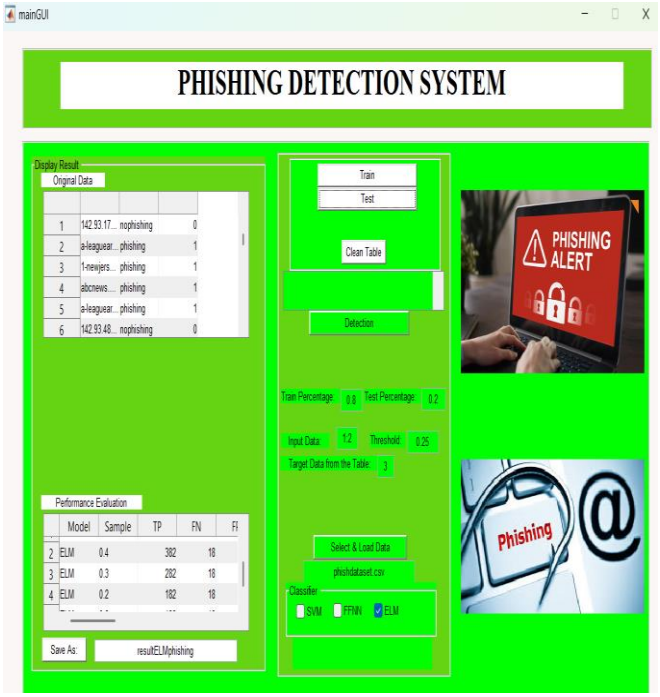


Figure 4.2: Graphical User Interface describing the Testing Process of the Techniques

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

The output was displayed in tabular form, with phishing URLs labeled as '1' and legitimate URLs as '0'. Performance evaluation metrics such as True Positives (TP), False Negatives (FN), and False Positives (FP) were recorded to assess detection efficiency. The ELM classifier exhibited the fastest classification time due to its analytical solution approach, while FFNN required multiple iterations to achieve stable predictions.

The experimental results revealed that FFNN achieved the highest detection accuracy, particularly with the 80%-20% split, indicating its effectiveness in learning complex patterns. SVM performed well with a 70%-30% split but struggled slightly in cases with ambiguous URLs. ELM, while computationally efficient, showed slightly lower accuracy compared to FFNN due to its random weight initialization. However, ELM excelled in real-time detection scenarios, making it a suitable choice for rapid classification applications. These findings highlight the trade-offs between accuracy and computational efficiency among the three classifiers, emphasizing the importance of selecting the appropriate model based on application needs.

Result of SVM

The results of the Support Vector Machine (SVM) for phishing detection in Table 4.1, based on the confusion matrix, demonstrate high accuracy and reliability across different data divisions. When using a 60-40 train-test split, the True Positive (TP) and True Negative (TN) values were 389 and 438, respectively, while False Positive (FP) and False Negative (FN) values remained low at 8 and 11. This resulted in a high specificity (SPEC) of 98.21% and sensitivity (SEN) of 97.25%, indicating the model's strong ability to correctly identify both phishing and legitimate instances. Additionally, the accuracy (ACC) was 97.75%, showing the effectiveness of the model in distinguishing phishing threats from non-threats. The precision (PREC) of 97.98% suggests that the model produces minimal false alarms, making it highly reliable for real-world applications.

Table 4.1: Result of Evaluation Performance with SVM

Data Division	0.6 x 0.4	0.7 x 0.3	0.8 x 0.2
TP	389	290	192
FN	11	10	8
FP	8	10	16
TN	438	324	207
FPR (%)	1.79	2.99	7.17
SPEC (%)	98.21	97.01	92.83
SEN (%)	97.25	96.67	96.00
PREC (%)	97.98	96.67	92.31
ACC (%)	97.75	96.85	94.33
Legitimate (Weighted)	0.46	0.46	0.45
Phishing (Weighted)	0.52	0.51	0.50
Time(sec)	42.35	39.89	43.27

With a 70-30 data split, the SVM model maintained strong performance, though there was a slight decline in specificity and accuracy. The True Positives were 290, while False Negatives slightly reduced to 10, indicating a minor improvement in detecting phishing cases. However, False Positives increased to 10, leading to a False Positive Rate (FPR) of 2.99% and a specificity of 97.01%. Despite this, the sensitivity remained high at 96.67%, and accuracy was still commendable at 96.85%. Precision was also 96.67%, meaning the model continued to effectively differentiate between phishing and legitimate sites, although with a slight increase in misclassification.

