Phishy?
Detecting Phishing Emails Using Machine Learning and Natural Language Processing

Md Fazle Rabbi, Arifa I. Champa, and Minhaz F. Zibran

Abstract Phishing emails, a type of cyberattack using fake emails, are difficult to recognize due to sophisticated techniques employed by attackers. In this paper, we use a natural language processing (NLP) and machine learning (ML) based approach for detecting phishing emails. We compare the efficacy of six different ML algorithms for the purpose. An empirical evaluation on two public datasets demonstrates that our approach detects phishing emails with high accuracy, precision, and recall. The findings from this work are useful in devising more efficient techniques for recognizing and preventing phishing attacks.

Keywords: Phishing, spam, email, classification, detection, machine learning, natural language processing

1 Introduction

Phishing is a common approach that cybercriminals use to steal sensitive information from individuals and organizations. Messaging apps, phone calls, social media, and emails are just a few of the many platforms that phishing attacks target. The outbreak of the coronavirus epidemic resulted in an increase in phishing attacks that was 220% higher than the yearly average [6]. According to the Anti-Phishing Working Group, there were a record-breaking total of 1,270,883 unique phishing

Md Fazle Rabbi
Idaho State University, Pocatello, ID, USA, e-mail: mdfazlerabbi@isu.edu

Arifa I. Champa
Idaho State University, Pocatello, ID, USA, e-mail: arifaislamchampa@isu.edu

Minhaz F. Zibran
Idaho State University, Pocatello, ID, USA, e-mail: minhazzibran@isu.edu

incidents reported in the third quarter of 2022, demonstrating the prevalence and persistence of this threat [2].

Email has long been arguably the most used platform for electronic communication in both formal and informal settings. Therefore, email platforms have been the most frequent targets for phishing attacks where cybercriminals create scam emails that appear genuine but contain traps for the end-user for becoming a victim of information leaks or other malicious attacks.

According to Forrester Research, 91% of all hacking attacks begin with a phishing email [23]. In 2022, 96% of organizations were targeted by email-related phishing attempts [24]. Phishing attacks are becoming increasingly sophisticated and can deceive even experienced users, as illustrated by the case of the John Podesta phishing attack [34]. This incident, involving a fake email that tricked the former chairman of the Hillary Clinton presidential campaign, demonstrated that anyone is susceptible to an email phishing attack.

Phishing often takes advantage of the naivety of end-users, and combating it requires human intervention in the form of awareness and training campaigns [7, 9, 25]. Despite the ongoing efforts in employee training and social awareness of email phishing attacks, these attacks continue to increase, resulting in an increasing number of email phishing victims. This indicates that we need more sophisticated approaches for the automatic detection of phishing emails before those emails are presented to the end-users, who may or may not be technically well-educated.

To develop an effective approach for detecting phishing emails, it is crucial to understand the characteristics of phishing emails that distinguish them from legitimate ones. To accomplish this, we examine the characteristics and traits of phishing emails to identify patterns that can be used to develop an effective detection model. In particular, we address the following research questions:

RQ1: *What is the most effective machine learning (ML) algorithm for detecting phishing emails?*
— Since different ML algorithms have their strengths and weaknesses in different application contexts, it is critical to identify the most effective algorithm for detecting phishing emails. We measure effectiveness in terms of accuracy, precision, recall, and F-score.

RQ2: *What impact does ML algorithm training time have on the performance of phishing email detection?*
— Lengthy training times may not be feasible in some operational settings, and if longer training times do not produce significant improvements in the performance of the model, the additional time and resources invested would just be wasted. Hence, a deep understanding of the relationship between training time and performance can assist in the development of more efficient phishing detection systems for adoption in practical settings.

RQ3: *To what extent does the 'subject' feature of an email contribute to the accurate detection of phishing emails?*
— One of an email's most noticeable and prominent features is the 'subject', and the content of that line has a significant impact on how the recipient perceives the

email's authenticity. Investigating the 'subject' feature's contribution to accurately detecting phishing emails can provide insights into the creation and identification of phishing emails, which will aid in the development of more effective phishing detection systems.

To address the aforementioned research questions, we use two phishing email datasets, which we process using natural language processing (NLP) and then we compare six machine learning algorithms by separately operating them on the two datasets.

The rest of the paper is structured as follows. Section 2 describes the data sets used in this work. Section 3 describes the methodology of this study. The findings from this study are presented and discussed in Section 4. In Section 5, we describe the limitations of our work. Previous work in the literature that are related to ours are discussed in Section 6. Finally, Section 7 concludes the paper.

2 Datasets

For our work, we use two publicly available datasets. Table 1 presents the number of phishing and legitimate emails in each of the datasets, which are further described below.

Table 1 Datasets used in our work

	Dataset	
Email	**Ling** [31]	**TREC** [10]
Legitimate	2,412	25,220
Phishing	481	50,199
Total	2,893	75,419

2.1 Ling Dataset

The Ling Spam dataset [31] is a collection of emails specifically focused on topics of interest to linguists. The dataset is organized around two main attributes: 'subject' and 'message,' as well as a 'label' indicating whether each email is spam or legitimate. The subject and body of an email are captured in 'subject' and 'message' respectively.

2.2 TREC Dataset

The TREC Public Corpus [10] is a collection of email messages collected between April 8 and July 6, 2007. For our study, a preprocessed TREC Public Corpus Dataset is collected from Kaggle [5]. This dataset includes four attributes: ‘subject,’ ‘email_to,’ ‘email_from,’ and ‘message,’ as well as a ‘label’ indicating whether the email is phishing or legitimate. The ‘email_to’ and ‘email_from’ attributes respectively capture the email addresses of the recipients and sender of the email.

