Journal of Information Technology and Digital World (ISSN: 2582-418X)
www.irojournals.com/itdw/

Outsmarting Phishers: A Comparative Analysis of Machine Learning Techniques

Gobika G.[1], Vidhyabharathi T.[2], Sangeetha V.[3]

[1,2]Student, [3]Assistant professor, Department of Computer Science with Data Analytics, Dr. N.G.P. Arts and Science College, Bharathiar University, Coimbatore, India.

E-mail: [1]gobika2324@gmail.com, [2]vidhyabharathi2401@gmail.com, [3]sangeetha.v@drngpasc.ac.in

Abstract

Phishing attacks threaten the security of the internet by stealing confidential data and money as well. As a way to prevent phishing, an extensive comparative study of the top most machine learning methods for phishing site detection was carried out. This research analyses the performance of ANN, RNN, XGBoost and Random Forest algorithms in the identification of phishing websites using the Kaggle dataset. These algorithms were selected due to their ability to uncover intricate associations and patterns from website information. The review examines the advantages and disadvantages each algorithm presents and compares them to each other based on accuracy efficient, precision, recall, F1 score, and computing efficiency. Through the comparison of these algorithms, the most effective algorithm for phishing detection is revealed, which can be useful to scholars and experts who focus on the improvement of on-line security. The research helps deposit the foundations for attacks prevention and facilitates the protection of online sensitive information. This study shows the effectiveness of using machine learning in the field of cybersecurity, especially with focus on the algorithms and how they can be optimized.

Keywords: Phishing Detection, Machine Learning, Website security, URL, Random Forest, XGBoost, Artificial Neural Network (ANN), Recurrent Neural Network (RNN), Classification Algorithm, Kaggle Datasets, Website security, Cybersecurity.

DOI: https://doi.org/10.36548/jitdw.2024.4.003
Received: 14-12-2024. received in revised form: 02-01-2025, accepted: 16-01-2025, published: 27-01-2025
© 2024 Inventive Research Organization. This is an open access article under the Creative Commons Attribution-NonCommercial International (CC BY-NC 4.0) License

1. Introduction

Indeed, in the conditions where we live, phishing is the most common cyberattack that every place has been confronted with the most serious consequences for individuals, businesses, and government agencies. Phishing is a form of attack that subjugates the reliance of users by means of fake websites through which a user enters confidential data such as login credentials, financial details, and other personal information. The most daunting aspect of the phishing attack is not only constant arising frequencies but also changes in sophistication such that most conventional measures for securing it fail. Consequently, there is an increasing urgency to develop advanced mechanisms for real-time identification and blocking of phishing websites [13].

Machine Learning has shown much promise and has been empowered by automated detection of phishing websites by features or patterns for those websites [14]. An important point separating these models from traditional rule-based methods is that they are able to learn from data, which can make them applicable to new threats, while being faster and more accurate in detection. Different algorithms like Decision Trees, Random Forest, Support Vector Machines (SVM), Neural networks do their work by detecting phishing, but the performance varies with each one of them [15,16].

The challenge is to find which is the best algorithm that can effectively differentiate phishing websites from legitimate ones.

The current study provides an elaborate comparative analysis of the prominent machine learning algorithms used in order to establish the strengths and weaknesses of each by evaluating their detection performance against an actual phishing attack. It is expected that such analysis will yield valuable insights towards optimization of phishing detection systems in a bid to safeguard individuals and organizations from the catastrophic damages triggered by phishers.

2. Related Work

Phishing website detection with ML has been under active research, with algorithms applied in the classification into genuine and phishing sites. This study compares four top machine learning algorithms such as Artificial Neural Networks (ANN), Recurrent Neural Networks (RNN), Extreme Gradient Boosting (XGBoost), and Random Forest (RF) [1-4].

These models perform well when it comes to identifying phishers and this study further investigates them by making them work with Phishing Website Detection, Kaggle dataset.