When using an 80-20 data split, the SVM model showed a more noticeable drop in specificity and precision. While True Positives were 192 and False Negatives decreased to 8, the False Positives increased to 16, leading to an FPR of 7.17% and a lower specificity of 92.83%. This decline suggests that the model had more difficulty distinguishing between phishing and legitimate sites as the training data size decreased. The accuracy also dropped to 94.33%, with a slight reduction in sensitivity to 96.00% and precision to 92.31%, indicating a higher risk of false phishing detections. Additionally, the processing time varied across data splits, with the longest time recorded at 43.27 seconds for the 80-20 split. These results suggest that while SVM remains highly effective for phishing detection, an optimal data division, such as 60-40 or 70-30, is preferable to balance accuracy, specificity, and computational efficiency.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

Result of ELM

The results of the Extreme Learning Machine (ELM) for phishing detection in Table 4.2, based on the confusion matrix, indicate a slightly lower performance compared to the Support Vector Machine (SVM). With a 60-40 data split, the True Positives (TP) and True Negatives (TN) were 379 and 428, respectively, while False Positives (FP) and False Negatives (FN) increased to 18 and 21. This resulted in a False Positive Rate (FPR) of 4.04% and a specificity (SPEC) of 95.96%, which is lower than that of the SVM model. The sensitivity (SEN) was 94.75%, showing a slight decrease in the model's ability to detect phishing attempts correctly. Although the accuracy (ACC) was still high at 95.39%, the model's precision (PREC) of 95.47% suggests that it may generate more false alarms compared to SVM, making it slightly less reliable for real-world deployment.

When using a 70-30 data split, the ELM model's performance slightly declined, with True Positives dropping to 280 and True Negatives to 314. The False Positives and False Negatives remained at 20 each, increasing the False Positive Rate to 5.99% and reducing specificity to 94.01%. The sensitivity also decreased to 93.33%, meaning the model was slightly less effective in identifying phishing websites correctly. Accuracy was recorded at 93.69%, showing a downward trend as the training data size decreased. Additionally, the precision also dropped to 93.33%, suggesting that while ELM remains effective, it struggles more with distinguishing between legitimate and phishing sites as the dataset shrinks.

With an 80-20 data split, the ELM model exhibited the most significant decline in performance. True Positives were 182, while False Negatives slightly increased to 18, indicating that some phishing sites went undetected. The False Positives increased to 26, leading to a False Positive Rate of 11.66% and a specificity of 88.34%, which is considerably lower than the previous splits. The overall accuracy dropped to 89.60%, with sensitivity at 91.00% and precision at 87.50%, showing a noticeable reduction in detection capability. Although the processing time remained relatively stable around 29.33 seconds, the higher rate of misclassification suggests that ELM is less robust than SVM in handling phishing detection, particularly when trained on smaller datasets.

Table 4.2: Result of Evaluation Performance with ELM

Data Division	0.6 x 0.4	0.7 x 0.3	0.8 x 0.2
TP	379	280	182
FN	21	20	18
FP	18	20	26
TN	428	314	197
FPR (%)	4.04	5.99	11.66
SPEC (%)	95.96	94.01	88.34
SEN (%)	94.75	93.33	91.00
PREC (%)	95.47	93.33	87.50
ACC (%)	95.39	93.69	89.60
Legitimate (Weighted)	0.45	0.44	0.42
Phishing (Weighted)	0.50	0.49	0.47
Time(sec)	28.21	29.17	29.33

Result of FFNN

The results of the Feed Forward Neural Network (FFNN) on the phishing detection system, as shown in Table 4.3, demonstrate its strong performance across three data divisions: 0.6 x 0.4, 0.7 x 0.3, and 0.8 x 0.2. The confusion matrices reveal that the model achieves high True Positives (TP) and True Negatives (TN) values, indicating its ability to correctly classify both phishing and legitimate instances. For instance, in the 0.6 x 0.4 division, the model correctly identifies 384 phishing instances (TP) and 433 legitimate instances (TN), with only 16 false negatives (FN) and 13 false positives (FP). This results in a high accuracy (ACC) of 96.57%, showcasing the model's effectiveness in distinguishing between the two classes. However, as the data division becomes more challenging, such as in the 0.8 x 0.2 division, the model's performance slightly declines, with TP dropping to 187 and TN to 202, while FP and FN increase to 21 and 13, respectively.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

The False Positive Rate (FPR) and Specificity (SPEC) metrics provide further insight into the model's ability to minimize misclassifications of legitimate instances. In the 0.6 x 0.4 division, the FPR is 2.91%, meaning only 2.91% of legitimate instances are incorrectly flagged as phishing, while the specificity is 97.09%, indicating a high rate of correctly identified legitimate instances. However, as the data division shifts to 0.8 x 0.2, the FPR increases to 9.42%, and specificity drops to 90.58%, reflecting a decline in the model's ability to correctly classify legitimate instances in more imbalanced datasets. Despite this, the model maintains a relatively high precision (PREC) of 89.90% in the 0.8 x 0.2 division, indicating that most instances predicted as phishing are indeed correct. This suggests that while the model remains effective, its performance is influenced by the distribution of the dataset, with more imbalanced divisions posing greater challenges.

