# Comparative Analysis of Phishing Email Classifiers

Ashwin Venkatakrishnan
*ashvenk@iu.edu*

Prateek Giridhar
*pgiridha@iu.edu*

Prinston Rebello
*prebello@iu.edu*

project-ashvenk-pgiridha-prebello

**Abstract**

In an increasingly interconnected world, the proliferation of spam and phishing emails has presented a pressing challenge for individuals and organizations. The need for efficient email classification has led to numerous attempts to combat this issue using machine learning algorithms. This project endeavors to contribute to this ongoing effort by implementing novel methods acquired during the current semester's coursework. Our primary objective is to develop a robust email classification system capable of categorizing incoming emails as "safe" or "phishing." Leveraging the techniques and knowledge gained in our recent coursework, the project primarily focuses on comparing the performance of three classifiers - Random Forest, Support Vector, and XGBoost Classifiers - by experimenting with different features extracted using methods learned over this semester. By harnessing the power of data mining, we aim to present a classification model that can effectively discern the nature of incoming emails. This project is poised to make a meaningful contribution to email security in the modern digital landscape. The ability to automatically identify and separate safe and phishing emails can enhance cybersecurity and improve user experience. We envision that our experiment will uncover useful insights on classifying emails, deepen our understanding of data mining techniques, and contribute to the ongoing battle against email-based threats.

**Keywords:** email classification, phishing, exploratory data analysis, natural language processing, random forest, support vector classification, gradient boost

## 1 Introduction

In an age where digital communication is the lifeblood of modern society, the battle against email-based threats such as spam, phishing, and their ilk rages on. While email providers like Outlook and Gmail employ algorithms to filter unwanted messages, false positives remain an enduring challenge. Consider, for instance, legitimate emails from educational institutions with essential links, unjustly relegated to the spam folder. The core objective of our project is to advance the state of email classification by meticulously categorizing incoming emails as "safe" or "phishing" while minimizing false positives. We aim to achieve this by extracting language features and computing the similarity between safe and unsafe emails. Our possible future work is to study the causes of a target falling for a phishing attack through an empirical study. Our present approach includes Exploratory Data Analysis (EDA) for pre-processing, computing the similarity score, and extracting language features using Natural Language Processing (NLP) techniques. Using these features, we will train and test three classifiers - Random Forest, Support Vector, and XGBoost - and present our findings through four experiments to highlight the strengths and weaknesses of each approach.

### 1.1 Previous work

For a comprehensive understanding, consider consulting journal papers Salloum et al. [2022], Karim et al. [2019], Park and Taylor [2015], Verma et al. [2020], Kleitman et al. [2018], Graham [2002], and the findings in the conference paper by Khan et al. [2020], Yearwood et al. [2010].

## 2 Methods

Our project relies on the Phishing Emails Dataset available on Kaggle. This dataset provides essential information - the email bodies and labels classifying emails as safe or unsafe. Our focus is primarily on the subset of data marked as unsafe.

### 2.1 Exploratory Data Analysis (EDA)

We begin by performing EDA of the email data to identify any correlations or similarities between unsafe and safe emails. This process plays a pivotal role in reducing false positives in email classification. During this analysis, the data is scrutinized for patterns and relationships. Additionally, we meticulously inspect the dataset for attribute violations to understand the underlying cause of misclassifications.

In our dataset, we observed that for 533 records, the "Email Text" attribute was set to the string "empty". Additionally, there were also 16 N/A or NULL records. As such, we cleaned the data set by replacing "empty" with N/A for the 533 records and then dropping all N/A records, which resulted in 549 records being dropped in total. With this step completed, the shape of the dataset reduced from (18650, 3) to (18101, 3).

Before we proceed to train the classification models, we perform text extraction and analysis using NLP techniques to gain meaningful insights from the email content.

