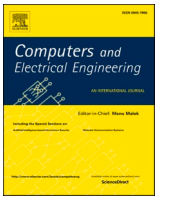
Computers and Electrical Engineering 120 (2024) 109625

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng



Novel interpretable and robust web-based AI platform for phishing email detection

Abdulla Al-Subaiey [a], Mohammed Al-Thani [a], Naser Abdullah Alam [b], Kaniz Fatema Antora [b], Amith Khandakar [c], SM Ashfaq Uz Zaman [d,*]

[a] Department of Computing Science, AFG College with the University of Aberdeen, Doha, Qatar
[b] Department of Computer Science and Engineering, University of Liberal Arts Bangladesh, Bangladesh
[c] Department of Electrical Engineering, College of Engineering, Qatar University, Qatar
[d] Qatar Emiri Naval Forces, Gulf Arabian, PO BOX 2237, Doha, Qatar

ARTICLE INFO

Keywords:
Phishing emails
Machine learning model
Email classification
Dataset
Explainable AI
User trust
Web-based application

ABSTRACT

Phishing emails continue to pose a significant threat, causing financial losses and security breaches. This study addresses limitations in existing research, such as reliance on proprietary datasets and lack of real-world application, by proposing a high-performance machine learning model for email classification. Utilizing a comprehensive and largest available public dataset, the model achieves a f1 score of 0.99 and is designed for deployment within relevant applications. Additionally, Explainable AI (XAI) is integrated to enhance user trust. This research offers a practical and highly accurate solution, contributing to the fight against phishing by empowering users with a real-time web-based application for phishing email detection.

1. Introduction

The proliferation of online scams and phishing attacks in the digital landscape presents a formidable challenge to cybersecurity efforts worldwide. Phishing, a malicious practice wherein attackers impersonate trusted entities to deceive individuals into divulging sensitive information, remains a pervasive threat. According to recent statistics from Phish Tank, there are over 45 thousand active phishing links, indicating the scale of the issue [1].

Addressing the complexities of phishing demands innovative approaches, among which Machine Learning (ML) and Artificial Intelligence (AI) stand out as promising avenues for strengthening defense mechanisms. ML and AI algorithms, fueled by extensive datasets and pattern recognition capabilities, offer real-time detection of evolving phishing tactics. As highlighted in Cloudflare's 2023 Phishing Threats Report [2] which analyzed data from over 13 billion emails, phishing attacks continue to evolve, with attackers increasingly leveraging deceptive links and identity deception tactics.

The urgency of combating phishing is underscored by its significant financial impact. According to the FBI, BEC attacks alone have cost victims worldwide over $50 billion [4]. Based on 4th Quarterly Report of Anti-Phishing Working Group, Inc there have been over 1077,501 phishing attacks in the fourth quarter of 2023 and 15 % for the attacks have been targeted through various webmail services [5]. As the figure (Fig. 2) below depicts,

* Corresponding author at: Qatar Emiri Naval Forces, Gulf Arabian, POBOX 2237 Doha, Qatar.
E-mail addresses: a.al-subaiey.20@abdn.ac.uk (A. Al-Subaiey), m.al-thani1.20@abdn.ac.uk (M. Al-Thani), naser.abdullah.cse@ulab.edu.bd (N. Abdullah Alam), kaniz.fatema.cse@ulab.edu.bd (K.F. Antora), amitk@qu.edu.qa (A. Khandakar).

https://doi.org/10.1016/j.compeleceng.2024.109625
Received 25 May 2024; Received in revised form 10 August 2024; Accepted 27 August 2024
Available online 10 September 2024

Moreover, studies indicate that 90 % of successful cyber-attacks originate from phishing attempts, therefore making developing robust detection and prevention strategies imperative.

This research aims to contribute to the ongoing efforts in combating phishing by harnessing the power of machine learning algorithms to classify phishing emails effectively. This project aims to develop a web-based application capable of discerning phishing emails from legitimate messages by leveraging insights gleaned from recent phishing trends.