Table 4.3: Result of Evaluation Performance with FFNN

Data Division	0.6 x 0.4	0.7 x 0.3	0.8 x 0.2
TP	384	285	187
FN	16	15	13
FP	13	15	21
TN	433	319	202
FPR(%)	2.91	4.49	9.42
SPEC(%)	97.09	95.51	90.58
SEN(%)	96.00	95.00	93.50
PREC(%)	96.73	95.00	89.90
ACC(%)	96.57	95.27	91.96
Legitimate (Weighted)	0.46	0.45	0.43
Phishing (Weighted)	0.51	0.50	0.48
Time(sec)	60.03	56.23	59.47

The Sensitivity (SEN) and Precision (PREC) metrics further highlight the model's effectiveness in detecting phishing instances. Across all divisions, sensitivity remains above 93%, with the highest value of 96.00% in the 0.6 x 0.4 division, indicating that the model successfully identifies most phishing instances. Precision also remains strong, with values above 89% in all divisions, demonstrating that the model's predictions are highly reliable. Additionally, the weighted scores for legitimate and phishing instances show consistent performance, with phishing-weighted scores slightly higher, reflecting the model's focus on accurately detecting phishing activities. The computational efficiency of the FFNN is also notable, with processing times ranging from 56.23 to 60.03 seconds, making it a practical solution for real-time phishing detection. Overall, the FFNN demonstrates strong performance across all metrics, though its effectiveness slightly diminishes with more imbalanced data divisions.

Combined Result based on Voting Mechanism

The results of the weighted majority voting approach, combining the probabilities from SVM, ELM, and FFNN models, show that the final phishing detection is consistently stronger than legitimate classification across all data splits. For the 60-40 data division, the summed probabilities for the phishing class reached 1.52, while the legitimate class was at 1.37. This indicates that phishing websites had a higher overall confidence level across the models, leading to a final classification as phishing. The slight variation in weighted probabilities across models suggests that while each contributes differently to the classification, phishing remains the dominant prediction. This confirms that the combined model effectively strengthens phishing detection by leveraging multiple classifiers as described.

For the 70-30 data split, the phishing class again had a higher summed probability of 1.50, compared to 1.34 for legitimate websites. The slight decrease in summed probability for phishing compared to the 60-40 split indicates a minor reduction in confidence, but phishing still remains the final detection. The decline in legitimate classification probability suggests that as the dataset size for training increases, the models collectively become more confident in distinguishing phishing attempts. Each model individually contributes to the final decision, and the agreement among them ensures that the classification is more robust. This reinforces the effectiveness of weighted majority voting in phishing detection, particularly in reducing misclassifications as described in Table 4.4.

With the 80-20 data division, the phishing class maintains dominance with a summed probability of 1.47, while the legitimate class drops to 1.32. This reduction in overall confidence for both classes suggests that with less training data, the models experience a slight decline in performance. However, phishing detection still emerges as the final classification, proving that even

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

with smaller training sets, the combined approach remains reliable. The declining trend in summed probabilities highlights that model confidence decreases with smaller training data, but phishing detection is still consistently prioritized. This result supports the conclusion that weighted majority voting effectively enhances phishing detection by consolidating the strengths of multiple models.

In Table 4.4, the combined results of weighted probabilities across all models (SVM, ELM, and FFNN) using weighted majority voting demonstrate a robust approach to phishing detection. By summing the weighted probabilities for legitimate and phishing classes across the three models, the final detection is determined by the class with the highest summed probability. For instance, in the 0.6 x 0.4 data division, the summed probabilities for phishing are 1.52, 1.50, and 1.47 for SVM, FFNN, and ELM, respectively, while the summed probabilities for legitimate instances are 1.37, 1.34, and 1.32. This indicates that phishing is consistently identified as the final detection class across all data divisions, as the summed phishing probabilities are higher than those for legitimate instances. This approach leverages the strengths of each model, ensuring a more reliable and accurate detection outcome.