### 2.2 Overview of Classification Experiments

For the analysis we aim to carry out in this paper, we employ Random Forest, Support Vector, and XGBoost classifiers to label emails as "Safe" or "Phishing". Our objective is to conduct four experiments:

1. perform direct classification using these algorithms,
2. compute the Similarity score of emails and perform classification using the email body and similarity score as features,
3. employ NLP techniques to extract language features from email body and use it to perform classification,
4. combine the Similarity score and language features along with the email body to perform classification.

The result of these four experiments will allow for an assessment of their respective accuracy and shed light on the effectiveness of each approach.

### 2.3 Experiment 1: Direct Classification

The purpose of this experiment is to establish a baseline for the performance of each classification algorithm when supplied with email data using minimal pre-processing steps. Following EDA, we split the data set into train (67%) and test sets (33%). Additionally, we vectorize the text using Term Frequency and Inverse Document Frequency (TF-IDF) vectorizer. This converts the email text into numerical values that can be fed to the three classifiers which are configured as follows:

1. Random Forest Classifier - uses 10 estimators
2. Support Vector Classifier - uses a regularization value of 100
3. XGBoost Classifier - uses 100 estimators and a learning rate of 0.1

Specifically for the XGBoost classifier, the labels of our cleaned data are encoded using Label Encoder to be better fit by the classification model, and the data itself is split into 5 folds using KFold with shuffling enabled. The performance of these classifiers, which will be discussed in the next section, is presented in the form of a classification report and a confusion matrix.

## 2.4 Experiment 2: Classification using Similarity

In this experiment, a salient feature, namely the similarity score, was derived for each email with a phishing email. The underlying hypothesis posits that a higher degree of similarity to a phishing email correlates with an increased likelihood of the email being classified as such. After this theoretical framework, a comprehensive preprocessing phase was executed, wherein extraneous symbols were eliminated, leaving only words and letters. The resultant processed data was then exported to a new column. Following this, the TF-IDF vectorizer was applied to convert the email text data into numerical data. Subsequently, a cosine similarity metric was employed to assess the degree of similarity of each email to a phishing email. The computed similarity scores were stored in a newly created column, thereby rendering the dataset classification ready. The classification model preparatory steps included the removal of the index and email type columns from the X-axis in both the training and testing datasets, and utilizing only the email type column for the Y-axis. Subsequently, three distinct classification models, RandomForest, Support Vector Classifier (SVC), and XGBoost (five-fold cross-validation strategy), were deployed.

## 2.5 Experiment 3: Classification using Language Features

In this experiment, our objective was to augment the predictive capabilities of our model through the extraction of additional features from email bodies and the implementation of diverse classification algorithms. The process involved two main phases: feature extraction and classification.

**Feature Extraction:** To delve deeper into the characteristics of spam emails, we initiated a feature extraction process. First, we utilized the Language Tool API to identify lexical errors within the email bodies. Subsequently, an aggregate score was computed for each email based on the total count of lexical errors, which was then appended to the dataset. Moving forward, we extracted URLs from the email bodies and employed Google's Safe Browsing APIs to discern the safety of these links. The outcomes were integrated into the dataset, revealing the presence of approximately 10 emails with suspicious links. Furthermore, sentiment and emotion analyses were conducted. Leveraging the Vader sentiment analysis tool, we computed compound scores to categorize emails into positive and negative sentiments. Simultaneously, we utilized the NRCLex package to calculate emotion scores, categorizing emails as either positive or negative. These sentiment and emotion scores were added to the dataset, enriching it with valuable information.

**Classification Models:** Our choice of classification models remained consistent with Experiments 1 and 2, encompassing Random Forest, Support Vector Classifier (SVC), and XGBoost. Additionally, we introduced an Artificial Neural Network (ANN) for comparison. To facilitate the processing of text-based information by numerical models, we transformed the email bodies into numerical format using Term Frequency-Inverse Document Frequency (TFIDF). Labeled features such as URL verification, emotion, and sentiment categories were transformed using LabelEncoder. With consistent parameters across all models, as described in Experiment 1, we conducted both training and testing on the dataset. Notably, the inclusion of the extracted features resulted in substantial improvements in test accuracies. Despite achieving commendable training accuracies, the ANN exhibited signs of overfitting, leading to lower-than-expected accuracies on the test data. This observation underscores the importance of feature engineering in refining model performance and provides valuable insights into the limitations of specific models.