3 Methodology

We use a three-phase procedure for recognizing phishing emails. First, we sanitize the dataset through a preprocessing phase. Second, subsets of the preprocessed data are then separately fed to six ML algorithms [11] to create ML models. Finally, the ML models are operated on separate subsets of the preprocessed datasets. The models’ performances are measured using a set of metrics described in Section 3.3.

To avoid the overhead of installing and configuring different tools and libraries on a local machine for operating the ML algorithms, we leverage a cloud-based platform, Google Colab [3], for the purpose. However, we use the processing power of our local machine rather than Colab’s GPU or TPU. Our local machine, which runs a Windows 10 operating system, is equipped with an Intel Core i7-8650U processor with a base clock speed of 1.90 GHz and turbo boost up to 4.2 GHz, and 16GB of DDR4 memory.

3.1 Data Preprocessing

In the preprocessing phase, we handle missing attribute values, eliminate duplicates/redundancy, clean data, and convert them to a format for feeding to the ML algorithms.

3.1.1 Handling Missing Attribute Values

In the email datasets, we notice that some emails do not include a ‘subject’ but have a message body while some emails include a ‘subject’ with an empty message body. In cases where the ‘subject’ attribute is missing for an email, we exclude the attribute from consideration and only use the ‘message’ body for that particular email. If the ‘message’ attribute is missing for a particular email, we exclude that entire email from our study. Table 2 presents the number of instances we encountered the missing attribute values in the two datasets used in our work.

Table 2 Missing attribute values

	Dataset	
Attribute	**Ling** [31]	**TREC** [10]
subject	62	793
message	0	1487

3.1.2 Removing Duplicates

In the TREC dataset, the ‘message’ attribute contains 14,885 instances of duplicate data, and when we combine the ‘subject’ and ‘message’ attributes, we identify 11,107 instances of duplicate emails. In the Ling dataset, the ‘message’ feature contains 34 instances of duplicate data, and when we merge the ‘subject’ and ‘message’ features, we discover 17 instances of duplicate emails. We eliminate the duplicates keeping only one instance from each duplicate set.

3.1.3 Data Cleansing

To prepare data for ML algorithms, we perform further cleanup operations as listed below:

- We convert all text to lowercase, to ensure consistency.
- We remove any leading and trailing white spaces from each email body.
- We replace actual email addresses, URLs, currency symbols, and contact numbers with the placeholders ‘MAILID’, ‘LINKS’, ‘MONEY’, and ‘contact number’, respectively.
- Any non-alphanumeric character found in the email’s subject or body/message is replaced with a white space.
- We remove stop words (e.g., ‘the’, ‘is’) from both the subject and body of each email to reduce noise.

3.1.4 Vectorization

To convert email information into numerical vectors, we use the term frequency-inverse document frequency (TF-IDF). This well-known NLP technique multiplies a word’s term frequency (TF) by its inverse document frequency (IDF). The TF refers to how many times a word appears in a document, whereas the IDF refers to how frequently a word appears across the entire corpus of documents. Table 3 presents the total number of phishing and legitimate email instances we obtain in each of the datasets after the completion of the preprocessing step.

Table 3 Datasets after preprocessing

	Ling [31]	**TREC** [10]
Legitimate	2408	24673
Phishing	468	38152
Total	2876	62825

3.2 ML Algorithms for Classification

For the detection of phishing emails, we separately attempt with six ML algorithms [11] as briefly introduced below.

Logistic Regression (LR): LR uses a linear combination to combine the input features before running them through a sigmoid function to create a probability. A binary prediction can be made by thresholding this probability. To reduce the binary cross-entropy loss, the model learns its parameters from training data using gradient descent.

K-Nearest Neighbors (KNN): KNN identifies related samples in the training dataset and applies a class label to the input sample. To determine the majority class label, the algorithm calculates the distance to the K nearest neighbors from the input data. KNN is simple to use and effective at managing high-dimensional and noisy data.

AdaBoost (AB): AdaBoost combines a number of weak classifiers to create a strong one. Weak classifiers are learned using weighted examples after each example has been given a weight during training. To create the final strong classifier, these weak classifiers are then combined.

Multinomial Naive Bayes (MNB): MNB uses the Bayes theorem to calculate the likelihood that a new email might fall into each class depending on its attributes. It determines key features, such as important words or phrases, and then calculates the likelihood that an email falls into each class.

Gradient Boosting (GB): GB integrates the predictions from various models to enhance performance. Using a gradient descent technique to the residual errors, the algorithm trains distinct models that correct the faults generated by the prior model. The weighted average of all the predictions from each model is the final prediction.

Random Forest (RF): RF combines the predictions of multiple decision trees to produce a more accurate final prediction. To avoid overfitting and enhance model performance, each decision tree is trained using a random subset of the data. The algorithm can find the most crucial factors in the data by concentrating on the most important characteristics of a phishing email and disregarding unimportant ones.

Further details of all these well-known ML algorithms can be found elsewhere [11]. The attributes in the datasets, as described in Section 2, are used as features while operating the ML algorithms on the datasets.

Table 4 Confusion matrix used in our work

	Prediction/Classification	
Actual	**Legitimate**	**Phishing**
Legitimate	TN	FP
Phishing	FN	TP

3.3 Evaluation Metrics

For each ML algorithm operated on each dataset, we record the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). We then measure the performance of each algorithm in terms of accuracy, recall, precision, and F-score, which are characterized according to the confusion matrix shown in Table 4. We then measure the performance of each algorithm in terms of accuracy, recall, precision, and F-score as defined as follows.