Table 4.4: Result of voting mechanism among SVM, FFNN and ELM

Model / Data Division	SVM		FFNN		ELM	
	Legitimate (Weighted)	**Phishing (Weighted)**	**Legitimate (Weighted)**	**Phishing (Weighted)**	**Legitimate (Weighted)**	**Phishing (Weighted)**
0.6 x 0.4	0.46	0.52	0.46	0.51	0.45	0.50
0.7 x 0.3	0.46	0.51	0.45	0.50	0.44	0.49
0.8 x 0.2	0.45	0.50	0.43	0.48	0.42	0.47
Summed Weight Probabilities	1.``37	1.52	1.34	1.50	1.32	1.47
Final Detection	Phishing (1.52)		Phishing (1.50)		Phishing (1.47)	

The efficiency of the models can be evaluated based on their individual contributions to the summed probabilities. SVM consistently provides the highest weighted probabilities for phishing across all data divisions, with values of 0.52, 0.51, and 0.50 for 0.6 x 0.4, 0.7 x 0.3, and 0.8 x 0.2, respectively. This suggests that SVM is the most effective model in distinguishing phishing instances, as it contributes the most to the final phishing detection. FFNN follows closely, with weighted probabilities of 0.51, 0.50, and 0.48, while ELM has slightly lower values of 0.50, 0.49, and 0.47. Despite these differences, all three models contribute significantly to the final decision, with phishing consistently emerging as the dominant class due to the higher summed probabilities.

In terms of efficiency, SVM stands out as the most effective model based on the results in Table 4.4. Its higher weighted probabilities for phishing across all data divisions indicate a stronger ability to correctly identify phishing instances compared to FFNN and ELM. However, FFNN and ELM also perform well, with their contributions ensuring a balanced and reliable final detection outcome. The weighted majority voting approach effectively combines the strengths of all three models, mitigating the weaknesses of individual models and enhancing overall detection accuracy. This collaborative approach ensures that the final detection is robust and reliable, even in more challenging data divisions like 0.8 x 0.2. Overall, the combined results highlight the effectiveness of using weighted majority voting for phishing detection, with SVM emerging as the most efficient model in this ensemble.

V. Discussion of Results

The weighted majority voting results show that phishing detection is consistently dominant across all models and data divisions. In the 60-40 data split, the summed probability for phishing classification was highest at 1.52, while legitimate classification had 1.37. This trend continues in the 70-30 and 80-20 splits, where phishing maintains a lead, albeit with slightly reduced confidence. The combination of SVM, ELM, and FFNN models strengthens phishing detection by aggregating their predictions, ensuring more robust classification. This confirms the efficiency of weighted majority voting in improving phishing detection accuracy, aligning with findings from existing literature that emphasize ensemble learning's ability to enhance classification reliability [40].

Analyzing Figures 4.3 to 4.8, SVM consistently demonstrates superior performance in key metrics such as False Positive Rate (FPR), Specificity, Sensitivity, Precision, and Accuracy. Figure 4.3 shows that SVM maintains the lowest FPR across all data splits, ensuring fewer legitimate sites are misclassified as phishing. Figure 4.4 further highlights that SVM achieves the highest specificity, reinforcing its ability to correctly classify legitimate sites. Similarly, Figures 4.5 and 4.6 demonstrate SVM's high

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

sensitivity and precision, indicating its reliability in detecting phishing attempts while minimizing misclassifications. These findings align with previous research indicating that SVM-based classifiers excel in distinguishing phishing websites due to their strong generalization capabilities [35].

Figure 4.7 confirms that SVM maintains the highest accuracy among the three models, proving its overall effectiveness in phishing detection. Additionally, Figure 4.8 illustrates that SVM processes detection faster than FFNN, although ELM has the shortest recognition time. However, considering both efficiency and accuracy, SVM emerges as the most balanced model for phishing detection. The results support the assertion by [36] that SVM's ability to handle high-dimensional data efficiently makes it a strong candidate for phishing detection. Thus, while weighted majority voting improves phishing classification by integrating multiple models, SVM proves to be the most efficient and reliable classifier among the three.

