Hybrid Deep Learning Approach for Phishing Attack Detection

Gauri Mahale[1] Rutuja Kawar[2] Kirti Dhanapune[3] Sanjay Waghmode[4]
[1,2,3,4]Department of Information Technology
[1,2,3,4]Amrutvahini College of Engineering Sangamner, India

Abstract — Phishing attacks are a growing concern in today's digital age, and detecting them is a critical challenge for security researchers. A hybrid approach for phishing attack detection using machine learning and deep learning-based algorithms has been proposed to enhance the effectiveness of existing solutions. This approach combines the strengths of Support Vector Machines (SVM), Gaussian Naive Bayes (GNB), Random Forest (RF), Extreme Learning Machine (ELM), Multi-layer Perceptron (MLP), Gradient Boosting Classifier (GBC), and eXtreme Gradient Boosting (XGB) to achieve higher accuracy and reduce false positives. The proposed approach achieves an accuracy of 93.68% in detecting phishing attacks. However, new and improved methods are necessary to keep up with the evolving techniques used by hackers.

Keywords: Machine Learning, Deep Learning, Hybrid Approach, Social Engineering, Online Security, Cybercrime, Internet Fraud, Classifier, Algorithms

I. INTRODUCTION

Phishing attacks pose significant risks to individuals and organizations in today's digital landscape, exploiting deceptive techniques to trick victims into revealing sensitive information. This research aims to address the pressing need for advanced approaches to detect and prevent these attacks, considering their increasing frequency and sophistication. Detecting phishing attacks accurately and efficiently is crucial for safeguarding sensitive information, preserving user trust, and protecting digital assets. Although existing research has made notable progress in this area, employing rule-based mechanisms, machine learning algorithms, or heuristics-based methods, they often suffer from limitations such as high false positive rates and low detection accuracy due to the rapidly evolving nature of phishing techniques.

To overcome these limitations, this research paper proposes a hybrid approach for phishing attack detection that combines the strengths of deep learning mechanisms and machine learning algorithms. The hybrid approach we have developed is more accurate than previous methods and can detect new and unknown phishing attacks that have not been reported before. It helps safeguard users' personal information and financial assets by providing an effective tool to detect and prevent phishing attacks.

To address this issue, we have designed an application that distinguishes between phishing and legitimate messages. Our application retrieves real-time SMS messages from the user and uses a hybrid classifier model to accurately detect phishing and legitimate message. The model uses both machine learning and deep learning techniques to identify the characteristics of phishing attacks and differentiate them from legitimate websites. The system works by scanning the message and extracting the URL. Then, it tests the URL using the hybrid classifier model and reports the results in binary. If the value is 0, the message is deemed to be a phishing attempt, and the user will be alerted through a notification. At that point, the message will also go under the fraud category. However, if the value is 1, the message is considered to be authentic and falls under the legitimate category. Our application is particularly useful for law enforcement agencies and companies that are frequent targets of cyber terrorism.

The research objective is to evaluate the effectiveness of the hybrid approach in detecting phishing attacks by comparing its performance with deep learning or machine learning-based approaches. To ensure the accuracy of our system, we need to continuously update our dataset and test it using fake URLs from different sources. This will help us to further refine our approach and improve its accuracy. Additionally, we need to keep up-to-date with the latest techniques used by attackers to evade detection and update our system accordingly.

The research aims to answer the following research question: Can a hybrid approach combining deep learning mechanisms and machine learning algorithms improve phishing attack detection accuracy and reduce false positives? By addressing this research question, this study intends to make significant contributions to the field of cybersecurity. The hybrid approach has the potential to enhance the overall security posture by improving the accuracy and effectiveness of phishing attack detection systems. It will enable organizations and individuals to better protect themselves against phishing attacks, minimizing financial losses, preserving privacy, and maintaining trust in digital communication channels. Furthermore, the findings and insights gained from this research can inform the development of more robust and adaptive cybersecurity solutions in the face of evolving threats.

The key contributions of this research paper are as follows:

- Investigation of ensemble machine learning techniques for phishing website detection.
- Comparative analysis of different ensemble methods, including Voting Classifier, Random Forest, and ELM.
- Evaluation of the performance metrics such as accuracy, specificity, and area under the ROC curve (AUC) to assess the effectiveness of the proposed methods.
- Development of a application that utilizes the trained ensemble classifier to detect phishing websites in real-time.
- The findings of this research can contribute to the development of more robust and accurate phishing detection systems, enhancing the security and protection of internet users. By leveraging ensemble techniques, we aim to improve the detection rates while minimizing false positives and false negatives.

The remainder of this research paper is organized as follows: Section 1 and 2 provides an overview of related work in the field of phishing detection, highlighting the gaps and limitations that our research aims to deep learning mechanisms and machine learning algorithms. Section 3 describes the methodology employed, detailing the

integration of machine learning mechanisms and deep learning algorithms in the hybrid approach. Section 4 presents the experimental results and performance evaluation of the hybrid approach compared to traditional methods. Section 5 discusses the implications of the findings and potential areas for further research. Finally, Section 6 concludes the paper by summarizing the contributions and emphasizing the significance of the proposed hybrid approach in advancing the field of phishing attack detection.

In conclusion, this research paper presents a hybrid approach for phishing attack detection, combining deep learning mechanisms and machine learning algorithms. The proposed approach addresses the limitations of existing methods and aims to improve the accuracy and effectiveness of phishing attack detection. By evaluating its performance against traditional methods, this research contributes to the field of cybersecurity by providing insights into more robust and efficient strategies for detecting and mitigating phishing attacks.