This paper is structured to provide a comprehensive understanding of "Novel Interpretable and Robust Web-based AI Platform for Phishing Email Detection". We begin by introducing the research question and highlighting the significance of the study in Section 1. Section 2 presents a concise review of relevant literature to establish the current understanding of the field. In Section 3, we delve into the methodological approach, detailing the employed dataset, preprocessing techniques, and investigated algorithms. Section 4 focuses on the web deployment of the developed solution, while Section 5 provides a critical discussion of the results obtained. Finally, Section 6 offers concluding remarks, summarizing the key findings, and outlining potential future directions for research.

2. Literature review

While the widespread adoption of artificial intelligence (AI)-based tools has undeniably simplified many aspects of our lives, cybersecurity experts at Kaspersky warn of a potential downside. They believe the rapid growth of AI tools could lead to a double-edged sword: increased accessibility for beneficial applications, but also for malicious actors. This accessibility could fuel the development of more sophisticated cyberattacks. [6].

Recent research work on phishing detection primarily focusses on detecting phishing websites from URLs [7,8,9,10], website domain names [11,12] or website contents [13,14,15,16]. Machine learning (ML) has already shown promise in combating phishing and spam emails. Several studies have extensively reviewed various ML approaches in this field [17,18]. Another study [19] provided a concise overview of both ML and deep learning strategies, highlighting the critical role of high recall scores for practical implementations.

[20] reviewed studies on Business Email Compromise (BEC) phishing attacks, a sophisticated email scam that impersonates legitimate senders. The analysis of articles published between 2012 and 2022 found several key points: 1) Machine Learning (ML) is a promising approach to detect these evolving attacks, with Decision Tree, Support Vector Machine, and Neural Network being common techniques. 2) Examining email body and header features is crucial for detection, with many studies focusing on both. 3) Future research should explore dynamic feature selection, realistic datasets, integrating Natural Language Processing with deep learning, and combining ML with Explainable AI (XAI) for better detection.

There has also been research focusing on spear-phishing detection, that delves into a critical gap within spear-phishing detection, focusing on enterprise credential theft. While existing solutions offer general protection, the paper proposes ECSPAD, a system specifically designed to combat targeted attacks against enterprise credentials. ECSPAD utilizes a multi-layered approach that analyzes domain names for similarities (SCP & NCC) and verifies senders against whitelisted sources. The research demonstrates success against typo squatting techniques, but would benefit from a more comprehensive evaluation encompassing false positives/negatives and potential limitations of the ECSPAD system itself [21].

[22] applied a Graph Convolutional Network and various NLP techniques on the CLAIR collection of fraud emails and achieved an accuracy of 98.2 %. GCN is, a type of convolutional network that uses graph to represent the relation between entities, it converts the document classification problem into a node classification problem [23,24]. [25] experimented with various pretrained transformers and machine learning algorithms on publicly available datasets. The experiment achieved a f1 score of 98.66 % and accuracy of 98.67 % using a fine-tuned BERT transformer. [26] used the Enron email corpus with 6000 emails (3000 spam, 3000 ham) for training and a separate 200-email set (100 spam, 100 ham) for testing. They compared Naive Bayes and SVM with text-based features. SVM achieved higher accuracy (95.5 %) than Naive Bayes. Future work will explore richer features and other algorithms. [27] used text tokenization and then implemented a RNN classifier.