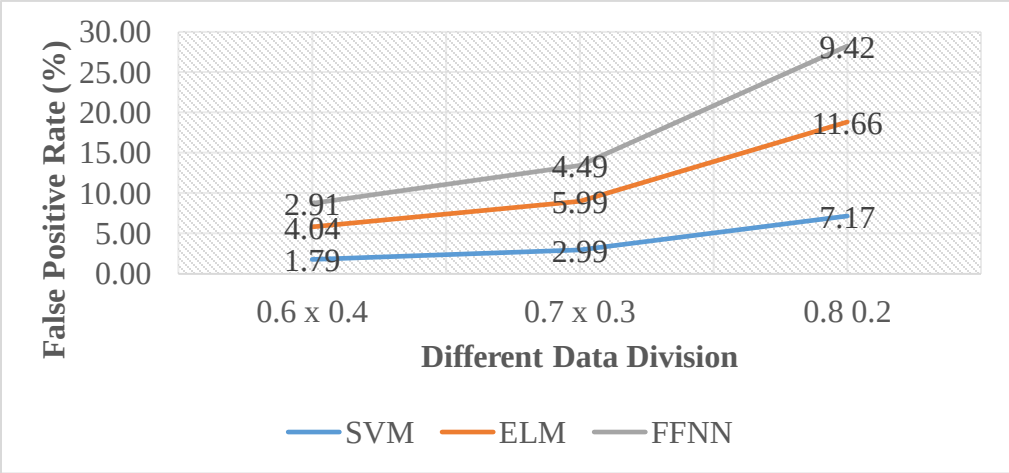


Figure 4.3: Graph of False Positive Rate for all the three Models

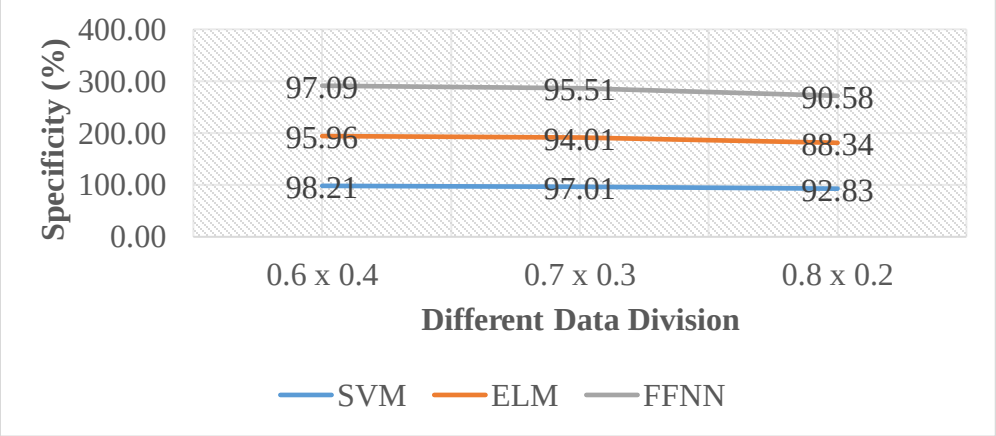


Figure 4.4: Graph of Specificity for all the three Models

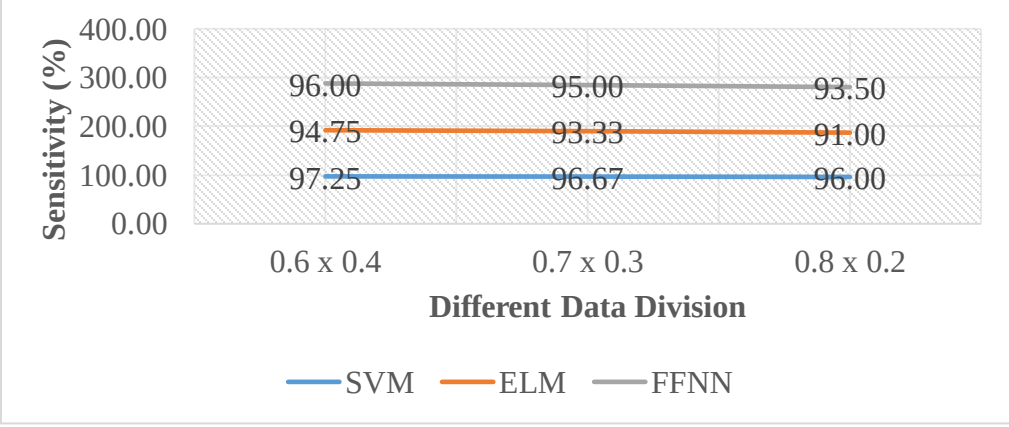


Figure 4.5: Graph of Sensitivity for all the three Models

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

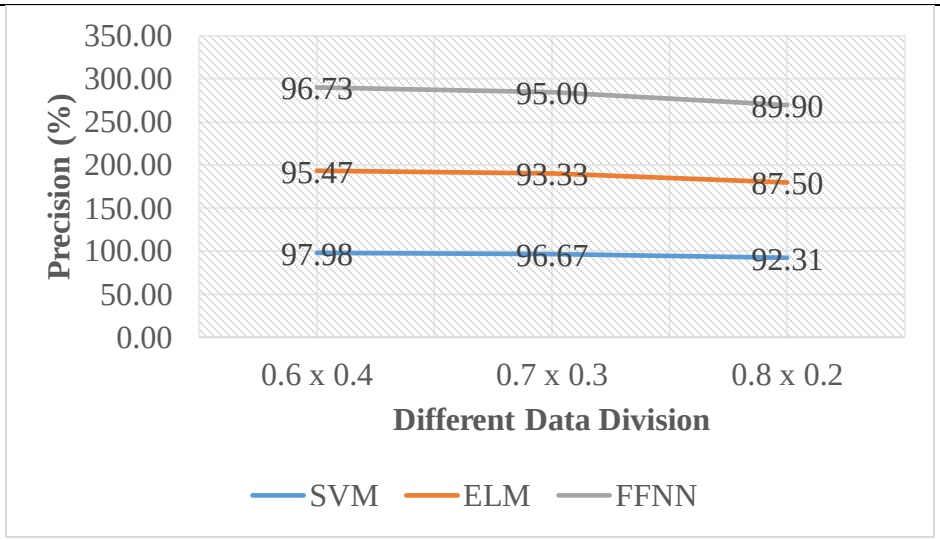


Figure 4.6: Graph of Precision for all the three Models

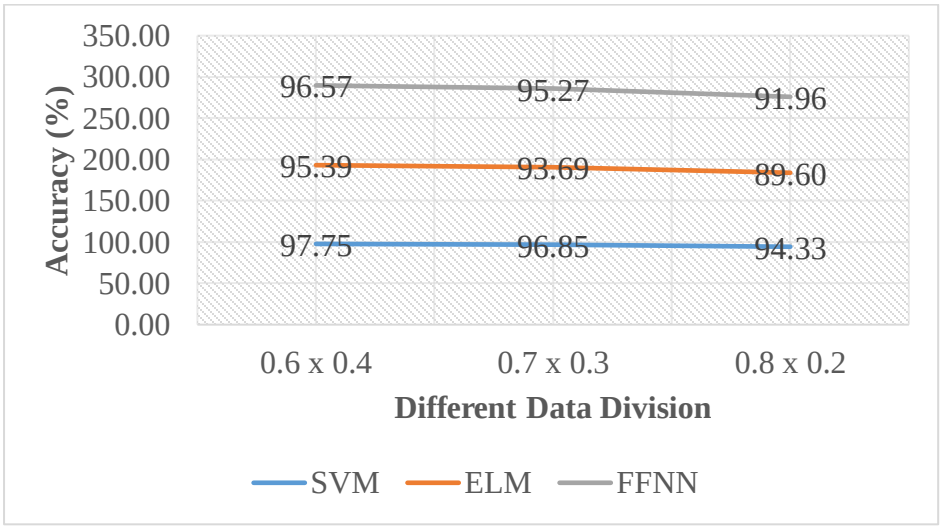


Figure 4.7: Graph of Accuracy for all the three Models

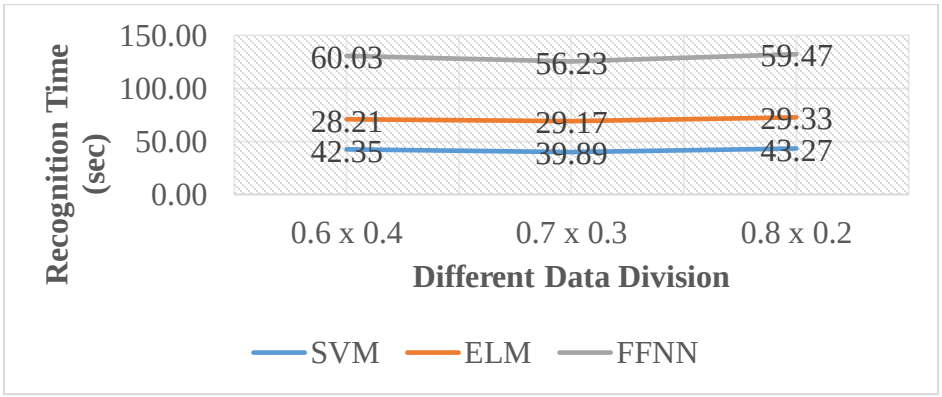


Figure 4.8: Graph of Recognition for all the three Models

VI. Conclusion

The weighted majority voting approach confirms that phishing detection is the dominant classification across all models, as the summed probability for phishing is consistently higher than that for legitimate websites. In all data divisions (60-40, 70-30, and 80-20), phishing maintains the highest probability. This result indicates that combining SVM, ELM, and FFNN improves phishing detection by leveraging their strengths, making the classification process more robust. The findings align with previous research, which suggests that ensemble learning techniques enhance accuracy and resilience in phishing detection. Therefore, the

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

integration of multiple models through weighted majority voting significantly improves phishing classification, reducing the likelihood of misclassifications.

Among the three models, SVM demonstrates superior efficiency across key metrics, including false positive rate (FPR), specificity, sensitivity, precision, accuracy, and detection time. SVM consistently maintains the lowest FPR, indicating fewer false alarms, while its high specificity ensures legitimate websites are correctly classified. Additionally, SVM achieves the highest sensitivity and precision, making it the most reliable model for detecting phishing attempts with minimal misclassification. Although ELM has the shortest detection time, SVM offers the best balance between accuracy and processing efficiency, making it the most effective model overall. These findings reaffirm that SVM is the most suitable classifier for phishing detection, as it consistently outperforms ELM and FFNN in both accuracy and sensitivity.