### 2.6 Experiment 4: Classification using Similarity and Language Features

In the course of this final experiment, all features derived from prior experiments were aggregated to construct a novel DataFrame. With this consolidated data set now poised for classification, both the Index and Email Type columns were excluded from the X-axis in both the training and test data sets like the previous experiments. Simultaneously, the Email Type column was retained as the dependent variable on the Y-axis. After the preparatory data manipulation, three predetermined machine learning models - RandomForest, Support Vector Classifier (SVC), and XGBoost (five-fold cross-validation strategy) - were executed.

## 3 Results

The following table is a summary of the accuracy obtained for each classifier through our experiments:

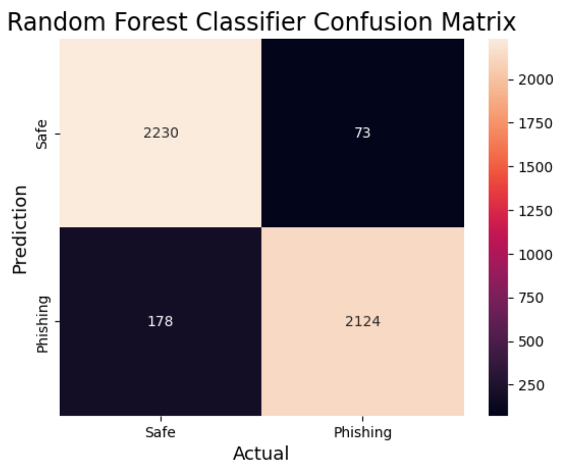
Table 1: Results for Different Approaches.

| **Approach** | **Random Forest** | **SVC** | **XGBoost** |
|---|---|---|---|
| Direct | 94.50% | 49.90% | 94.80% |
| Similarity | 94.30% | 63.30% | 93.14% |
| NLP | 94.58% | 95.77% | 95.60% |
| NLP + Similarity | 95.46% | 93.80% | 94% |

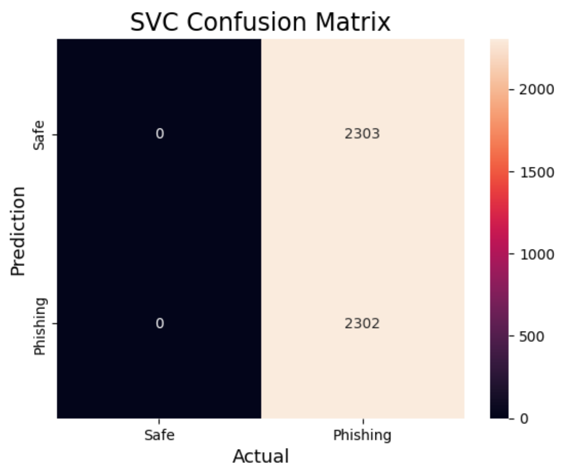
### 3.1 Experiment 1

The following is the confusion matrix-based visualization of the result for the 3 classifiers.

Figure 1: Experiment 1 Result

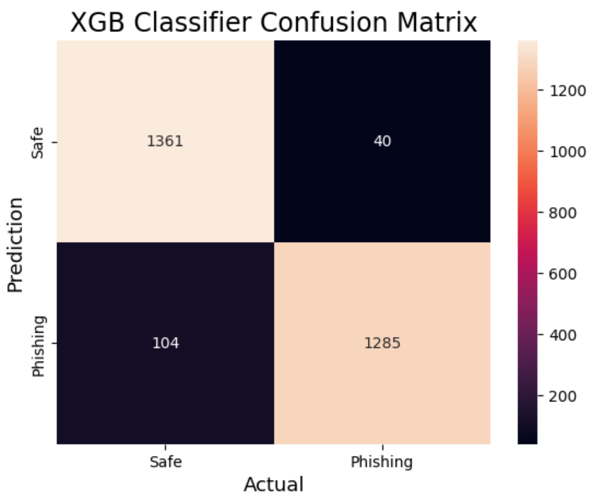


(a) Confusion Matrix for Random Forest.