2.1 Phishing Detection using Artificial Neural Networks (ANN)

The use of artificial neural networks in detecting phishing websites is widely known because these networks can learn complex patterns in data. Earlier on, ANN were used to classify phishing websites, an action with promising results. In the same manner, researchers reported that ANN models produced an efficiency value of 96%. The present study further supports these findings, confirming similar results. The present ANN results are consistent with other earlier findings, which yield an accuracy level of 96.92%. The ANN detects very complex relationships of the features and provides effective ways to apply accurate measures in the phishing detection process[5,6].

2.2 Phishing Detection using Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNNs) has shown that it does particularly well with complex sequence-based tasks. This statement is equally true for the Long Short-Term Memory (LSTM) networks used as a subset of RNN. Thus, Long Short-Term Memory recurrent neural nets were applied to phishing detection and did quite well by looking at URL structure. The work has also been with Recurrent Neural Networks (RNN), resulting in an improved accuracy thus showing the capability of the LSTM model in learning sequential patterns in URL structure and domain information [7-9].

2.3 Phishing Detection using XGBoost

Usage for classification tasks has been improvised on the part of XGBoost, which has made it highly efficient. The XGBoost was used for phishing website detection and, in this case, several URL features that utilized high accuracy have been listed ever since. Much of the literature has shown XGBoost outperforming many techniques in different uses. Of all the used classifiers in the comparison of this dataset, XGBoost is the one with the highest accuracy, bringing it ahead of ANN, RNN, and Random Forest. These results point out that XGBoost demonstrates the most effective performance with regard to handling the most complex interactions [10-12].

2.4 Phishing Detection using Random Forest (RF)

An ensemble method such as Random Forest (RF) is commonly utilized for classification tasks due to being quite robust and capable of handling large-feature sets. Many similar studies have identified the strength of RF. This study intended to analyse the performance of RF and compare it with XGBoost. The accuracy rate is not very far from that of XGBoost but still competitive, even for high-dimensional data, showing how durable the model can be in the field of prediction.

3. Proposed Work

An illustration of the methodology adopted in this study for detecting phishing websites using machine learning algorithms is shown in Figure 1.

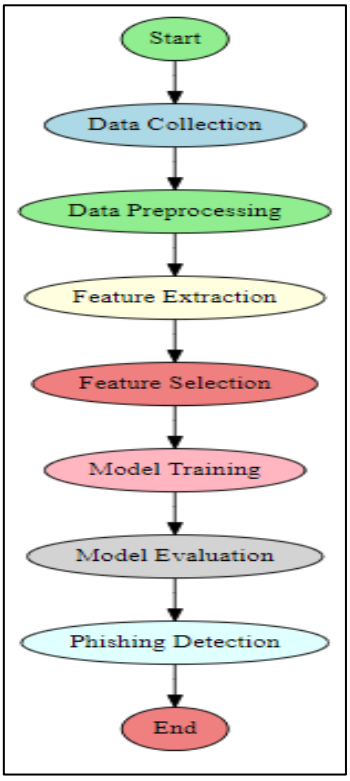

Figure 1. An Approach for Phishing Site Detection.

3.1 Machine Learning Techniques

3.1.1 Artificial Neural Networks (ANN)

Artificial Neural Networks (ANN) are one branch of machine learning models that are inspired by the biological structure of the human brain. An ANN is composed of connected

layers of nodes (neurons), which react to incoming signals in a similar way to biological neurons. The most classic kind of ANN for classification tasks is the feedforward neural network in which information flows from input layer to one or more hidden layers culminating in the output layer.

3.1.2 Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNN) are neural network architectures specifically crafted for sequencing work. This contrasts with traditional feedforward neural networks, as it contains loops that allow information to be passed from one step to the next, thus lending them particularly well to time-series type data or data with sequential influences among elements. While simple RNNs limit themselves by problems like vanishing gradients, superior revisions have lately been proposed, like Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU).