Researchers [28] explored machine learning for spam detection, with Naive Bayes achieving the best results. Bio-inspired algorithms like Genetic Algorithms and Particle Swarm Optimization acted as coaches, fine-tuning the machine learning models to achieve impressive accuracy (even 100 % in some cases). This work paves the way for more effective spam filtering by combining machine learning with optimization techniques. [29] proposes ELCADP, a novel ensemble model for lifelong spam classification. It tackles concept drift and catastrophic forgetting through dynamic data partitioning based on drift detection (EDDM). ELCADP achieves superior performance on the "Enron-Spam" dataset compared to other stream mining methods in terms of accuracy, precision, recall, and F1-score. However, its effectiveness with "virtual concept drift" (new class with same features) remains untested. Future work could explore ELCADP's application to phishing classification, a domain prone to this drift. [30] investigates ensemble methods for spam detection using a multinomial Naive Bayes baseline. Trained on the Kaggle "spam.csv" dataset, the ensemble achieved 98 % accuracy. [18] proposes a machine learning approach to detect unsolicited bulk emails (UBEs) using content and behavior-based features. The Random Forest classifier achieved 98.4 % accuracy on a ham-spam dataset and 99.4 % on a ham-phishing dataset. Future work includes improving robustness and exploring graphical features. [31] proposes an open-source tool for extracting a wide range of features from emails for spam detection. The tool extracts 140 features from the SpamAssassin Public Email Corpus (containing 5051 ham and 1000 spam emails). The performance of four machine learning models (J48 Decision Tree, Multilayer Perceptron, Naive Bayes, Random Forest) was evaluated using these features, achieving very high accuracy (except for Naive Bayes) compared to a previous study. The best performing model was Random Forest. [32] proposes XCSR#, a modification of XCSR (Learning Classifier Systems) to address sentiment analysis and spam detection in social media text. XCSR# tackles sparsity in the data by introducing "don't care" intervals, allowing classifiers to focus on relevant features. Compared to XCSR, XCSR# achieved significant improvement in both tasks

(sentiment analysis and spam detection) on datasets like tweets (2034 samples) and SMS spam (5575 samples). However, XCSR# has a potentially higher computational cost due to its more complex condition matching process. [33] proposes a novel email phishing detection method using a hybrid SVM-probabilistic neural network approach. A probabilistic neural network predicts the likelihood of something belonging to a specific category, instead of giving a simple yes/no answer. It achieves improved accuracy 97.5 % by dynamically collecting informative features from email text and leveraging content analysis. However, the real-world deployment of the model remains unaddressed. [34] applied RCNN using multilevel vectors and attention mechanisms combined with NLP techniques such as word2vec on the email body and header data and achieved 99 % accuracy.

Researchers designed D-Fence [35], a system to effectively detect phishing emails. D-Fence analyzes email structure, text, and URLs, achieving AUPRC of 98.51 % using CNN + LSTM with recall of 76.50 %. This surpasses individual analysis methods. While D-Fence offers excellent detection, the study explores ways to optimize its processing speed for real-world use. This involves selecting efficient algorithms and reducing unnecessary processing steps in each analysis module. Although it is an impactful system, it is mainly designed to be a secure email gateway. [36] experimented using various deep learning methods such as, RNN, CNN, LSTM, BERT and achieved an astonishing accuracy of 99.61 %, precision of 99.87 %, recall of 99.23 %, and f1-score of 99.55 %. However, the model was only trained on Enron, SpamAssassin and UCI Repository the model lacks generalizability. [37] researchers experimented with various BERT-based models and addressed the challenge of imbalanced datasets by applying a technique to create synthetic emails for under-represented classes, improving model accuracy. However, as there is limited information on the type of phishing emails the models were trained on it can be concluded the models are not generalized. [38] researchers address limitations in current phishing detection models. The researchers improved model robustness by training with "adversarial examples" (tricky phishing emails), achieving better real-world phishing detection. However, simply adding AI-generated phishing emails to training data wasn't effective. They proposed a promising defensive technique using K-Nearest Neighbour to identify disguised phishing emails with accuracy of 94 %. A transformer-based model is proposed here to classify phishing emails, achieving an impressive 99.51 % accuracy on standard datasets like Nazario and SpamAssassin. However, the research doesn't mention generalizability beyond the two specific datasets used for training (Nazario and SpamAssassin), raising concerns about its performance on unseen phishing tactics [39].

Researchers also explored if federated learning (FL) is a viable option for phishing email detection as FL offers a way to train a global anti-phishing AI model without data sharing between organizations, addressing privacy concerns. While FL performs well with balanced data and few participants, its effectiveness can drop with more organizations or unbalanced data. This study suggests that FL can still be useful, especially with models like BERT that handle unbalanced data better [40].