(b) Confusion Matrix for SVC.

Figure 2: Experiment 1 Result

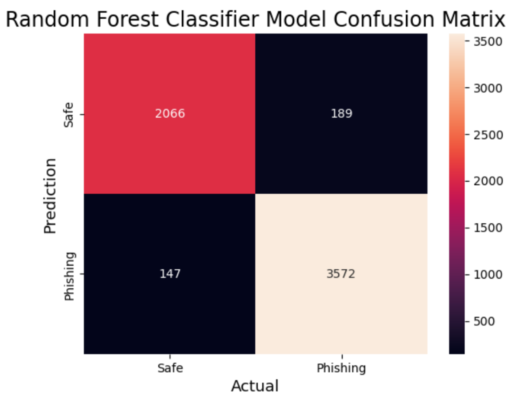


(a) Confusion Matrix for XGBoost.

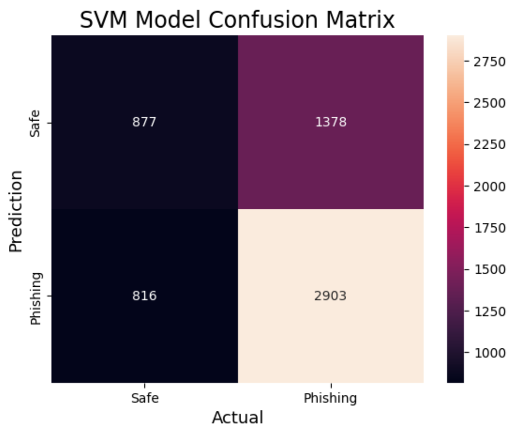
## 3.2 Experiment 2

The following is the confusion matrix-based visualization of the result for the 3 models.

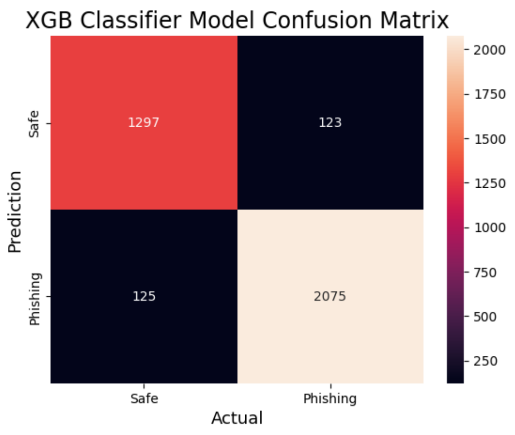
Figure 3: Experiment 2 Result



(a) Confusion Matrix for Random Forest.



(b) Confusion Matrix for SVC.

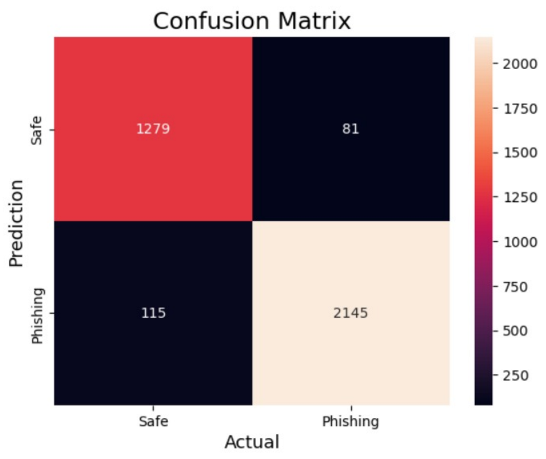


(c) Confusion Matrix for XGBoost.