VII. Recommendations

Based on the results of weighted majority voting across SVM, ELM, and FFNN for phishing detection, it is recommended that SVM be prioritized due to its superior performance in key evaluation metrics such as accuracy, specificity, sensitivity, and false positive rate. However, integrating multiple models through ensemble learning should be encouraged to enhance detection robustness and minimize classification errors. Future research should explore hybrid approaches that combine SVM with deep learning techniques to further improve phishing detection accuracy. Additionally, optimizing feature selection and model tuning can enhance efficiency, ensuring faster and more precise classification of phishing threats.

Acknowledgement

I appreciate TETFUND for her financial support toward the successful completion of this research.

References

1. Adebowale, M. A., Lwin, K. T., & Hossain, M. A. (2023). Intelligent phishing detection scheme using deep learning algorithms. Journal of Enterprise Information Management, 36(3), 747-766
2. Alsharaiah, M., Abu-Shareha, A., Abualhaj, M., Baniata, L., Adwan, O., Al-saaidah, A., &Oraiqat, M. (2023). A new phishing-website detection framework using ensemble classification and clustering. International Journal of Data and Network Science, 7(2), 857-864.
3. Kalla, D., Samaah, F., Kuraku, S., & Smith, N. (2023). Phishing detection implementation using databricks and artificial Intelligence. International Journal of Computer Applications, 185(11), 1-11.
4. Lee, J., Ye, P., Liu, R., Divakaran, D. M., & Chan, M. C. (2020). Building robust phishing detection system: an empirical analysis, pp. 1-12.
5. Mughaid, A., AlZu'bi, S., Hnaif, A., Taamneh, S., Alnajjar, A., &Elsoud, E. A. (2022). An intelligent cyber security phishing detection system using deep learning techniques. Cluster Computing, 25(6), 3819-3828.
6. Shaukat, M. W., Amin, R., Muslam, M. M. A., Alshehri, A. H., & Xie, J. (2023). A hybrid approach for alluring ads phishing attack detection using machine learning. Sensors, 23(19), 8070.
7. Sharma, H., Meenakshi, E., & Bhatia, S. K. (2020). A comparative analysis and awareness of phishing attacks. International Journal of Advanced Science and Technology, 29(3), 11753-11766.
8. Aleroud, A., & Zhou, L. (2017). Phishing environments, techniques, and countermeasures: A survey. Computers & Security, 68, 160-196.
9. Chiew, K. L., Tan, C. L., Wong, K., Yong, K. S., & Tiong, W. K. (2019). A new hybrid ensemble feature selection framework for machine learning-based phishing detection system. Information Sciences, 484, 153-166.
10. Gupta, B. B., Tewari, A., Jain, A. K., & Agrawal, D. P. (2017). Fighting against phishing attacks: state of the art and future challenges. Neural Computing and Applications, 28(12), 3629-3654.
11. Abdelhamid, N., Ayesh, A., &Thabtah, F. (2014). Phishing detection based associative classification data mining. Expert Systems with Applications, 41(13), 5948-5959.
12. Zhu, E., Chen, Y., Ye, C., Li, X., & Liu, F. (2019). OFS-NN: An effective phishing website detection model based on optimal feature selection and neural network. IEEE Access, 7, 73271-73284.
13. Sahingoz, O. K., Buber, E., Demir, O., & Diri, B. (2019). Machine learning based phishing detection from URLs. Expert Systems with Applications, 117, 345-357.
14. Jain, A. K., & Gupta, B. B. (2018). A novel approach to protect against phishing attacks at client side using auto-updated white-list. EURASIP Journal on Information Security, 2018(1), 1-11.
15. Ramesh, G., Krishnamurthi, I., & Kumar, K. S. S. (2020). An efficacious method for detecting phishing webpages through target domain identification. Decision Support Systems, 138, 113360.
16. Abdelnabi, S., Krombholz, K., & Fritz, M. (2020). VisualPhishNet: Zero-day phishing website detection by visual similarity. In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (pp. 1681-1698).
17. Bahnsen, A. C., Bohorquez, E. C., Villegas, S., Vargas, J., & González, F. A. (2017). Classifying phishing URLs using recurrent neural networks. In 2017 APWG Symposium on Electronic Crime Research (eCrime) (pp. 1-8). IEEE.