3.1.3 Extreme Gradient Boosting (XGBoost)

Extreme Gradient Boosting (XGBoost) is a leading method in gradient boosting applied to ensemble learning. Known for its speed and accuracy, XGBoost stands out as the most used tool in many of the machine learning tasks. It builds an ensemble of decision trees, adding every new one on each iteration to fix the errors in the decisions of the previous trees. Every tree is trained and optimized to reduce the residual errors in the ensemble.

3.1.4 Random Forest (RF)

Random Forest (RF) is an ensemble learning methodology built on the concept of bagging (bootstrap aggregating). Random Forest trains many decision trees using a random subset of the data and someday combines output with the majority rule to make final predictions for classification. For regression, the output average of predictions from different decision trees is considered final.

3.2 Data Collection

This research employed a dataset known as the "Phishing Website Dataset"(https://www.kaggle.com/datasets/shashwatwork/web-page-phishing-detection-dataset) acquired from the Kaggle site. The dataset consists of 89 attributes that are based on random phishing and legitimate sites. It has a total of 11,430 entries and balanced classes, thus

supporting training and evaluating machine learning models. Key Features for Phishing Website Detection are

1. **URL-based Features**

URL Length, Number of Dots in the URL, Number of Slashes in the URL, Number of Hyphens in the URL, Presence of Suspicious Keywords, Presence of HTTPS, Presence of IP Address in the URL. Features are directly obtained from the URL and might prove substantially useful in bringing about real understanding of site characteristics.

2. **Domain Features**

Domain Name Length, Subdomain Count, Suspicious Domain. These features are extracted from the domain of the URL (the main part of the URL, excluding subdomains and TLD).

3. **Other Content-based Features**

DNS Lookup, Domain Registration Length, URL Redirection, Target Variable (Label).

3.3 Data Preprocessing

Preparatory steps were undertaken to undertake rigorous and thorough preprocessing of data to ensure the best possible performance. This contained treating missing values by imputation with mean or median, or mode replacement, normalization, and standardization. The next step involved feature scaling and one-hot encoding of categorical variables. Duplicate values, outliers, and noise were also removed, and transformations such as log transformation were applied. Finally, feature selection and extraction complete the dataset refinement process. Data quality, bias reduction, accuracy enhancement in the model, and consistency improve performance derived from the model and results produced and make the training process efficient.

3.4 Feature Extraction

The system is implemented using the Phishing Website Detection dataset from Kaggle, employing lexical and content-based feature extraction techniques for URLs. Various Python libraries, including pandas, scikit-learn, and urlparse for extracting these features. This process

included the extraction of 39 new features known to be predictive of phishing attempts. These features are characteristic of URLs, such as: length of the URL, domain information, suspicious keywords, etc.

3.5 Feature Selection

Phishing detection relied on the following techniques: correlation matrices and recursive feature elimination. The models were then refined by removing features of low variance and those suffering from high multicollinearity as these were found to complicate the models without improving performance.

3.5.1 Normalization

Normalization was applied during data preprocessing before training the machine learning models. Normalizing the data would imply that it would scale the features in a uniform manner, thereby benefiting model outcomes, particularly those models requiring feature scaling. For example, networks sensitive to feature scale, which are extensively utilized, are ANN and RNN. The following techniques were made use of:

A. Min-Max Scaling

The goal is to bring all features within a common range, typically between 0 and 1. The purpose of this method is to modify the values in each variable by mean of minus the smallest value and then divide by range (maximum - minimum).

B. Standardization

Standardization (Z-score normalization) is another method used to normalize the features so that they have an average of 0 and a standard deviation of 1. It is very useful to the algorithms having the assumption that standardized features are normal from a distribution point of view.

C. Handling Categorical Features

In other words, the categorical features are "HTTPS presence," "Suspicious Domain," and "IP Address," which have been changed to a numerical value of 0 as HTTP and 1 as HTTPS for real-time binary feature presentation.