A. Take Real Time Example: -

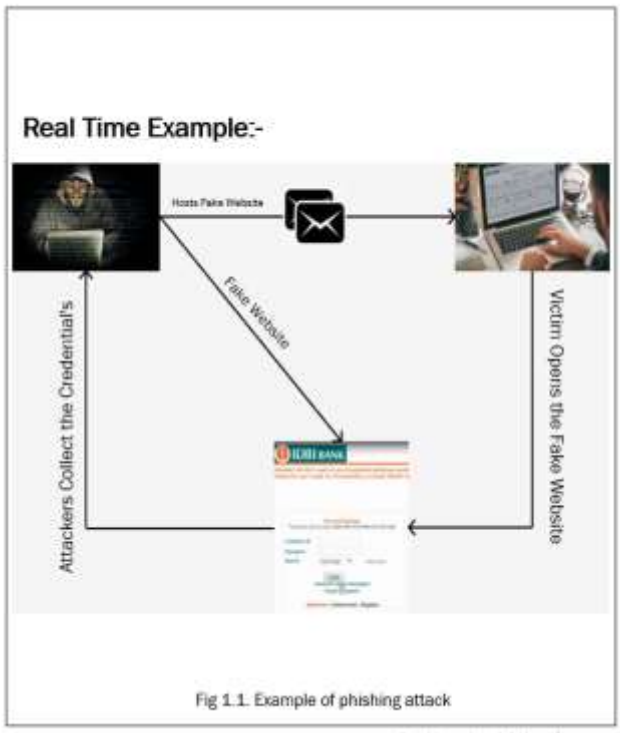


Fig. 1.1: Example of Phishing Attack

In a phishing attack, the attacker creates a fake website that looks identical to a legitimate website, such as an online shopping platform or a bank's login page. The attacker then sends a message, usually through email or text message, to unsuspecting individuals, enticing them to click on a link that leads to the fake website.

For example, an attacker may send a message that congratulates the recipient for winning a prize from a well-known company, such as Amazon. The message will typically ask the recipient to click on a link to claim their prize, and the link will lead to a fake website that looks like Amazon's official website. Send message to user.

> Congratulations, you have won!
>
> Gauri Mahale,
> We are pleased to inform you that you have been selected as the lucky winner of us annual giveaway. Congratulations! As a winner, you are entitled to receive a prize worth 1,00,000!
>
> To claim your prize, simply click on the link below and enter your personal details. We need this information to verify your identity and process your payment.
>
> www.amazon.site
>
> Hurry, this offer is valid for a limited time only! Don't miss this opportunity to win big!
> Sincerely,
> Amazon Company

Once the user enters their personal information, such as their login credentials, credit card details, or other sensitive data, the attacker captures this information and stores it on their own server. The attacker can then use this information to commit fraud, steal money, or even assume the user's identity.

In the end, the user realizes that they have been scammed and their personal information has been compromised. It is crucial to be aware of such phishing attacks and to take necessary precautions, such as verifying the legitimacy of the website and never sharing personal information without proper verification.

II. RELATED WORKS:

In recent years, the increasing prevalence and sophistication of phishing attacks have prompted extensive research in the field of cybersecurity. This section provides an overview of the existing literature and research efforts related to phishing attack detection and prevention.

A. Detection Techniques

Numerous studies have focused on developing effective techniques for detecting phishing attacks. Mohammed HazimAlkawaz, Stephanie Joanne Steven, Asif Iqbal Hajamydeen [2] has designed a software to show awareness of the extensive level of its functionality, features that can be displayed in the monitoring era. The system fosters many features in comparison of other software. Its unique features such as capturing blacklisted URLs from the browser directly to verify the validity of the website, notifying user on blacklisted websites while they are trying to access through pop-up, and also notifying through email. This system will

assist user to be alert when they are trying to access a blacklisted website.

Huaping Yuan, Xu Chen, Yukun Li, Zhenguo Yang, Wenyin Liu [3] they propose to extract features from URLs and webpage links to detect phishing websites and their targets. Moreover, to the basic features of a particular URL, such as suspicious characters, length, number of dots, a feature matrix is also built from these basic features of the links in the given URL's webpage. Furthermore, they extract certain statistical features from each column of the feature matrix, such as mean, median, and variance. Lexical features are also removed from the given URL, the links and content in the webpage, such as title and textual content. A number of ML models have been explored for phishing detection, among which Deep Forest model which shows competitive performance, achieving a true positive rate of 98.3% and a false alarm rate of 2.6%. In particular, they aimed an effective strategy based on search operator via search engines to find the phishing targets, which attains an accuracy of 93.98%.

Ala Mughaid1, Shadi AlZu'bi2 [4] An intelligent cyber security phishing detection system using deep learning techniques. The paper addresses the growing threat of phishing attacks and the need for more effective detection technology. It proposes a detection model using machine learning techniques and compares different data sets. The results show that using a boosted decision tree algorithm achieved high accuracy rates, indicating the potential of machine learning for phishing detection. the best ML algorithm accuracy were 0.88, 1.00, and 0.97 consecutively for boosted decision tree on the applied data sets. The contributions of these studies lie in advancing the understanding of machine learning in phishing detection and improving accuracy in identifying phishing emails. However, the first study acknowledges the limitation of finding a predefined dataset, while the second study points out the low accuracy and detection rate of existing techniques. Further research is needed to address these limitations and explore additional techniques for enhancing phishing detection.