Literature review exposes limitations in phishing email detection. Most research relies on inaccessible private datasets or small public ones, hindering model generalizability and real-world deployment. Additionally, a gap exists between high-performing models and their practical application. This study addresses these shortcomings by proposing a robust model trained on a comprehensive public dataset and designed for practical use. To the best of our knowledge, this is the first study to propose a platform that integrates machine learning and web development to detect phishing email. An overall framework is shown in Fig. 3. The framework integrates various machine learning algorithms to a web based platform that allows technically challenged users to validate phishing emails. The framework utilizes the major publicly available datasets which increases the robustness and the generalizability of the machine learning network. In this study, we present a pioneering approach to phishing email detection for general public. The research work aims to:

- Address limitations of prior research:
  - Overcome the issue of using proprietary datasets by leveraging a diverse and comprehensive public dataset.
  - Improve generalizability by going beyond small, publicly available datasets used in past studies.
- Focuses on real-world applicability:
  - Design a model for deployment within relevant applications, bridging the gap between theory and practice.
  - Aim for a model that can be integrated into websites or applications for real-world use.
- Enhances transparency and trust:
  - Integrates Explainable AI (XAI) to make the model's prediction process more transparent and understandable to users.

The following section details the methodological approach employed in this research endeavor.

3. Methodology

The data processing pipeline for this study involved several key stages. First, six spam email datasets were meticulously chosen based on their unique characteristics. These datasets were then merged to create a unified corpus for analysis. Following this, a text preprocessing step was performed, which included tokenization to break down text into meaningful units, and the removal of punctuation and stop words to refine the data. Notably, subject and body text from specific datasets were merged into single "text_combined" columns to streamline further processing. Finally, the preprocessed data from both initial datasets (mdf_1 and mdf_2) were harmonized and integrated based on the "text_combined" column, resulting in a final dataset containing approximately 82,500 emails (42,891 spam and 39,595 legitimate). This prepared dataset was then subjected to feature engineering techniques like TF-IDF and Word2Vec to convert the textual data into a numerical format suitable for machine learning algorithms. Then the dataset was split into training and testing sets and the models were trained and tested. Finally, the best performing model was deployed in a web application.

The image (Fig. 3) below summarizes the complete process:

The detailed account of the research process, spanning from initial data acquisition and integration to the model's final

implementation is provided. The meticulous attention to data preparation and feature extraction, coupled with a judicious choice and clear exposition of machine learning algorithms, underpins the model's reliability and efficacy in identifying phishing attacks.

The article focused on developing a robust model for phishing email detection. To achieve this, six distinct spam email datasets were merged to create a diverse and comprehensive data pool. This merging strategy aimed to enhance the model's ability to generalize across different types of phishing emails and improve its robustness.

The preprocessing steps involved in the project were meticulously detailed. The first step was text cleaning, which involved removing irrelevant characters like special symbols and HTML tags that could distort the text data. The cleaned text was then tokenized, splitting it into individual words or tokens. All tokens were converted to lowercase to maintain uniformity, and common words (stopwords) were removed to focus the model on more informative words. Additionally, stemming and lemmatization techniques were applied to reduce words to their root forms and enhance the model's ability to recognize patterns.

Feature extraction was performed using the TF-IDF technique, which assigns weights to words based on their frequency in a document relative to their frequency across all documents. This approach helps highlight significant words in specific emails, aiding the model in detecting phishing emails. The dimensionality of the TF-IDF features was reduced to focus on the most informative features.

Feature engineering involved merging the textual features (sender email, date, subject, body) into a single column. This merging process aimed to capture the contextual relationships between these features and improve the model's performance by better understanding the overall structure and intent of the email.