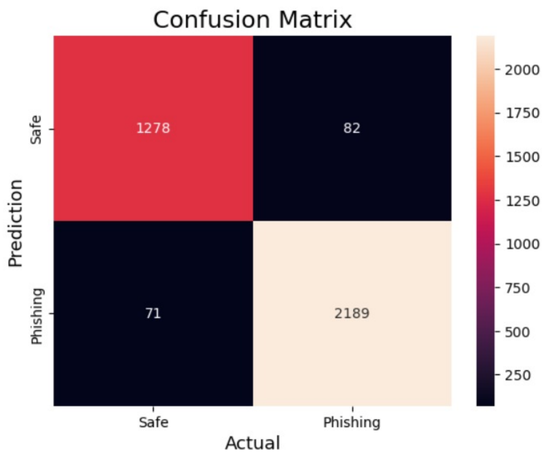
## 3.3 Experiment 3

The following is the classification report and confusion matrix-based visualization of the result for the 4 models.

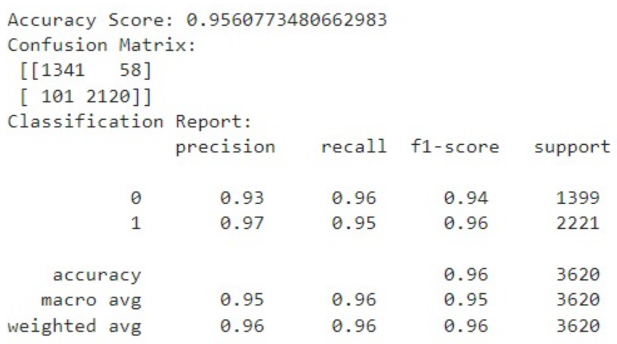
Figure 4: Experiment 3 Result



(a) Confusion Matrix for Random Forest.

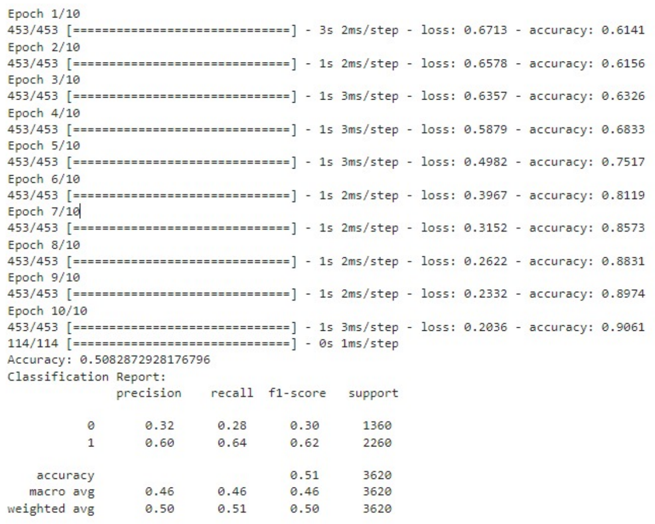


(b) Confusion Matrix for SVC.



```
Accuracy Score: 0.9560773480662983
Confusion Matrix:
 [[1341   58]
 [ 101 2120]]
Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.96      0.94      1399
           1       0.97      0.95      0.96      2221

    accuracy                           0.96      3620
   macro avg       0.95      0.96      0.95      3620
weighted avg       0.96      0.96      0.96      3620
```

(c) Classification Report for XGBoost.