• **Accuracy** is measured by the percentage of correctly categorized instances in the dataset. Mathematically

$$\text{accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{1}$$

• **Precision** measures the number of actual positive cases among all those that are predicted to be positive. Mathematically,

$$\text{precision} = \frac{TP}{TP+FP} \tag{2}$$

• **Recall** calculates the proportion of true positive cases among all positive examples. Mathematically defined as,

$$\text{recall} = \frac{TP}{TP+FN} \tag{3}$$

• **F-score** is the harmonic mean of precision and recall. Mathematically defined as follows:

$$\text{F-score} = 2 \times \frac{p \times r}{p+r} \tag{4}$$

True positive rate (TPR), also known as sensitivity, indicates how accurately phishing email detection systems identify true positive instances. Another essential metric is the false positive rate (FPR), also known as specificity, which represents the percentage of true negative cases that a detection system incorrectly interprets as positive.

For a typical detection system, when TPR increases, FPR also goes high. But high TPR and low FPR are desirable. Thus, by plotting TPR versus FPR at various classification levels, the receiver operating characteristic (ROC) curve is used to assess the overall effectiveness of a phishing email detection system. These evaluation indicators assist pinpoint problem areas and offer insights into how well a classification model is performing.

3.4 Applying the ML Algorithms

We operate ML algorithms on the two datasets, Ling and TREC, using only their common ‘subject’ and ‘message’ features, and excluding TREC’s additional ‘email_to’ and ‘email_from’ features.

From the Ling dataset, we first randomly pick 80% of phishing emails and 80% of legitimate emails and combine them to create our training subset of the Ling dataset. The rest 20% phishing and legitimate emails are combined to create the testing subset of the Ling dataset. Using the same procedure, we also create separate training and testing subsets of the TREC dataset. The training subsets are used to train the ML algorithms and the testing subsets are used to evaluate the algorithms’ performances.

4 Analysis and Findings

We now describe our analyses for addressing each of the research questions and derive their answers.

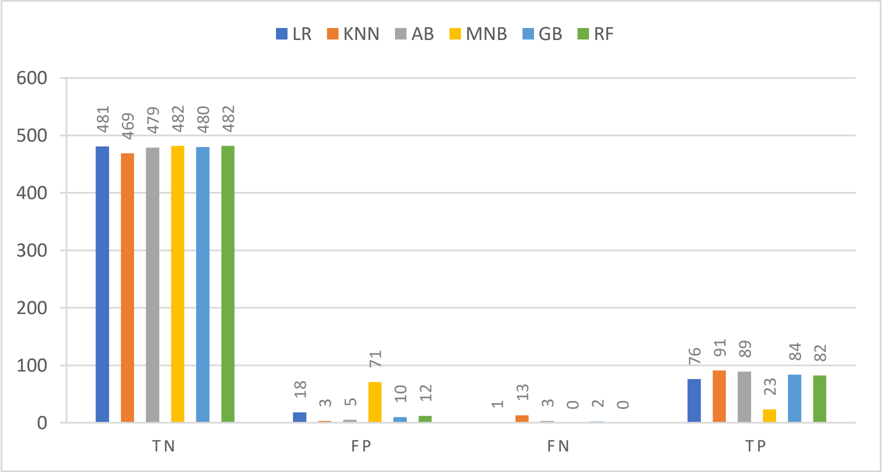

Fig. 1 TN, FP, FN, and TP observed for the six ML algorithms operated on the *Ling* dataset

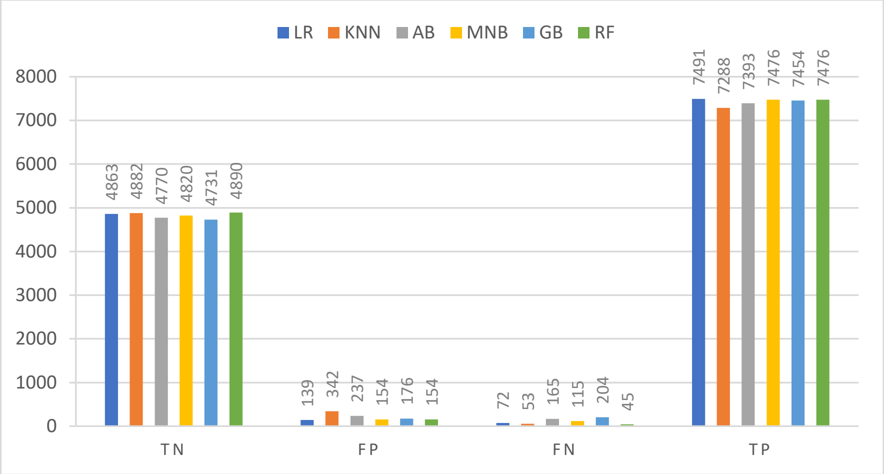

Fig. 2 TN, FP, FN, and TP observed for the six ML algorithms operated on the *TREC* dataset

4.1 Performance of ML Algorithms (RQ1)

Figure 1 and Figure 2 present the number of TP, TN, FP, and FN in the ML algorithms' detection of phishing emails in the Ling and TREC datasets respectively. In the current context, it is important to minimize FN predictions, which happen when the algorithm mistakenly classifies a phishing email as a legitimate one. With respect to the number of FNs, the RF algorithm appears to have outperformed others with zero FN in the Ling dataset and only 45 in the TREC dataset.