The project compared two vectorization techniques, TF-IDF and Word2Vec, and ultimately chose TF-IDF for the final model due to its superior performance in phishing email detection. The models trained included Support Vector Machines (SVM), Multinomial Naive Bayes, and Random Forest. The performance of each model was evaluated based on metrics like the F1 score to determine their effectiveness in classifying emails as phishing or legitimate.

To provide insights into the model's decision-making process, the project utilized the LIME technique for Explainable AI (XAI). LIME visualizations were used to analyze which features (words) contributed most to the classification of an email as spam or not spam. This analysis helped in understanding the model's behavior and provided a basis for further model refinement.

The final model was deployed in a web-based application using Flask, allowing users to input email text and receive predictions. The deployment process involved creating a virtual environment to isolate dependencies and ensure compatibility.

In summary, the article demonstrated a meticulous approach to dataset merging, preprocessing, feature extraction, feature engineering, model training, and deployment. The use of Explainable AI techniques further enhanced the understanding of the model's decision-making process.

3.1. Evaluation of spam and phishing email datasets for research

This section analyses various email datasets commonly used in research on spam and phishing detection. Each dataset offers unique advantages and limitations, influencing its suitability for specific research goals.

Enron Phishing Email Dataset:

- Source: Extracted from the publicly available Enron email corpus.
- Size: Varies depending on source and filtering criteria (This research used 15,791 ham and 13,976 spam emails).
- Strengths: Readily available, large size.
- Weaknesses: Limited diversity (focused on Enron employees), potential bias towards specific attack tactics. Models trained solely on this data may not generalize well.

CEAS 2008 Spam Challenge Corpus:

- Source: Compiled by conference organizers, likely from public feeds or contributed examples. (Availability and terms of use require contacting organizers)
- Size: 39,154 samples (This research used 17,312 ham and 21,842 spam emails).
- Strengths: Diverse, reflecting a range of spam tactics prevalent during collection period.
- Weaknesses: Potential bias towards older spam tactics, limited accessibility. Models trained solely on this data may struggle with contemporary spam techniques.

Ling-Spam Corpus:

- Source: Publicly available collection of emails from the Linguist List online forum (2412 ham and 481 spam emails).
- Strengths: Easy access, balanced split between ham and spam for training models. Serves as a common benchmark for evaluating new techniques.
- Weaknesses: Limited size, domain-specific focus on linguistics (may not generalize well to broader spam types).

Nazario Spam Dataset:

- Source: Varied sources, potentially including public archives or honeypots (This research used a subset from Zenodo).

- Strengths: Focuses on phishing emails, useful for evaluating anti-phishing techniques. Serves as a baseline for new detection methods.
- Weaknesses: Unclear origin can introduce bias (public sources might contain filtered-out scams). Limited size and potentially outdated tactics compared to contemporary phishing attempts.

Nigerian Fraud Dataset:

- Source: Finding publicly available, well-documented datasets can be challenging. This research used a subset from Zenodo.
- Strengths: Targets a specific type of financial fraud (advance-fee scams), allowing tailored models. Provides insights into scammer tactics. Can serve as a benchmark for scam detection when properly labelled.
- Weaknesses: Limited size and availability compared to common spam types. Source bias (public sources might contain easily identifiable scams).

SpamAssassin Public Corpus:

- Source: Freely available collection maintained by Apache Software Foundation (This research used 4091 ham and 1718 spam emails).
- Strengths: Easy access, pre-labelled for straightforward training and evaluation. Serves as a common benchmark for comparing techniques.
- Weaknesses: Limited size compared to some datasets, potentially outdated tactics depending on version used. Potential bias towards spam types prevalent during corpus collection.

While each dataset has limitations, combining them can create a richer resource. Merging these datasets can address individual drawbacks and provide a more comprehensive view of spam and phishing tactics. This combined dataset can then be used to train models with greater generalizability and effectiveness in real-world scenarios.