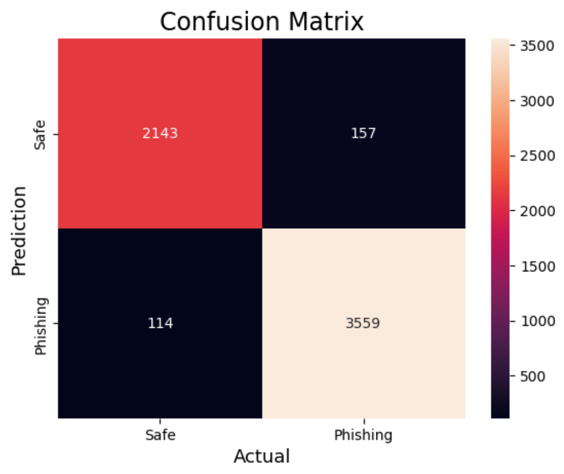
```
Epoch 1/10
453/453 [==============================] - 3s 2ms/step - loss: 0.6713 - accuracy: 0.6141
Epoch 2/10
453/453 [==============================] - 1s 2ms/step - loss: 0.6578 - accuracy: 0.6156
Epoch 3/10
453/453 [==============================] - 1s 3ms/step - loss: 0.6357 - accuracy: 0.6326
Epoch 4/10
453/453 [==============================] - 1s 3ms/step - loss: 0.5879 - accuracy: 0.6833
Epoch 5/10
453/453 [==============================] - 1s 3ms/step - loss: 0.4982 - accuracy: 0.7517
Epoch 6/10
453/453 [==============================] - 1s 2ms/step - loss: 0.3967 - accuracy: 0.8119
Epoch 7/10
453/453 [==============================] - 1s 2ms/step - loss: 0.3152 - accuracy: 0.8573
Epoch 8/10
453/453 [==============================] - 1s 2ms/step - loss: 0.2622 - accuracy: 0.8831
Epoch 9/10
453/453 [==============================] - 1s 2ms/step - loss: 0.2332 - accuracy: 0.8974
Epoch 10/10
453/453 [==============================] - 1s 3ms/step - loss: 0.2036 - accuracy: 0.9061
114/114 [==============================] - 0s 1ms/step
Accuracy: 0.5082872928176796
Classification Report:
              precision    recall  f1-score   support

           0       0.32      0.28      0.30      1360
           1       0.60      0.64      0.62      2260

    accuracy                           0.51      3620
   macro avg       0.46      0.46      0.46      3620
weighted avg       0.50      0.51      0.50      3620
```

(d) Classification Report for ANN.

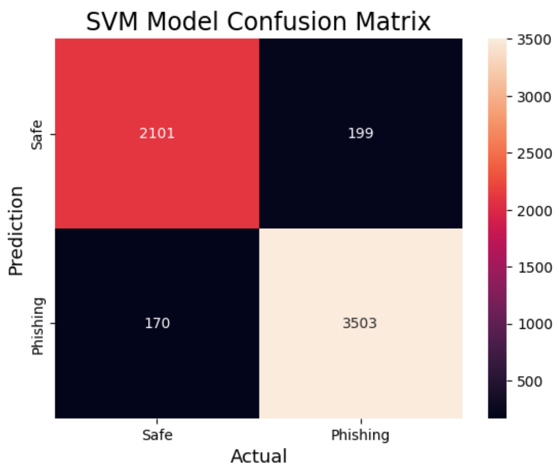
## 3.4 Experiment 4

Below is the confusion matrix-based visualization of the result for all the 3 models.

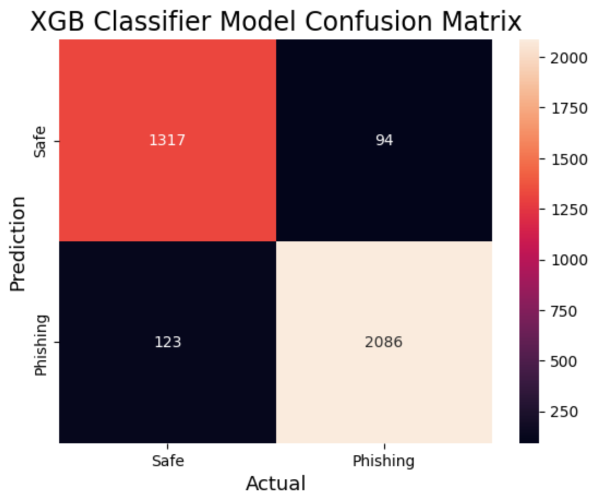
Figure 5: Experiment 4 Result



(a) Confusion Matrix for Random Forest.