D. Feature Engineering

Attributes have been created related to the URL, such as URL length, number of dots, and suspicious keys. These attributes were normalized using the min-max scale.

3.6 Dataset Splitting

The dataset was split into a training and test dataset using an 80-20 split in order to estimate model performance during the training. The training set consisted of 8,844 instances used to train the models while the testing set comprised 2,211 instances that were used for evaluation. This gave a perfect split for the performance metrics in terms of accuracy, precision, recall, and F1-score because they would be computed based on the test set.

3.7 Model Training

Four different machine learning algorithms, including ANN, RNN, XGboost, and Random Forest, were trained on this dataset from Kaggle for the detection of phishing websites. Through missing value handling, normalizing, and scaling features, the dataset was then pre-processed and reduced to 80 percent of the final training data and evaluated with a random sample of 20 percent using various evaluation metrics, such as accuracy, precision, recall, and F1-score. Comparing performance across the models are illustrated in Table 1 and 2 helps to identify which model is the best in terms of phishing detection algorithms because it indicates both training and testing accuracy for each model.

3.8 Model Selection

Model Selection for phishing website detection will be optimally carried out through Confusion Matrix analysis for different algorithms to lead to the calculation of accuracy, precision, recall, and f1 score. The predictions are classified into four main categories by Confusion Matrix: True Positives (TP), which indicate identified phishing websites; False Positives (FP), which are legitimate websites flagged as phishing; False Negatives (FN), which are phishing websites misclassified as legitimate; and True Negatives (TN), showing correct identification of non-phishing websites. These categories are the basis for calculating performance metrics, which are shown in the Figure 2.

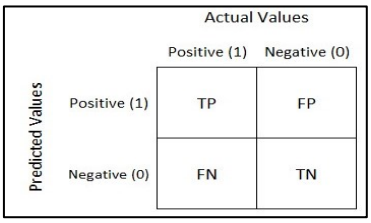

Figure 2. TP, FP, FN, TN Arrangement

Phishing site detection accuracy is defined as the ratio of correctly identified instances to the total number of accurately predicted instances. This metric relates to overall model performance in distinguishing between safe and malicious websites. This can be evaluated using the formula shown below.

$$Accuracy = (TP + TN) / (TP + TN + FP + FN)$$

Precision relates to the number of true positives predicted by the phishing website detection model and the total predictions made. It emphasizes the model's ability to reduce incidents of false positives, even when there is a case of a valid website being classified as phishing. Hence, a high precision score indicates that when such a website has been identified by the model as phishing, it is likely to be true positive; This can be evaluated using the formula shown below.

$$Precision = \frac{TP}{TP+FP}$$

Recall is a measure of how well the model can discover the proportion of true phishing websites. This only means how successful the model is in detecting false negatives, which means phishing sites are classified as legitimate. This can be evaluated using the formula shown below.

$$Recall = \frac{TP}{TP + FN}$$

The F1 Score is one of the most significant metrics for evaluating binary classification models, especially when classes are not balanced. It is obtained as the harmonic mean of precision ($TP / (TP + FP)$) and recall ($TP / (TP + FN)$). This metric, therefore, provides a balance in measuring performance. Therefore, the higher the F1 Score, the better the precision and recall of the system. Although keys have different meanings in terms of false positives and false negatives, they can be evaluated using the formula shown below

$$F1\ Score = 2 * \frac{Precision \times Recall}{Precision + Recall}$$

4. Results and Discussion

Table 1, Tab1e 2 and Figure 3 provides a summary of the evaluation metrics for the respective algorithms:

Table 1. Performance Metrics for Phishing Websites (Positive-1)

Model	Accuracy	Precision	Recall	F1 Score
ANN	96.92	0.98	0.95	0.96
RNN	96.74	0.98	0.95	0.96
XGBoost	97.06	0.98	0.95	0.97
Random Forest	96.66	0.97	0.95	0.96