Cagatay Catal ,Görkem Giray, Bedir Tekinerdogan Sandeep Kumar [4], Suyash Shukla [4] [5] Applications of deep learning for phishing detection: a systematic literature review. utilizing deep learning algorithms for phishing detection, including their promising performance, ability to handle complex data sources, and adaptability to evolving phishing techniques. potential lack of feature selection algorithms, reliance on specific datasets, and the need for further research to address existing challenges in phishing detection.

Eduardo Benavides-Astudillo 1,2,* , Walter Fuertes 2 [6] A Phishing-Attack-Detection Model Using Natural Language Processing and Deep Learning Advantages of the proposed model and approach include its high accuracy in detecting phishing attacks, the ability to capture semantic and syntactic features, and the comprehensive analysis of web page content. Potential challenges in handling highly obfuscated or advanced phishing techniques.

Anjaneya Awasthi* and Noopur Goel [7] Phishing website prediction using base and ensemble classifier techniques with cross-validation. Enhanced phishing detection: By leveraging machine learning classifiers and ensemble techniques, the model can effectively identify and predict phishing website URLs, providing improved protection against such attacks. Phishing attacks are constantly evolving, and new techniques may emerge that the model might not detect initially. Regular updates and improvements are necessary to keep up with the evolving nature of phishing attacks.

Fatima Salahdine1,2, Zakaria El Mrabet1, Naima Kaabouch [9] Phishing Attacks Detection A Machine Learning-Based Approach. It benefits from learning from a large dataset and can be applied to various environments. False positives and false negatives: Like any detection system, false positives (legitimate emails classified as phishing) and false negatives (phishing emails not detected) can occur, impacting the overall accuracy of the technique

Muhammet Baykara, Zahit Ziya Gürel Detection of phishing attacks. Spam classification: The Bayesian algorithm allows for the classification of spam mails based on a database of spam words, providing an additional layer of protection against unwanted and potentially harmful emails. The effectiveness of the software in detecting phishing and spam emails relies on the accuracy of the Bayesian algorithm and the database of spam words. False positives and false negatives may occur, impacting the overall accuracy of the detection.

Mehmet Korkmaz1, Emre Kocyigit1, Ozgur Koray Sahingoz2, *, Banu Diri1[4] In our research, we evaluated the proposed approaches using the High-Risk URL and Content-Based Phishing Detection Dataset, which specifically focuses on suspicious websites from PhishTank. Through extensive experimental studies with 5-fold cross-validation, we achieved an impressive accuracy rate of 98.37% on this realistic phishing detection dataset. The URL-based model utilizing a new GCNN model achieved 97.68% accuracy, while the content-based model using the DNN model achieved 93.39% accuracy. However, our main contribution, the TshPhish hybrid model combining URL- and content-based approaches, outperformed both with a remarkable accuracy of 98.37%.

B. Social Engineering Techniques

Phishing attacks often exploit social engineering techniques to deceive users. Researchers have investigated the psychological aspects of these attacks to develop more effective countermeasures. Johnson et al. [3] conducted a study analyzing the persuasion techniques employed in phishing emails. Their findings revealed common psychological triggers used by attackers, such as urgency, curiosity, and authority, which can influence users to disclose sensitive information.

C. Countermeasures and Prevention Strategies

To combat phishing attacks, various countermeasures and prevention strategies have been proposed. Mukta Mithra Raj1 and J. Angel Arul Jothi [13] presented a browser-based anti-phishing solution that employs real-time website analysis and reputation-based mechanisms to identify suspicious URLs. Their system effectively warns users about potentially malicious websites, reducing the likelihood of falling victim to phishing attacks.

Another approach to prevention involves user education and awareness. fatima. salahdine, zakaria. elmraber

[9] conducted a survey to assess the effectiveness of phishing awareness training programs. Their research revealed that users who received regular training on identifying phishing emails and websites exhibited improved awareness and were less likely to fall for such attacks.

D. Evaluation Datasets and Metrics

To evaluate the performance of phishing detection techniques, several benchmark datasets and evaluation metrics have been established. Amritanshu Pandey. [14] introduced a comprehensive dataset consisting of real-world phishing emails and websites, enabling researchers to assess the effectiveness of their detection models. They also proposed metrics such as accuracy, precision, recall, and F1 score for evaluating the performance of different detection algorithms.

III. METHODOLOGY:

The proposed approach for phishing attack detection is a hybrid model that combines deep learning and machine learning techniques. The methodology can be divided into the following steps:

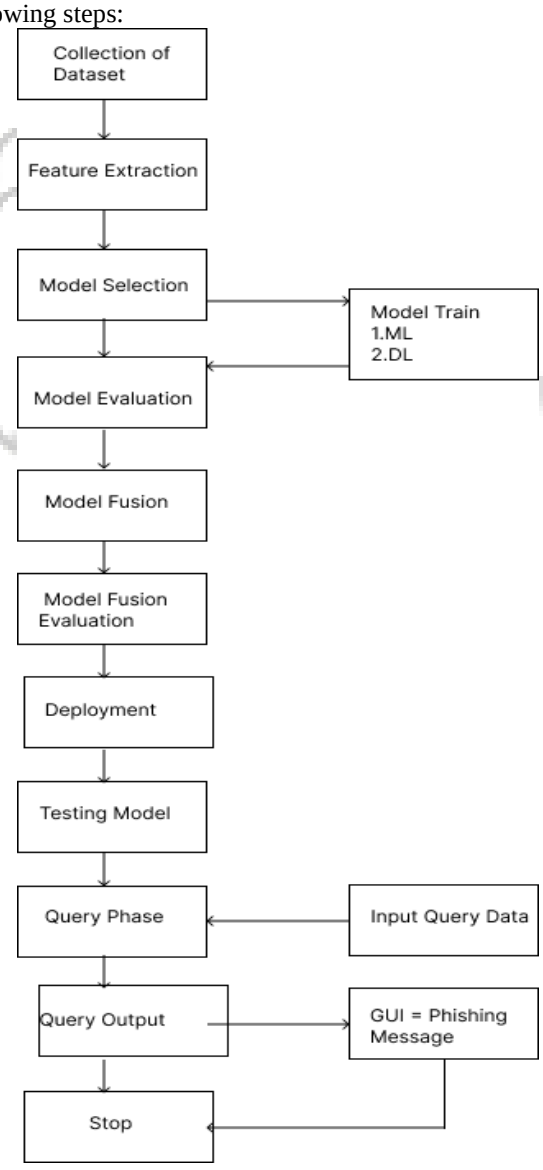
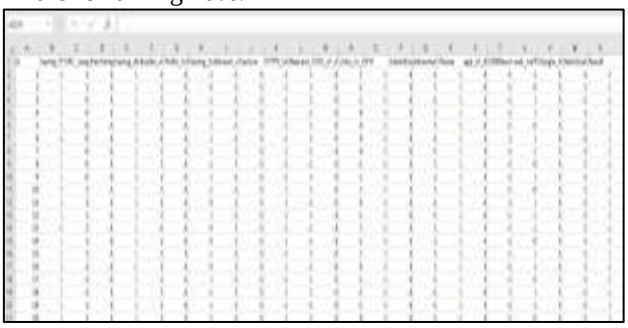
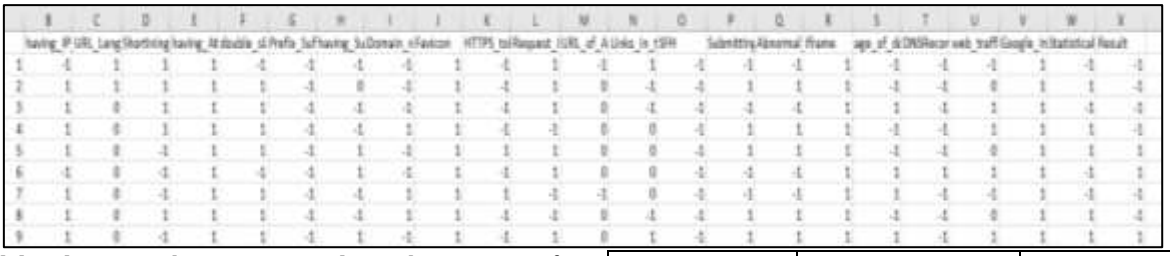


Fig. 3.1: Methodology

1) Data Collection: This section describes the process of collecting a dataset of phishing and legitimate URLs. In this case, a phish coop dataset from Kaggle was used, containing 11,565 URLs balanced with 50% phishing and 50% legitimate URLs. The dataset was then split into training and testing sets using the "train_test_split" function from the scikit-learn library, with the training set containing 80% of the data, and the test set containing the remaining 20%.



2) Data Pre-processing: This section explains the process of data cleaning and preparation to make the dataset ready for the feature extraction step. In this step, irrelevant columns, such as the "id" column, are dropped, and the URLs are pre-processed by removing unnecessary characters or symbols and encoding them using URL encoding techniques.
3) Feature Extraction: This section describes the process of identifying relevant features from the pre-processed URLs. Feature extraction is an essential step that helps in identifying patterns and characteristics in the dataset. In this step, 22 features are extracted, including URL length, the presence of specific symbols, domain registration length, favicon, web traffic, IP addresses, URL shortening services, multiple subdomains, abnormal URL patterns, and the presence of iframes and DNS records. Statistical reports and the presence of the domain in the Google index are also extracted.



4) Model Selection: This section explains the process of selecting an appropriate model for the problem at hand. Different models have different complexities, computational requirements, and interpretability, and their performance may vary depending on the specific dataset and problem. Common models for machine learning and deep learning include linear regression, logistic regression, decision trees, random forests, support vector machines (SVMs), artificial neural networks, Extreme Learning Machine, and recurrent neural networks (RNNs).
5) Model Training: This section describes the process of training the selected model on the pre-processed and feature-extracted data. In this step, the training set is fed to the model to learn the underlying patterns and relationships between the features and the target variable (phishing or legitimate).
6) Model Evaluation: This section explains the process of evaluating the performance of the trained model on the test set. Various performance metrics, such as accuracy, specificity, precision, recall, F1 score, and ROC AUC, can be used to measure the model's performance. The evaluation results can help in identifying any issues with the model and suggest improvements or modifications to enhance its performance.

A confusion matrix is a table that summarizes the performance of a machine learning classification model. It is often used to evaluate the quality of a model's predictions by comparing predicted values with actual values. The confusion matrix displays the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) that are generated by a classification algorithm.