(b) Confusion Matrix for SVC.



(c) Confusion Matrix for XGBoost.

## 4 Discussion

Table 1 summarizes the accuracy achieved by the three classifiers for each experiment that we conducted. Overall, we achieved the highest accuracy with the Random Forest model (95.46%) in Experiment 4, and the SVC (95.77%) and XGBoost (95.6%) models in Experiment 3.

We see a significant improvement in the performance of SVC in each of experiments 2, 3, and 4, where we have computed similarity scores and performed feature extraction using NLP. Specifically, the results of Experiment 3 prove our hypothesis that adding features like URL verification, lexical errors, sentiment, and emotion to the data set helps the classifier better fit the data and classify unseen emails as "Safe" or "Phishing". With Experiment 4, we hypothesized that combining similarity score as a feature to the language features extracted in Experiment 3 could further improve the classifiers' performance. This is proved true, with each model showing a meaningful improvement in accuracy over the baseline model. However, the accuracy achieved in Experiment 3 is promising and aligns with a larger goal - if we can understand what parts of an email cause people to fall for a phishing attack, in terms of the language, appearance, and user behavior, we can figure out ways to extract such characteristics as features using NLP techniques to help inform email users of potential attacks.

## 5 Author Contribution Statement

The project started with each of us conducting a literature survey on phishing email classification and together, we discussed our findings before settling on our objective and writing the project proposal. During implementation, we performed EDA and split the work as follows:

1. Ashwin performed data pre-processing and conducted Experiment 1 to establish the baseline performance metrics for the three classifiers.
2. Prateek performed data pre-processing, extracted similarity as a feature, and conducted Experiments 2 and 4.
3. Prinston performed data pre-processing, used NLP techniques to extract language features, and conducted Experiment 3.

Finally, we pitched ideas to record the presentation and completed this report by having each person speak and write about their respective experiment.

## References

Said Salloum, Tarek Gaber, Sunil Vadera, and Khaled Shaalan. A systematic literature review on phishing email detection using natural language processing techniques. *IEEE Access*, 10: 65703–65727, 2022.

Asif Karim, Sami Azam, Bharanidharan Shanmugam, Krishnan Kannoorpatti, and Mamoun Alazab. A comprehensive survey for intelligent spam email detection. *IEEE Access*, 7:168261–168295, 2019.

Gilchan Park and Julia M Taylor. Using syntactic features for phishing detection. *arXiv preprint arXiv:1506.00037*, 2015.

Priyanka Verma, Anjali Goyal, and Yogita Gigras. Email phishing: Text classification using natural language processing. *Computer Science and Information Technologies*, 1(1):1–12, 2020.

Sabina Kleitman, Marvin KH Law, and Judy Kay. It's the deceiver and the receiver: Individual differences in phishing susceptibility and false positives with item profiling. *PloS one*, 13(10): e0205089, 2018.

Paul Graham. A plan for spam. `http://www.paulgraham.com/spam.html`, 2002.

Sohail Ahmed Khan, Wasiq Khan, and Abir Hussain. Phishing attacks and websites classification using machine learning and multiple datasets (a comparative analysis). In *Intelligent Computing Methodologies: 16th International Conference, ICIC 2020, Bari, Italy, October 2–5, 2020, Proceedings, Part III 16*, pages 301–313. Springer, 2020.

John Yearwood, Musa Mammadov, and Arunava Banerjee. Profiling phishing emails based on hyperlink information. In *2010 International Conference on Advances in Social Networks Analysis and Mining*, pages 120–127. IEEE, 2010.