Table 2. Performance Metrics for Phishing Websites (Negative-0)

Model	Accuracy	Precision	Recall	F1 Score
ANN	96.92	0.96	0.99	0.97
RNN	96.74	0.96	0.98	0.97
XGBoost	97.06	0.96	0.98	0.97
Random Forest	96.66	0.96	0.98	0.97

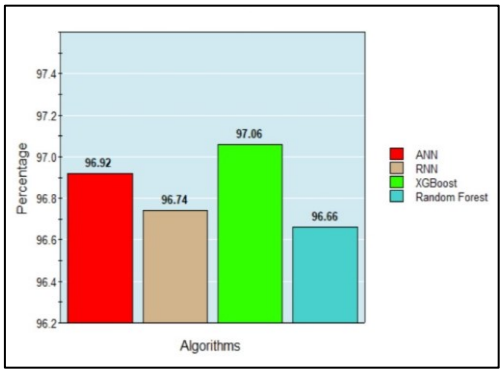

Figure 3. Performance Metrics Comparison

Google Collab was used for implementing the machine learning models and the methods used in evaluating the performance of the proposed framework. The four tested

algorithms showed that XGBoost performed the best, with 97.06% accuracy as shown in Figure 3. It has a precision of 0.98 and a considerably high recall of 0.95 for positive cases, while those for negative cases lead to a precision of 0.96 and, subsequently, a recall of about 0.98. The confusion matrix for XGBoost is shown in Figure 4.

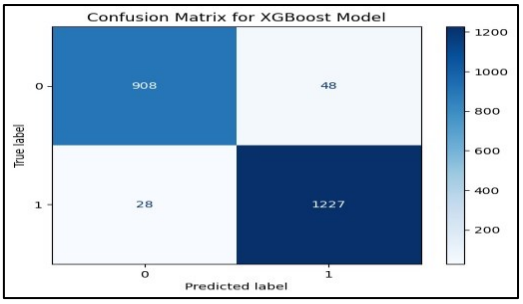

Figure 4. Confusion Matrix for XGBoost

This ANN achieved an accuracy of 96.92%. The positive cases had a precision of 0.96 and a recall of 0.99, while the negative cases had a precision of 0.98 and a recall of 0.95. The confusion matrix for ANN is shown in Figure 5.

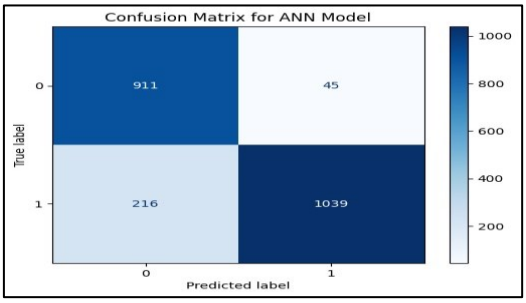

Figure 5. Confusion Matrix for ANN

The RNN obtained an accuracy of 96.74% with a precision of 0.96 and a recall of 0.98 for the positive cases, whereas the precision and recall scores obtained were 0.98 and 0.95 for negative cases, respectively. The confusion matrix for RNN is shown in Figure 6.

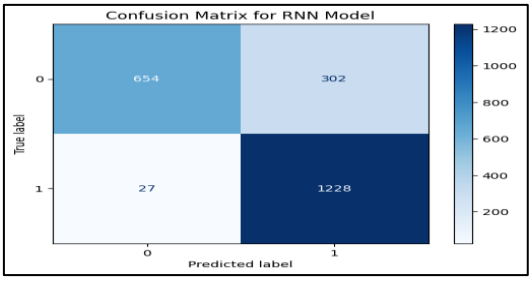

Figure 6. Confusion Matrix for RNN

The Random Forest achieved an overall accuracy of 96.66% along with a precision of 0.96 and recall of 0.98 for the positive class, while negative cases had precision of 0.97 and recall of 0.95. The confusion matrix for Random Forest is shown in Figure 7