Table 5 Performance of the six ML algorithms when operated on the *Ling* dataset

Algorithms	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**ROC**	**F-score (%)**
LR	96.70	97.26	96.70	0.9755	96.84
KNN	97.22	97.21	97.22	0.9343	97.17
AB	98.61	98.63	98.61	0.9785	98.62
MNB	87.67	96.98	87.67	0.9358	90.99
GB	97.92	98.06	97.92	0.9782	97.95
RF	97.92	98.18	97.92	0.9879	97.97

The performance ML algorithms on the Ling dataset are presented in Table 5. As seen in the table, the RF algorithm achieves the highest ROC score (0.9879) and second highest F-score (97.97%), which is slightly (0.65%) lower than that of Adaboost. Adaboost is found to have achieved slightly higher accuracy, precision, recall, and F-score compared to those of RF but the differences always remain less than 0.7% only.

Table 6 Performance of the six ML algorithms when operated on the *TREC* dataset

Algorithms	Accuracy (%)	Precision (%)	Recall (%)	ROC	F-score (%)
LR	98.32	98.32	98.32	0.9813	98.32
KNN	96.86	96.93	96.86	0.9637	96.84
AB	96.80	96.80	96.80	0.9654	96.80
MNB	97.86	97.86	97.86	0.9769	97.86
GB	96.98	96.98	96.98	0.9687	96.98
RF	98.42	98.43	98.42	0.9817	98.41

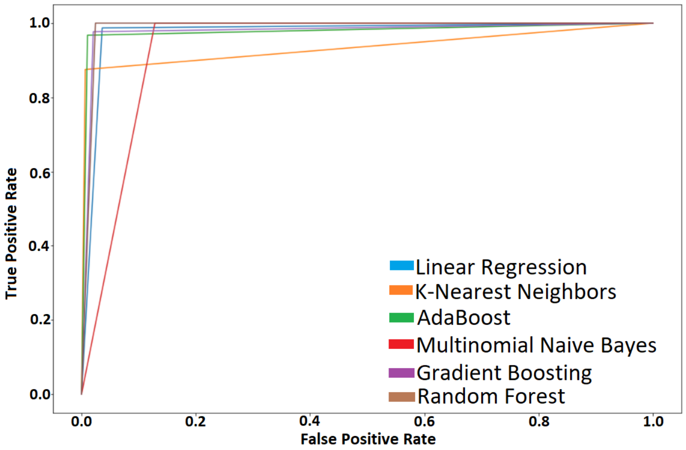

Fig. 3 ROC Curve for the six ML algorithms when operated on the *Ling* Dataset

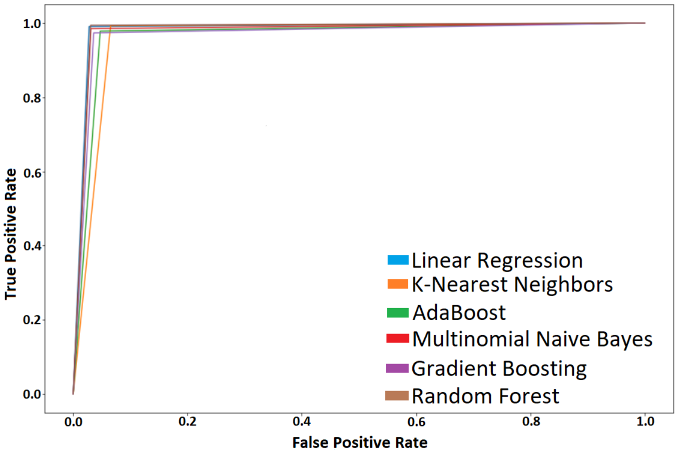

Fig. 4 ROC Curve for the six ML algorithms when operated on the *TREC* Dataset

The classification performance of the ML algorithms on the TREC dataset is presented in Table 6. As seen in the table, the RF algorithm clearly outperforms the other five algorithms achieving the highest scores in all the evaluation metrics. This is possibly because RF is effective in identifying phishing emails in large datasets such as TREC as it can improve the model's generalization performance and reduce overfitting by incorporating the predictions of multiple decision trees.

Figure 3 and Figure 4 present the ROC curves for the ML algorithms operated on the Ling and TREC datasets respectively. The closer an ROC curve reaches to the upper-left corner, the better the performance of the corresponding classifier. On both datasets, RF performs the best, with its ROC curve being the closest to the upper left corner.

Based on the observations discussed above, we now derive the answer to the research question RQ1 as follows.

Ans. to RQ1: *In detecting phishing emails, the classification performance of the RF algorithm appears superior to the other ML algorithms in our study. This superiority is more evident when the algorithms are operated on the larger dataset. Especially the low number of FNs makes RF particularly suitable for phishing email detection.*

4.2 ML Algorithms' Training Time (RQ2)

We calculate the training time of six ML algorithms while separately training the datasets. Figure 5 and Figure 6 respectively present the training time taken by the ML algorithms when trained on the Ling and TREC datasets.

As seen in the figures, on both datasets, KNN and MNB algorithms have significantly shorter training times compared to the other ML algorithms. This is because these two algorithms are computationally simpler than the others.