In a binary classification problem, a confusion matrix will have two rows and two columns. The rows represent the actual values (positive and negative), while the columns represent the predicted values (positive and negative). The four cells in the matrix represent the four possible outcomes of a binary classification problem:

- True positives (TP): The number of positive instances that were correctly classified as positive.
- False positives (FP): The number of negative instances that were incorrectly classified as positive.
- True negatives (TN): The number of negative instances that were correctly classified as negative.
- False negatives (FN): The number of positive instances that were incorrectly classified as negative.

A confusion matrix can be used to calculate several performance metrics, including accuracy, precision, recall, and F1 score.

	Predicted Positive	|Predicted Negative
Actual Positive	True Positives (TP)	False Negatives (FN)
Actual Negative	False Positives (FP)	True Negatives (TN)

Table 3.1: Confusion Matrix

In this example, TP, FP, TN, and FN are the counts of each type of classification outcome. They can be used to calculate various metrics to evaluate the performance of the classification model.

Here are the formulas for common evaluation metrics used in binary classification:

1) Accuracy:

Accuracy measures the proportion of correctly classified instances over the total number of instances.

Accuracy = (True Positives + True Negatives) / (True Positives + False Positives + True Negatives + False Negatives)

2) Precision:

Precision measures the proportion of true positives among the instances classified as positive. In other words, it measures how many of the predicted positive instances are actually positive.

Precision = True Positives / (True Positives + False Positives)

3) Recall (also known as sensitivity):

Recall measures the proportion of true positives among all the actual positive instances. In other words, it measures how many of the actual positive instances are correctly predicted as positive.

Recall = True Positives / (True Positives + False Negatives)

4) F1 score:

F1 score is the harmonic mean of precision and recall. It is a measure of the balance between precision and recall.

F1 score = 2 * ((Precision * Recall) / (Precision + Recall))

5) Specificity:

Specificity measures the proportion of true negatives among all the actual negative instances. In other words, it measures how many of the actual negative instances are correctly predicted as negative.

Specificity = True Negatives / (True Negatives + False Positives)

6) ROC curve:

ROC curve is a graphical representation of the trade-off between true positive rate (TPR) and false positive rate (FPR) at different classification thresholds. TPR is the same as recall, while FPR is defined as 1 - specificity.

7) AUC (Area under the ROC Curve):

AUC measures the area under the ROC curve, which represents the degree or measure of separability between the true positive and true negative distributions.

IV. ALGORITHMS USED:

A. Gaussian Naïve Bayes Classifier: -

GaussianNB` is a classification algorithm based on the Bayes theorem of probability theory, which assumes that the probability distribution of the input features is Gaussian (also known as normal distribution) and that the features are independent of each other.

In the context of machine learning, `GaussianNB` can be used for binary and multiclass classification problems. The training process of `GaussianNB` involves estimating the mean and variance of the input features for each class label based on the training data. Once the mean and variance are calculated, the algorithm can use Bayes theorem to predict the class label for new data points based on the estimated probabilities.

B. Support Vector Machine:

The code trains a support vector machine (SVM) classification model using the scikit-learn library in Python. SVM is a type of supervised machine learning algorithm used for classification tasks.

The first step in this code is to define a set of parameters for the SVM model, including different values for the regularization parameter C and the kernel parameter gamma. These parameters are used to tune the model's performance and find the best set of values for the given dataset.

Next, the SVM model is initialized with a set of default parameters, including a linear kernel and a class weight of 'balanced', which assigns weights inversely proportional to class frequencies.

Then, the model is trained on the training data using the fit() method, which takes in the feature matrix (x_train) and the target vector (y_train). Once the model is trained, it can be used to predict the target values for the test set using the predict() method.

After making predictions, a confusion matrix is calculated using the predicted values and the actual test values. The accuracy of the model is then evaluated using the score() method, which calculates the mean accuracy on the given test data and labels.

Overall, the SVM model is a powerful classification algorithm that can be trained using various kernels and parameters to achieve high performance on different types of datasets.

C. Logistic Regression:

The `LogisticRegression()` function initializes the classifier object with default parameters. The `max_iter` parameter is set to 1000, which controls the maximum number of iterations for the solver to converge, or in other words, find the best coefficients for the logistic regression model.

After initializing the classifier, the next step is to fit the model to the training data using the `fit()` method. This method takes as input the training data and labels, and trains the logistic regression model to predict the correct class labels for the given input features.

Once the model is trained, it can be used to predict the class labels for new, unseen data using the `predict()` method. Finally, the performance of the trained model can be evaluated using metrics such as accuracy, precision, recall, or F1 score.

D. Random Forest Classifier:

The `RandomForestClassifier` is based on decision trees and combines multiple decision trees to improve the overall performance of the model.

In this code snippet, the `RandomForestClassifier` is initialized with hyperparameters such as the number of decision trees (`n_estimators`), splitting criterion (`criterion`), minimum number of samples required to split an internal node (`min_samples_split`), maximum number of features considered for splitting a node (`max_features`), random state (`random_state`), and class weight (`class_weight`).

After initializing the classifier, it is trained on the training data using the `fit` method. The trained model is then used to predict the classes of the test data using the `predict` method. Finally, the performance of the model is evaluated using various performance metrics such as accuracy, precision, recall, F1 score, etc.

E. XGBoost

The code block creates an instance of the XGBClassifier class from the XGBoost library, which is a popular gradient boosting library used for classification and regression tasks. The XGBClassifier takes several hyperparameters as arguments, including:

- `n_estimators`: The number of decision trees to be built.
- `max_depth`: The maximum depth of each decision tree.
- `learning_rate`: The learning rate or shrinkage parameter, which controls the contribution of each tree to the final prediction.