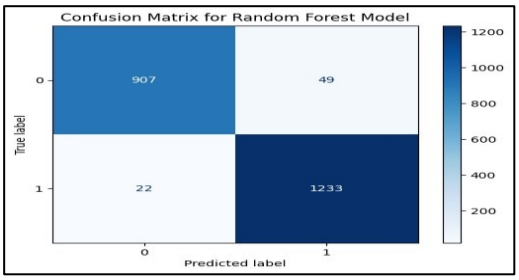

Figure 7. Confusion matrix for Random Forest

4.1 Model Execution

The following Figures, 8, 9, and 10, provide results for the model execution. The results appear accurate, and while the metrics indicate good performance, ensure the dataset is balanced during training and testing.

```
xgb_model = xgb.XGBClassifier(use_label_encoder=False, eval_metric='logloss')
xgb_model.fit(X_train, y_train)

y_pred_xgb = xgb_model.predict(X_test)
print("XGBoost Accuracy:", accuracy_score(y_test, y_pred_xgb))
print(classification_report(y_test, y_pred_xgb))
```

```
XGBoost Accuracy: 0.9706015377657169
              precision    recall  f1-score   support

           0       0.96      0.98      0.97      1255
           1       0.98      0.95      0.97       956

    accuracy                           0.97      2211
   macro avg       0.97      0.97      0.97      2211
weighted avg       0.97      0.97      0.97      2211

/usr/local/lib/python3.10/dist-packages/xgboost/core.py:158: UserWarning: [12:03:21] WARNING: /wor
Parameters: { "use_label_encoder" } are not used.

  warnings.warn(smsg, UserWarning)
```

Figure 8. Performance of XGBoost

```
+ Code  + Text

[45] Epoch 44/50
885/885 ━━━━━━━━━━━━━━━━━━━━ 3s 1ms/step - accuracy: 0.9856 - loss: 0.0336
Epoch 45/50
885/885 ━━━━━━━━━━━━━━━━━━━━ 3s 1ms/step - accuracy: 0.9852 - loss: 0.0329
Epoch 46/50
885/885 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.9838 - loss: 0.0388
Epoch 47/50
885/885 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.9869 - loss: 0.0356
Epoch 48/50
885/885 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.9858 - loss: 0.0358
Epoch 49/50
885/885 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - accuracy: 0.9857 - loss: 0.0322
Epoch 50/50
885/885 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - accuracy: 0.9837 - loss: 0.0358
70/70 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
ANN Accuracy: 0.9692446856625961
              precision    recall  f1-score   support

           0       0.96      0.99      0.97      1255
           1       0.98      0.95      0.96       956

    accuracy                           0.97      2211
   macro avg       0.97      0.97      0.97      2211
weighted avg       0.97      0.97      0.97      2211
```

Figure 9. Performance of ANN

```
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.9669832654987282
              precision    recall  f1-score   support

           0       0.96      0.98      0.97      1255
           1       0.97      0.95      0.96       956

    accuracy                           0.97      2211
   macro avg       0.97      0.97      0.97      2211
weighted avg       0.97      0.97      0.97      2211
```

Figure 10. Performance of Random Forest

5. Conclusion

This study on comparative analysis justifies the validity of machine learning algorithms in detecting phishing websites. The result reveals that the method produces an achieved accuracy of 97.06%, which is a greater accuracy than any other method, using Extreme Gradient Boosting (XGBoost). Artificial Neural Network (ANN), Recurrent Neural Network (RNN), and Random Forest all performed impressively, yielding accuracies of 96.92%, 96.74%, and 96.66%, respectively. These results will help build a stronger phishing detection system, increasing the online safety of sensitive information. The current research delves into the powerful potential of machine learning for phishing detection, focusing on innovative approaches. The future aspects involves developing browser extensions capable of real-time phishing site detection, which can alert users before they unknowingly share sensitive information. Another area of investigation explores multimodal phishing detection methods that combine text, image, and behavioral features to enhance accuracy and reliability in identifying phishing attempts.