The MNB algorithm uses probability theory to determine the likelihood that a new data point is a member of a certain class based on the feature values, whereas the KNN algorithm compares the new data point to all existing data points and classifies it based on the nearest neighbors. Both of these algorithms require less computation compared to others such as GB or RF. Both GB and RF utilize complex decision trees, and ensemble methods, and thus take more time to execute, which is also reflected in the figures. Based on the observations, the answer to the research question RQ2 is derived as follows.

Ans. to RQ2: *KNN and MNB algorithms have much shorter training times compared to the other ML algorithms studied.*

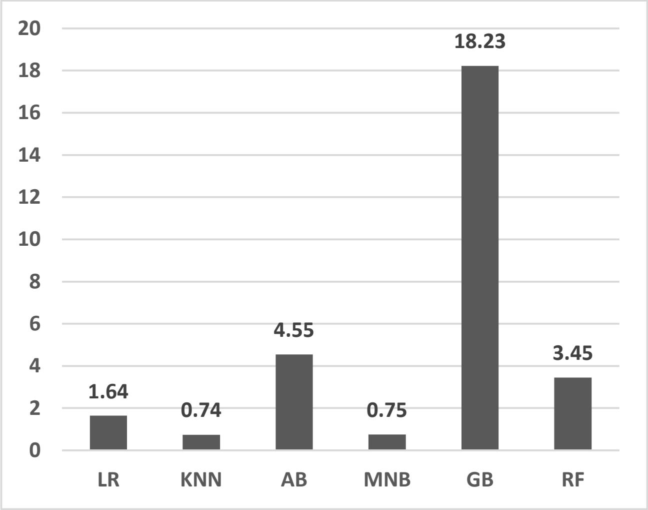

Fig. 5 Time (in seconds) consumed in training each of the six ML algorithms on the *Ling* dataset

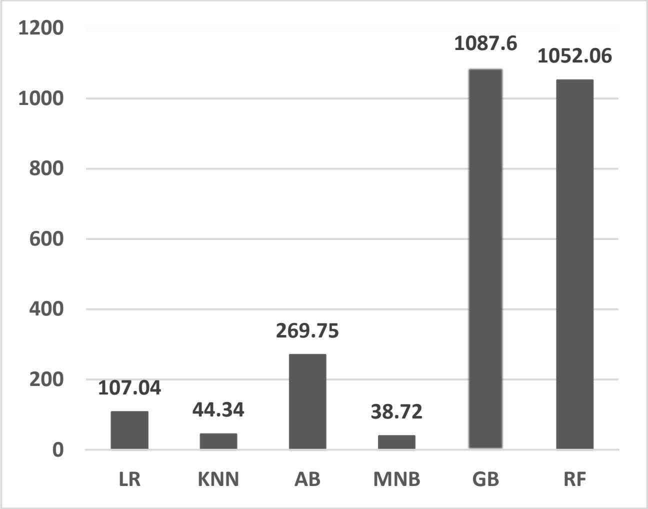

Fig. 6 Time (in seconds) consumed in training each of the six ML algorithms on the *TREC* dataset

4.3 Impact of the 'subject' Feature (RQ3)

To investigate the importance of the 'subject' feature in accurately detecting phishing emails, we operate the ML algorithms on the dataset according to the procedure described in Section 3.4, but at this phase, we make the ML algorithms disregard the 'subject' feature in the datasets.

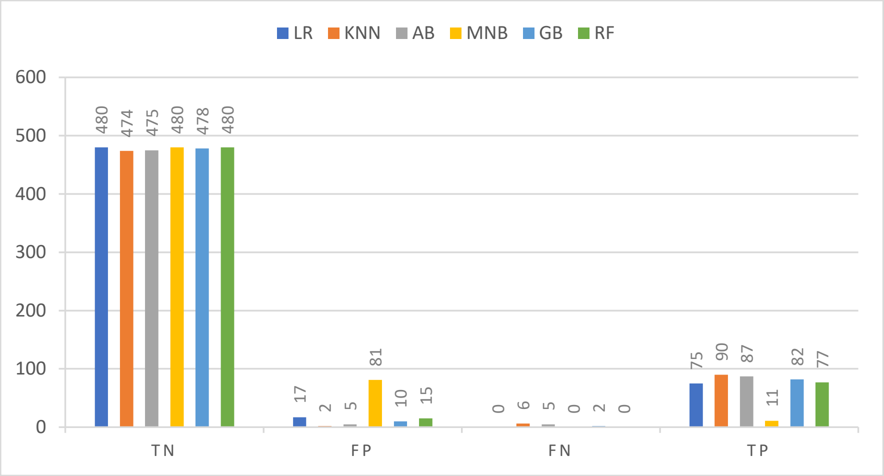

Fig. 7 TN, FP, FN, and TP for the ML algorithms on *Ling* dataset excluding the ‘subject’ feature

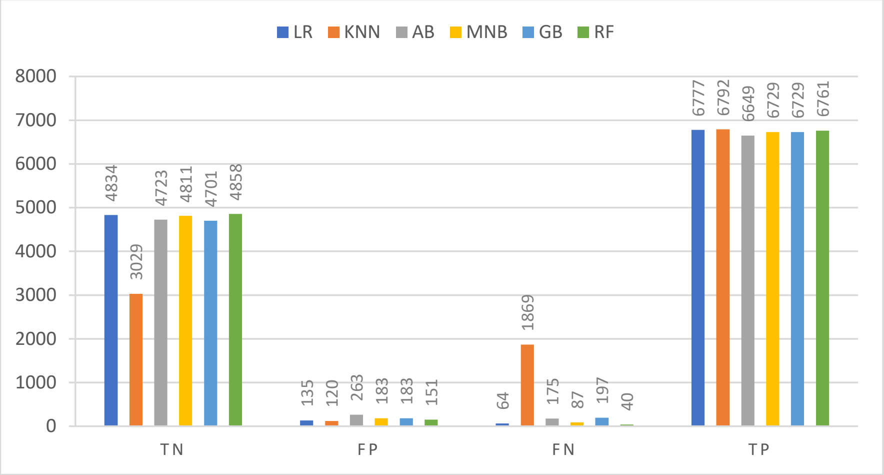

Fig. 8 TN, FP, FN, and TP for the ML algorithms on *TREC* dataset excluding the ‘subject’ feature