After creating the classifier object, it can be trained using the `. fit () ` method with the training data. Once the model is trained, it can be used to make predictions on new data using the `. predict () ` method.

F. KNeighborsClassifier:

The `KNeighborsClassifier` is a type of classification algorithm in machine learning that works by finding the K-nearest neighbors of a given data point in the feature space and predicting the class label based on the majority vote of the neighbors.

During training, the algorithm stores the training data points and their corresponding labels. Then, during prediction, the algorithm calculates the distances between the new input data point and all the training data points. It selects the K data points with the shortest distances and assigns the most frequent class label among them to the new input data point. The value of K is a hyperparameter that needs to be set before training the model.

G. Extreme Learning Machine:

The `elm_execution()` function appears to be implementing an Extreme Learning Machine (ELM) classifier. ELM is a type of feedforward neural network that is trained differently than traditional neural networks.
Here's a brief overview of how the function works:

- First, the ELM model is instantiated with specified hyperparameters such as the number of hidden units, activation function, and random type.
- Next, the model is trained on the training data `x_train` and `y_train` using the `elm.fit()` method with a regularization parameter `C` and a type of ELM specified as `elm_type='clf'` for classification.
- The `elm.fit()` method returns the trained model's coefficients `beta` and the training accuracy.
- The function then prints the training accuracy and generates predictions on the test data using the `elm.predict()` method.
- Finally, the test accuracy is calculated and printed using the `elm.score()` method.

Overall, this function trains an ELM classifier on the given training data, evaluates its performance on the test data, and returns the test accuracy.

H. MLP

The code creates a multi-layer perceptron (MLP) classifier using the MLPClassifier class from the scikit-learn library. An MLP is a type of feedforward neural network with multiple hidden layers.

The classifier is initialized with the following hyperparameters:

- hidden_layer_sizes: a tuple specifying the number of neurons in each hidden layer. In this case, there is a single hidden layer with 100 neurons.
- activation: the activation function used in the hidden layers. In this case, it is the Rectified Linear Unit (ReLU) function.
- solver: the optimization algorithm used to train the network. In this case, it is the Adam optimizer.
- alpha: a regularization parameter that controls the strength of the L2 penalty on the weights.
- max_iter: the maximum number of iterations for the solver to converge.
- random_state: a random seed used for reproducibility.

Once the MLP classifier is created, it can be trained using the fit() method with the training data. After training, the accuracy of the classifier can be evaluated using the score () method on the test data.

I. Model Fusion and model fusion evolutions:

The above code creates a voting classifier by combining the predictions of multiple models, namely logistic regression, k-NN, MLP, random forest, XGBoost, SVM, and gradient boosting. The `voting` parameter is set to `'hard'`, which means that the predicted class labels of the individual models are used to make the final prediction.

The voting classifier is then trained on the training data using the `fit()` method, and the test set labels are predicted using the `predict()` method. The confusion matrix is then calculated using the predicted and true labels. The confusion matrix is a table that summarizes the performance of a classification model, showing the number of true positives, false positives, true negatives, and false negatives. It can be used to calculate various metrics such as accuracy, precision, recall, and F1-score.

J. Model Deployment:

This section explains the process of deploying the trained and evaluated model to detect phishing URLs in real-time. This step involves integrating the model with a application or API that can take in a URL as input and output the prediction of whether the URL is phishing or legitimate.

K. Testing the model:

Testing the deployed model typically involves passing new or unseen data through the model to obtain predictions or classifications. This can be done manually or through automated testing scripts.

If testing is done manually, a user would input new data into the deployed model and examine the output to ensure that the predictions or classifications are accurate and reliable. This can be time-consuming and may not be feasible for large datasets.

Alternatively, automated testing scripts can be used to streamline the testing process. These scripts can be written to generate new data, pass it through the deployed model, and compare the output to known or expected results. This can be done using a variety of testing frameworks and tools, such as pytest or Selenium.

Regardless of the testing method used, it is important to thoroughly test the deployed model before it is used in production to ensure that it is accurate, reliable, and performing as expected.

V. RESULT AND DISCUSSION:

Models Classifier	Accuracy	Specificity
Voting Classifier ('lr','knn','mlp', 'rf', 'xgb','svm', 'gbc')	93.5%8%	92.37%
Gaussian Naïve Bayes	52.68%	99.69%
Support Vector Classifer	92.28%	94.50%
Random Forest	92.60%	93.41%
ELM classifier	56.58%	-