References

[1] Ian Fette, Norman Sadeh and Anthony Tomasic, "Learning to detect phishing websites", Proceedings of the 16th international conference on World Wide Web, 2007. 649-656

[2] Ding, X.; Liu, B.; Jiang, Z.; Wang, Q.; Xin, L. “Spear Phishing Emails Detection Based on Machine Learning” in Proceedings of the 2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD), Dalian, China, 5–7 May 2021; 354–359.

[3] Gascon, Hugo, Steffen Ullrich, Benjamin Stritter, and Konrad Rieck. "Reading between the lines: content-agnostic detection of spear-phishing emails." In Research in Attacks, Intrusions, and Defenses: 21st International Symposium, RAID 2018, Heraklion, Crete, Greece, September 10-12, 2018, Proceedings 21, Springer International Publishing, 2018. 69-91.

[4] Chandrasekaran, Madhusudhanan, Krishnan Narayanan, and Shambhu Upadhyaya. "Phishing email detection based on structural properties." In NYS cyber security conference, vol. 3, 2006. 2-8.

[5] Ahmed, Abdulghani Ali, and Nurul Amirah Abdullah. "Real time detection of phishing websites." In 2016 IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), IEEE, 2016. 1-6.

[6] Rathod, Sunil B., and Tareek M. Pattewar. "Content based spam detection in email using Bayesian classifier." In 2015 International Conference on Communications and Signal Processing (ICCSP), IEEE, 2015. 1257-1261

[7] Daeef, Ammar Yahya, R. Badlishah Ahmad, Yasmin Yacob, Naimah Yaakob, Mohd Warip, And Mohd Nazri Bin. "Phishing Email Classifiers Evaluation: Email Body And Header Approach." Journal of Theoretical & Applied Information Technology 80, no. 2 (2015).

[8] Dewis, Molly, and Thiago Viana. "Phish responder: A hybrid machine learning approach to detect phishing and spam emails." Applied System Innovation 5, no. 4 (2022): 73.

[9] Dhanaraj, S., and V. Karthikeyani. "A study on e-mail image spam filtering techniques." In 2013 international conference on pattern recognition, informatics and mobile engineering, Salem. IEEE, 2013. 49-55

[10] M. Khonji, Y. Iraqi and A. Jones, "Phishing detection: A literature survey", IEEE Communications Surveys Tutorials, vol. 15, no. 4, 2013. 2091-2121

[11] Giri KJ, Parah SA, Bashir R, Muhammad K. 2021. “An efficient approach for phishing detection using machine learning” in Giri KJ, Parah SA, Bashir R, Muhammad K, eds. Multimedia security. Algorithms for intelligent systems. Singapore: Springer. 239-253.

[12] Sahingoz, Ozgur Koray, Ebubekir Buber, Onder Demir, and Banu Diri. "Machine learning based phishing detection from URLs." Expert Systems with Applications 117 (2019): 345-357.

[13] A.K. Jain and B.B. Gupta, "Towards detection of phishing websites on client-side using machine learning based approach", Telecommunication Systems, vol. 68, no. 4,2018. 687-700

[14] N. Abdelhamid, A. Ayesh and F. Thabtah, "Phishing detection based Associative Classification data mining", Expert Systems with Applications, vol. 41, no. 13, 2014.5948-5959

[15] Androutsopoulos, Ion, John Koutsias, Konstantinos V. Chandrinos, George Paliouras, and Constantine D. Spyropoulos. "An evaluation of naive bayesian anti-spam filtering." arXiv preprint cs/0006013 (2000).

[16] Wu, Chih-Hung. "Behavior-based spam detection using a hybrid method of rule-based techniques and neural networks." Expert systems with Applications 36, no. 3 (2009): 4321-4330.