In Figure 7, we present the number of TN, FP, FN, and TP found in the ML algorithms’ detection of phishing emails in the *Ling* dataset excluding the ‘subject’ feature. If we compare the results in this figure with those of Figure 1, we notice that, when the ‘subject’ feature is excluded, there is a slight decrease in TP for all the algorithms. Figure 8 presents the number of TN, FP, FN, and TP found in the ML algorithms’ detection of phishing emails in the *TREC* dataset excluding the ‘subject’ feature. Now, if we compare the results in this figure with those in Figure 2, we again see that all the ML algorithms’ TP also decreases for the TREC dataset when the ‘subject’ feature is disregarded.

Table 7 Performance of the six ML algorithms on the *Ling* dataset, excluding the 'subject' feature

Algorithms	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**ROC**	**F-score (%)**
LR	97.03	97.58	97.03	0.9829	97.15
KNN	98.60	98.59	98.60	0.9666	98.59
AB	98.25	98.25	98.25	0.9676	98.25
MNB	85.84	98.31	85.84	0.9278	90.86
GB	97.90	98.05	97.90	0.9779	97.94
RF	97.38	97.81	97.38	0.9848	97.47

Table 7 presents the accuracy, precision, recall, and F-score of the ML algorithms' detection of phishing emails in the *Ling* dataset while disregarding the 'subject' feature. A comparison of the results in this table with those in Table 5 reveals the following. In this relatively smaller dataset, when the 'subject' feature is excluded, LR and KNN perform better, while AB, MNB, GB, and RF perform worse. Because the Ling dataset is relatively smaller, exclusion of the 'subject' feature might not have a substantial impact on the algorithms' performances. Alternatively, the 'subject' may not really have a distinguishable impact in the determination of whether an email is phishing or legitimate in general, irrespective of the algorithms in use. We further examine these possibilities by looking at the performances of the ML algorithms on the larger *TREC* dataset.

Table 8 Performance of the six ML algorithms on *TREC* dataset, excluding the 'subject' feature

Algorithms	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**ROC**	**F-score (%)**
LR	98.31	98.32	98.31	0.9817	98.31
KNN	83.16	88.55	83.16	0.8730	84.04
AB	96.29	96.29	96.29	0.9608	96.29
MNB	97.71	97.72	97.71	0.9753	97.71
GB	96.78	96.78	96.78	0.9670	96.78
RF	98.38	98.40	98.38	0.9820	98.38

In Table 8, we present the performance of the ML algorithms' phishing email detection in the *TREC* dataset without taking into account the 'subject' feature. If we compare the results of this table with those in Table 6, we notice that accuracy, precision, recall, and F-score decreases for all the algorithms when the 'subject' feature is excluded. This larger TREC dataset includes a varied collection of emails, making it more challenging for the ML algorithms to correctly distinguish phishing emails based only on message bodies without taking the 'subject' feature into account. Based on the observations and discussion above, we, therefore, derive the answer to the research question RQ3 as follows.

Ans. to RQ3: *Inclusion of the 'subject' feature increases accuracy of the ML algorithms in detection of phishing emails. However, this impact of the 'subject' feature is negligible when ML algorithms are applied on small datasets, but significant when the algorithms operate on larger datasets having a diverse collection of phishing and legitimate emails.*

5 Threats to Validity

Our phishing email detection model is developed using only two datasets (i.e., TREC and Ling datasets). The Ling dataset was released in 2003 while the TREC dataset was made available in 2007. These datasets, being old, might not have captured the latest tricks and strategies adopted in modern phishing attacks. Thus the generalizability of our findings can be argued limited in scope.

We solely focus on the 'subject' and 'message' (i.e., email body) features of the dataset in this work. However, the sender of an email, along with other header details, email attachments, and URLs in the email body can offer useful information and boost a classification model's performance. We plan to include all these in our future work.

We examine the effectiveness of six ML algorithms in detecting phishing emails. There are a number of other ML algorithms, which are popular but not included in our work. We plan to include them in future extension to this work.

The methodology of this study including the procedures for data collection and analysis are documented in this paper. The ML algorithms are well-known while the datasets used in this study are freely available to the public. Therefore, it should be possible to replicate this work of ours.

6 Related Work

ML and NLP have been used in many software engineering studies [13, 16–18, 36] while many other studies explored different security aspects of software systems [4, 15, 19, 20, 26, 29, 30]. Sattar et al. [32] applied predictive model analysis for detecting web-spams in web-graphs. Several recent attempts to detect phishing emails have tried with machine learning [1, 14, 35] and deep learning [8, 12, 14, 22, 28] techniques. These studies have used various strategies to improve the performance of detection algorithms. Some studies used a single algorithm [1, 8, 28], while others have compared several algorithms to identify the most effective approaches [14, 35]. Researchers have also tried using single [1, 35] and multiple datasets [8, 14, 28], as well as combination of multiple datasets [14].