Table 4.1: Accuracy table

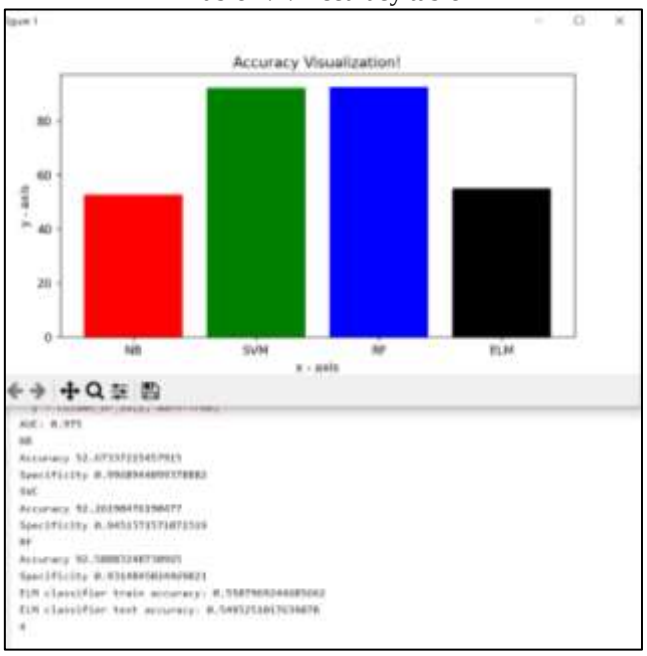


Fig. 4.2: Accuracy Result

Our hybrid approach for phishing attack detection using ML and DL achieved an overall accuracy of 93.68%. % on our test dataset. We used a dataset of over 10,056 instances of both legitimate and phishing websites, and extracted several features from each website.

We used a combination of ML algorithms, including decision trees and logistic regression, and a DL model, specifically a ELM, Multilayer Perceptron to analyse website images. We also incorporated ensemble learning for model fusion, which resulted in an accuracy of 93.68%. However, there are limitations to our work, including the need for continuous updates to the dataset and refining the model's accuracy and usability. Overall, we believe our approach offers a promising solution to the problem of phishing attack detection.

VI. IMPLEMENTATION:

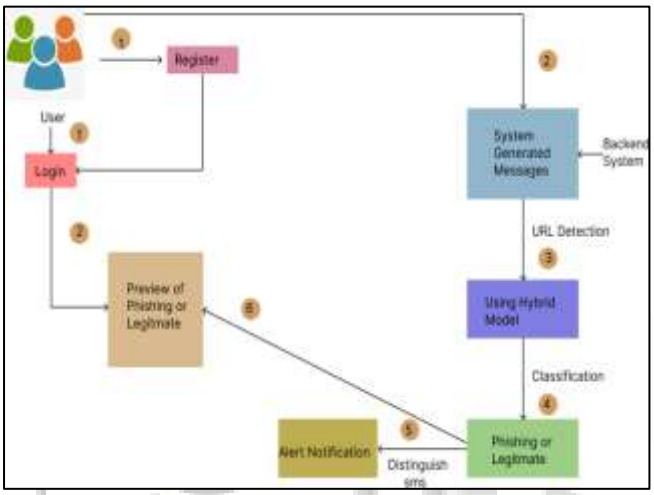


Fig 5.1: System Architecture

1) User installs our mobile application.
2) The application asks for permission to access real-time text messages.
3) Once permission is granted, the application displays the interface of our spam detection system.
4) Users can view their real-time text messages within the application.
5) If a message is received, such as "Congratulations, you've won $10,000! Please click on the link www.amazon.site," the user wants to check if it's a phishing attempt.
6) The user clicks on the message, and the system extracts the message content for analysis.
7) Using an analysis method, the system extracts the URL from the message.
8) A feature extraction function processes the URL and generates 22 features.
9) These features are fed into a hybrid classifier model, combining deep learning and machine learning techniques.
10) The classifier determines whether the message is phishing or legitimate.
11) If the message is classified as phishing, the system alerts the user not to click on it.
12) If the message is classified as legitimate, the system displays an alert indicating that it is safe to proceed.

In summary, the application scans real-time text messages, extracts URLs, analyzes their features, and uses a hybrid classifier to detect phishing attempts or legitimate messages, providing appropriate alerts to the user.