ISSN 2278-2540 | DOI: 10.51583/IJLTEMAS | Volume XIV, Issue VI, June 2025

18. Venugopal, S., Viswanath, S., & Anitha, R. (2020). A blockchain-based framework for phishing detection and prevention. In Blockchain Technology for Industry 4.0 (pp. 239-257). Springer, Singapore.
19. Azeez, N. A., & Venter, I. M. (2021). Towards ensuring scalability, interoperability and efficient access control in a multi-domain IoT-based healthcare system. Journal of Ambient Intelligence and Humanized Computing, 12(6), 6213-6230.
20. Mohebzada, J. G., El Zarka, A., Bhojani, A. H., & Darwish, A. (2022). The phishing landscape: A look at current trends and research. Future Generation Computer Systems, 128, 187-204.
21. Wu, C. Y., Kuo, C. C., & Yang, C. S. (2019). A phishing detection system based on machine learning. In 2019 International Conference on Intelligent Computing and its Emerging Applications (ICEA) (pp. 28-32). IEEE.
22. Alnemari, S., & Alshammari, M. (2023). Detecting phishing domains using machine learning. Applied Sciences, 13(8), 4649.
23. Alshingiti, Z., Alaqel, R., Al-Muhtadi, J., Haq, Q. E. U., Saleem, K., & Faheem, M. H. (2023). A deep learning-based phishing detection system using CNN, LSTM, and LSTM-CNN. Electronics, 12(1), 232.
24. Karim, A., Shahroz, M., Mustofa, K., Belhaouari, S. B., & Joga, S. R. K. (2023). Phishing detection system through hybrid machine learning based on URL. IEEE Access, 11, 36805-36822.
25. Aldakheel, E. A., Zakariah, M., Gashgari, G. A., Almarshad, F. A., & Alzahrani, A. I. (2023). A Deep learning-based innovative technique for phishing detection in modern security with uniform resource locators. Sensors, 23(9), 4403.
26. Choudhary, T., Mhapankar, S., Bhddha, R., Kharuk, A., & Patil, R. (2023). A Machine Learning Approach for Phishing Attack Detection. Journal of Artificial Intelligence and Technology, 3(3), 108-113.
27. Benavides-Astudillo, E., Fuertes, W., Sanchez-Gordon, S., Nuñez-Agurto, D., & Rodríguez-Galán, G. (2023). A phishing-attack-detection model using natural language processing and deep learning. Applied Sciences, 13(9), 5275.
28. Chinnasamy, P., Krishnamoorthy, P., Alankruthi, K., Mohanraj, T., Kumar, B. S., & Chandran, L. (2024). AI Enhanced Phishing Detection System. In 2024 Third International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS) (pp. 1-5). IEEE.
29. Siva, N., Sivaiah, B. V., Reddy, S. S., Irfan, S., Kumar, U. P., &Nayagara, S. N. (2024). Phishing Detection System through Hybrid Machine Learning Based on URL. In 2024 5th International Conference for Emerging Technology (INCET) (pp. 1-5).
30. Hasan, B. M. S., & Abdulazeez, A. M. (2021). A review of principal component analysis algorithm for dimensionality reduction. Journal of Soft Computing and Data Mining, 2(1), 20-30
31. Salem, N., & Hussein, S. (2019). Data dimensional reduction and principal components analysis. Procedia Computer Science, 163, 292-299.
32. Aksu, D., Abdulwakil, A., & Aydin, M. A. (2017). Detecting phishing websites using support vector machine algorithm. PressAcademia Procedia, 5(1), 139-142.
33. Anupam, S., & Kar, A. K. (2021). Phishing website detection using support vector machines and nature-inspired optimization algorithms. Telecommunication Systems, 76(1), 17-32.
34. Ahmad, W., Ayub, N., Ali, T., Irfan, M., Awais, M., Shiraz, M., & Glowacz, A. (2020). Towards short term electricity load forecasting using improved support vector machine and extreme learning machine. Energies, 13(11), 2907.
35. Zhang, Y., Hong, J., & Cranor, L. (2020). Cantina+: A feature-rich machine learning framework for detecting phishing websites. ACM Transactions on Information and System Security (TISSEC), 14(2), 1-28.
36. Ma, J., Saul, L. K., Savage, S., & Voelker, G. M. (2009). Beyond blacklists: Learning to detect malicious web sites from suspicious URLs. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2009, 1245-1254
37. Machado, E., Pinto, T., Guedes, V., & Morais, H. (2021). Electrical load demand forecasting using feed-forward neural networks. Energies, 14(22), 7644.
38. Muruganandam, S., Joshi, R., Suresh, P., Balakrishna, N., Kishore, K. H., &Manikanthan, S. V. (2023). A deep learning based feed forward artificial neural network to predict the K-barriers for intrusion detection using a wireless sensor network. Measurement: Sensors, 25, 100613.
39. Ketkar, N., Moolayil, J., Ketkar, N., &Moolayil, J. (2021). Feed-forward neural networks. Deep Learning with Python: Learn Best Practices of Deep Learning Models with PyTorch, 93-131.
40. Mohammad, R. M., Thabtah, F., & McCluskey, L. V. (2015). Intelligent rule-based phishing website classification. IET Information Security, 9(4), 267-277