Islam et al. [14] compared four ML algorithms on TREC and Ling datasets, using seven features of TREC and two features of Ling. Similar to our work, they also used the Ling and TREC datasets, and RF was reported to have the highest accuracy compared to the four ML algorithms included in their work. However, in their work, the accuracy of RF did not exceed 95% on the TREC dataset (and 96% on the Ling dataset) and while in our study RF achieves more than 98% accuracy on the TREC dataset (and over 97% on the Ling dataset).

Agarwal and Kumar [1] combined Naive Bayes (NB) with Particle Swarm Optimization and evaluated its performance on the Ling dataset. They found that while using NB alone resulted in an accuracy *below* 90%, the integrated approach achieved an accuracy above 90%. We have also used the Ling dataset and achieved almost 98% accuracy on this dataset while the Multinomial Naive Bayes (MNB) algorithm in our work has been able to achieve more than 97% accuracy on the TREC dataset.

Pan et al. [28] used a Semantic Graph Neural Network on the TREC dataset but did not compare their method with any other algorithms. Dhavale [8] employed a convolutional neural network-based approach on the TREC dataset without comparing their approach with any other algorithms. Similar to ours, Dhavale's method also achieved approximately 98% accuracy. However, unlike the work of Dhavale [8] or Pan et al. [28], we compared the performances of six ML algorithms.

Unlike any of the work discussed above, in our study, we examined the impact of a particular email feature on the six ML algorithms' detection of phishing emails. This also makes our work unique from the earlier work in the literature.

7 Conclusion and Future Work

In this paper, we have presented a performance comparison of six different machine learning (ML) algorithms in distinguishing phishing emails from legitimate ones. The algorithms are trained and evaluated on two publicly available datasets (i.e., Ling and TREC datasets). Classification performances are measured in terms of accuracy, precision, recall, F-score, and ROC. The training time required for the ML algorithms is also compared for measuring their applicability in a practical context. We have also examined the importance of a particular feature in the identification of phishing emails.

Our study reveals that the Random Forest (RF) algorithm performs better than other ML algorithms, particularly for larger datasets. Due to its low false negatives, RF can be particularly suitable for pragmatic application in phishing email detection, if training time is not a limiting factor. Classification performance of the ML algorithms generally decreases when the 'subject' feature is disregarded. The impact of this particular feature is found more substantial when the ML algorithms are operated on larger datasets. KNN (K-nearest Neighbours) and MNB (Multinomial Naive Bayes) algorithms tool much less time in training compared to other ML algorithms in our study. In our work, Gradient Boosting (GB) has taken the longest time in training, but this algorithm has not produced the best results in terms of classification performances. This suggests that simply choosing the algorithm with the longest training time may not necessarily lead to better results. It is important to choose an ML algorithm based on the size and nature of the datasets as well as classification performance requirements.

Our future research will address the limitations of this work and will investigate the use of additional features to improve the performance of ML algorithms in phishing email detection. There are other benchmark datasets for phishing emails, such as the Enron [21], Nazario [27], and SpamAssassin [33] corpora. These datasets are available in different formats, such as mdir and mbox. They can be combined into a single dataset in a single format, such as CSV, to create a bigger, more diversified dataset. This remains within our plans for future work. The planned extension of this work will also include additional ML algorithms. We hypothesize that applying deep learning approaches, such as convolutional neural networks (CNN), could improve the results by automatically recognizing complex patterns and features that might be difficult to detect with conventional ML approaches. We also plan to extend this work in this direction in the future.

Acknowledgement

This work is supported in part by a grant from the Center for Advanced Energy Studies (CAES) in Idaho, USA.

References

1. Agarwal, K., Kumar, T.: Email spam detection using integrated approach of naïve bayes and particle swarm optimization. In: 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 685–690. IEEE (2018)
2. APWG: Phishing Activity Trends Reports. https://www.f5.com/labs/articles/threat-intelligence/2020-phishing-and-fraud-report (Verified: March 2023)
3. Bisong, E.: Building machine learning and deep learning models on Google cloud platform. Springer (2019)
4. Champa, A., Rabbi, M., Eishita, F., Zibran, M.: Are we aware? an empirical study on the privacy and security awareness of smartphone sensors. In: 21st IEEE International Conference on Software Engineering, Management and Applications (SERA), p. (to appear) (2023)
5. Chatterjee, A.: Preprocessed TREC 2007 Public Corpus Dataset. https://www.kaggle.com/datasets/imdeepmind/preprocessed-trec-2007-public-corpus-dataset (Verified: March 2023)
6. D. Warburton: 2020 Phishing and Fraud Report. https://www.f5.com/labs/articles/threat-intelligence/2020-phishing-and-fraud-report (Verified: March 2023)
7. Das, A., Baki, S., El Aassal, A., Verma, R., Dunbar, A.: Sok: a comprehensive reexamination of phishing research from the security perspective. IEEE Communications Surveys & Tutorials **22**(1), 671–708 (2019)
8. Dhavale, S.: C-asft: convolutional neural networks-based anti-spam filtering technique. In: Proceeding of International Conference on Computational Science and Applications: ICCSA 2019, pp. 49–55 (2020)
9. Dou, Z., Khalil, I., Khreishah, A., Al-Fuqaha, A., Guizani, M.: Systematization of knowledge (sok): A systematic review of software-based web phishing detection. IEEE Communications Surveys & Tutorials **19**(4), 2797–2819 (2017)