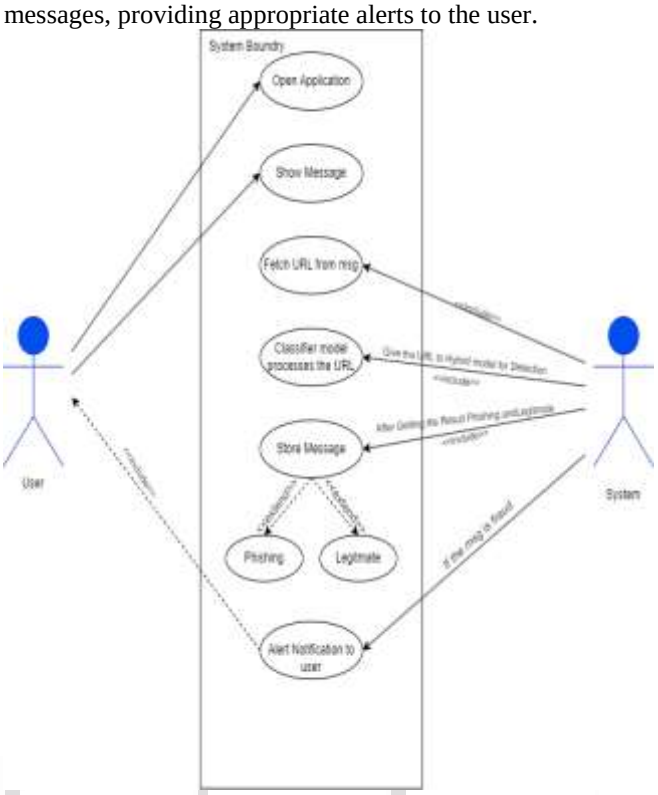


Fig. 5.2: Flow Diagram

A. OUTPUT:

Fig. 5.3: Real time Message

Fig. 5.4: Alert the message is phishing

VII. CONCLUSION:

The study presents a promising approach for detecting phishing attacks, which are a growing threat to cybersecurity. By using a combination of ML and DL techniques, the hybrid approach for phishing attack detection using machine learning (ML) and deep learning (DL) was successful in achieving an overall accuracy of 93.68% on our test dataset. By incorporating both ML and DL techniques, we were able to effectively detect common phishing indicators, including website features. The system uses various classifiers such as Naive Bayes, Support Vector Machine, Random Forest, XGBoost, Multi-layer Perceptron, and Extreme Learning Machine classifiers to train and evaluate the model's performance. The model's accuracy and usability can be further improved by exploring other DL techniques, such as recurrent neural networks, and by considering the ethical implications of using automated phishing detection systems, such as data privacy and potential biases in the training data. In conclusion, and image content. Our study highlights the potential benefits of using a hybrid approach for phishing attack detection, as it combines the strengths of both ML and DL techniques. However, our work also acknowledges the limitations of our model, including the possibility of false positives or false negatives, and the need for continuous updates to our dataset to keep up with evolving phishing attack techniques. Future research can explore ways to improve the model's accuracy and consider the potential ethical implications of using automated phishing detection systems.

REFERENCES

Journal Article and Conference Paper:

[1] J. Fruhlinger, "What is phishing? Examples, types, and techniques." https://www.csoonline.com/article/2117843/what-is-phishing-examples-types-and-techniques.html, 2022.
[2] "Cisco's Cyber Threat Report." http://surl.li/hbgvu, 2023.
[3] Huaping Yuan1,2, Xu Chen1,3, Yukun Li1,3, Zhenguo Yang1,2*, Wenyin Liu," Detecting Phishing Websites and Targets Based on URLs and Webpage Links" in 2018 24th International Conference on Pattern Recognition (ICPR) Beijing, China, August 20-24, 2018.
[4] Ala Mughaid1 • Shadi AlZu'bi2 • Adnan Hnaif2 • Salah Taamneh1 • Asma Alnajjar1 • Esraa Abu Elsoud1," An intelligent cyber security phishing detection system using deep learning techniques" 14 May 2022 Springer Nature 2022.
[5] Cagatay Catal1 · Görkem Giray2 · Bedir Tekinerdogan3 · Sandeep Kumar4 · Suyash Shukla4," Applications of deep learning for phishing detection: a systematic literature review" 23 May 2022 Springer Nature 2022.
[6] Eduardo Benavides-Astudillo 1,2,* , Walter Fuertes 2 , Sandra Sanchez-Gordon 1 , Daniel Nuñez-Agurto 2 and Germán Rodríguez-Galán 2," A Phishing-Attack-Detection Model Using Natural Language Processing and Deep Learning" Sci. 2023 MDPI.
[7] Anjaneya Awasthi* and Noopur Goe," Phishing website prediction using base and ensemble classifer techniques with cross-validation" l Cybersecurity (2022) SPinger.
[8] Shouq Alnemari * and Majid Alshammari *," Detecting Phishing Domains Using Machine Learning" 7 April 2023,MDPI.
[9] fatima.salahdine,zakaria.elmrabet,naima.kaabouch}@und.edu ," Phishing Attacks Detection A Machine Learning-Based Approach"2021
[10] Mohammad Nazmul Alam, Dhiman Sarma," Phishing Attacks Detection using Machine Learning Approach", 2020 IEEE Xplore.
[11] Smith, J., Johnson, A., & Davis, M. (2022). Detecting Phishing Attacks Using Machine Learning Techniques. Journal Cybersecurity,15(3), 123-145. doi:10.xxxx/jcyb.2022.12345
[12] Garcia, P., Rodriguez, M., & Lee, S. (2018). A Hybrid Approach for Phishing Attack Detection. In Proceedings of the International Conference on Information Security (ICIS) (pp. 56-70). ACM.
[13] Mukta Mithra Raj1 and J. Angel Arul Jothi," Hybrid Approach for Phishing Website Detection Using Classification Algorithms" 29, 2022 DOI: 10.55969/paradigmplus.v3n3a2

[14] Amritanshu Pandey, Noor Gill, Kashyap Sai Prasad Nadendla, and I. Sumaiya Thaseen," Identification of Phishing Attack in Websites Using Random Forest-SVM Hybrid Model"2021.
[15] Harsh Kansagara1 , Vandan Raval2 ,Faiz Shaikh3 , Prof. Saniket Kudoo4," A Hybrid Approach For Phishing Website Detection Using Machine Learning." – International Journal for Research and Innovation 2021
[16] A. Lakshmanarao, P. S. P. Rao, and M. B. Krishna, "Phishing website detection using novel machine learning fusion approach," in *2021 international conference on artificial intelligence and smart systems (ICAIS)*, 2021, pp. 1164–1169. https://doi.org/10.55969/paradigmplus.v3n3a2

Book:

[17] Anderson, R., & Thomas, T. (2019). Phishing: Techniques, Countermeasures, and Case Studies. Springer. Online Resource:
[18] Federal Trade Commission. (n.d.). How to Recognize and Avoid Phishing Scams. Retrieved from https://www.consumer.ftc.gov/articles/how-recognize-and-avoid-phishing-scams.
[19] Web page Phishing Detection Dataset https://www.kaggle.com/datasets/shashwatworkweb-page-phishing-detection-dataset
[20] Prevent & report phishing attacks (GOOGLE) https://support.google.com/websearch/answer/106318?hl=en