10. G. V. Cormack: Trec 2007 spam track overview. In: Proc. of the 16th Text REtrieval Conference (TREC), vol. 500, p. 274 (2007)
11. Géron, A.: Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media, Inc. (2022)
12. Halgaš, L., Agrafiotis, I., Nurse, J.R.: Catching the phish: Detecting phishing attacks using recurrent neural networks (rnns). In: 20th International Conference on Information Security Applications, pp. 219–233 (2020)
13. Islam, M., Ahmmed, M., Zibran, M.: Marvalous: Machine learning based detection of emotions in the valence-arousal space in software engineering text. In: 34th ACM/SIGAPP Symposium on Applied Computing (SAC), pp. 1786–1793 (2019)
14. Islam, M., Al Amin, M., Islam, M., Mahbub, M., Showrov, M., Kaushal, C.: Spam-detection with comparative analysis and spamming words extractions. In: 9th International Conference on Reliability, Infocom Technologies and Optimization, pp. 1–9 (2021)
15. Islam, M., Zibran, M.: A comparative study on vulnerabilities in categories of clones and non-cloned code. In: 10th IEEE Intl. Workshop on Software Clones, pp. 8–14 (2016)
16. Islam, M., Zibran, M.: Leveraging automated sentiment analysis in software engineering. In: 14th IEEE International Conference on Mining Software Repository (MSR), pp. 203–214 (2017)
17. Islam, M., Zibran, M.: Deva: Sensing emotions in the valence arousal space in software engineering text. In: 33rd ACM/SIGAPP Symposium on Applied Computing (SAC), pp. 1536–1543 (2018)
18. Islam, M., Zibran, M.: Sentistrength-SE: Exploiting domain specificity for improved sentiment analysis in software engineering text. Journal of System and Software **145**, 125–146 (2018)
19. Islam, M., Zibran, M., Nagpal, A.: Security vulnerabilities in categories of clones and non-cloned code: An empirical study. In: 11th ACM/IEEE Intl. Symposium on Empirical Software Engineering and Measurement, pp. 20–29 (2017)
20. Joseph, R., Zibran, M., Eishita., F.: Choosing the weapon: A comparative study of security analyzers for android applications. In: Intl. Conference on Software Engineering, Management and Applications, pp. 51–57 (2021)
21. Klimt, B., Yang, Y.: The enron corpus: A new dataset for email classification research. In: Machine Learning: ECML 2004: 15th European Conference on Machine Learning, Pisa, Italy, September 20-24, 2004. Proceedings 15, pp. 217–226. Springer (2004)
22. Magdy, S., Abouelseoud, Y., Mikhail, M.: Efficient spam and phishing emails filtering based on deep learning. Computer Networks **206**, 108,826 (2022)
23. MimeCast: How to Stop Phishing Attacks (Whitepaper). https://www.mimecast.com/resources/white-papers/how-to-stop-phishing-attacks/ (Verified: March 2023)
24. MimeCast: The State of Email Security 2023 (E-book). https://www.mimecast.com/state-of-email-security/ (Verified: March 2023)
25. Mukherjee, A., Agarwal, N., Gupta, S.: A survey on automatic phishing email detection using natural language processing techniques. Int. Res. J. Eng. Technol **6**(11), 1881–1886 (2019)
26. Murphy, D., Zibran, M., Eishita, F.: Plugins to detect vulnerable plugins: An empirical assessment of the security scanner plugins for wordpress. In: Intl. Conference on Software Engineering, Management and Applications, pp. 39–44 (2021)
27. Nazario, J.: The online phishing corpus. http://monkey.org/ jose/wiki/doku.php (Verified: March 2023)
28. Pan, W., Li, J., Gao, L., Yue, L., Yang, Y., Deng, L., Deng, C.: Semantic graph neural network: a conversion from spam email classification to graph classification. Scientific Programming **2022**, 1–8 (2022)
29. Rajbhandari, A., Zibran, M., Eishita, F.: Security versus performance bugs: How bugs are handled in the chromium project. In: Intl. Conference on Software Engineering, Management and Applications, pp. 70–76 (2022)

30. Rodriguez, J., Zibran, M., Eishita, F.: Finding the middle ground: Measuring passwords for security and memorability. In: 20th IEEE International Conference on Software Engineering, Management and Applications, pp. 77–82 (2022)
31. Sakkis, G., Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Spyropoulos, C.D., Stamatopoulos, P.: A memory-based approach to anti-spam filtering for mailing lists. Information retrieval **6**, 49–73 (2003)
32. Sattar, N., Arifuzzaman, S., Zibran, M., Sakib, M.: Detecting web spam in webgraphs with predictive model analysis. In: 3rd International Workshop on Big Data Analytic for Cyber Crime Investigation and Prevention, pp. 4299–4308 (2019)
33. Schwartz, A.: Apache SpamAssassin. https://spamassassin.apache.org/ (Verified: March 2023)
34. Uchill, J.: Typo led to podesta email hack: Report. The Hill **13** (2016)
35. Unnithan, N.A., Harikrishnan, N., Vinayakumar, R., Soman, K., Sundarakrishna, S.: Detecting phishing e-mail using machine learning techniques. In: Proc. 1st anti-phishing shared task pilot 4th acm iwspa co-located 8th acm conf. data appl. secur. privacy (codaspy), pp. 51–54 (2018)
36. Zibran, M.: On the effectiveness of labeled latent dirichlet allocation in automatic bug-report categorization. In: 38th ACM/IEEE International Conference on Software Engineering (ICSE), pp. 713–715 (2016)