3.2. Data collection and preprocessing

Six widely used spam email datasets were carefully selected based on their unique attributes. These datasets underwent a merging process to create a unified dataset for analysis. Among these, the Enron and Ling datasets contained three crucial columns: subject, body, and label, amalgamated into a singular data frame denoted as mdf_1. The CEAS, Nazario, Nigerian Fraud, and SpamAssassin datasets included sender, receiver, subject, body, date, and label columns, all integrated into another data frame labeled mdf_2.

3.3. Text processing

3.3.1. Tokenization and text cleaning

Utilizing advanced Natural Language Processing (NLP) techniques, the textual data underwent tokenization to break down words into meaningful units. Further, punctuation marks and stop words were removed to refine the text for subsequent analysis.

3.3.2. Text combination

The subject and body columns from mdf_1 were merged into a unified column labeled 'text_combined'. Similarly, the sender, date, subject, and body columns from mdf_2 were merged into a single column named 'text_combined'. This consolidation was executed to streamline further processing stages.

3.4. Data integration and preparation

The datasets originating from mdf_1 and mdf_2 was harmonized into a cohesive dataset. The data was merged in the 'text_combined' column. This integrated dataset comprised 42,891 spam emails and 39,595 legitimate (ham) emails, totaling almost 82,500 [44].

3.5. Feature engineering: text preprocessing and vectorization

The textual data within the email corpus requires preprocessing and transformation into a numerical format suitable for machine learning algorithms. This section details the feature engineering process, specifically focusing on vectorization techniques employed: Term Frequency-Inverse Document Frequency (TF-IDF) and Word2Vec.

3.5.1. Term frequency-inverse document frequency (TF-IDF)

TF-IDF is a statistical weighting scheme that evaluates the importance of a word within a document relative to the entire document collection (corpus) [45]. It considers two crucial factors:

1. Term Frequency (TF): Measures the frequency of a word appearing within a specific document. A higher TF indicates the word is more prominent within that document.
2. Inverse Document Frequency (IDF): Captures the global importance of a word across the entire corpus. Words that appear frequently across all documents will have a lower IDF score, signifying less discriminatory power. Conversely, words unique to a few documents will have a higher IDF, indicating their potential relevance for distinguishing specific content.

The mathematical formulation of TF-IDF for a word w in a document d within a corpus D is:

$$TF - IDF(w,\ d)\ =\ TF(w,\ d)\ \times\ IDF(w,\ D) \tag{1}$$

- $TF(w, d)$ is often calculated as the number of times word w appears in document d divided by the total number of words in document d.
- $IDF(w, D)$ is calculated using the logarithm of the total number of documents in the corpus ($|D|$)divided by the number of documents containing word $w(df(w))$.

TF-IDF addresses the shortcomings of simply using word frequency. By incorporating the IDF component, it reduces the weight of frequently used words that appear frequently across all documents and emphasizes terms specific to a particular document or category. In the context of phishing email detection, TF-IDF can highlight unique keywords or phrases commonly used in phishing attempts, enhancing the model's ability to identify such emails.

3.5.2. Word2Vec

Word2Vec is a neural network-based technique that represents words as numerical vectors. It leverages the idea that words with similar meanings tend to appear in similar contexts within a corpus. The training process allows Word2Vec to learn these semantic relationships and embed words into a vector space, where words with closer meanings will have more similar vector representations [46]..

3.5.3. Vectorization process

In this study, both TF-IDF and Word2Vec were employed to convert the preprocessed text within the 'text_combined' column into numerical representations. This transformation allows machine learning algorithms to analyze the textual features and identify patterns indicative of phishing emails.

3.6. Model development and evaluation

The implementation of 80–20 split is rooted in a deep understanding that with 82,486 samples lies in balancing computational efficiency with sufficient data for reliable model evaluation. The 80–20 split is computationally less expensive and faster to execute, making it ideal for initial model assessment and rapid prototyping. With a large dataset like this, the 20 % test set provides a substantial sample size for assessing model performance, ensuring that results are representative. The merged dataset was split into an 80–20 train test ratio. After the split, there were 65,988 training samples and 16,498 testing samples out of 82,486. In the 80 % of the total samples allocated for training, 10 % is used for validation purposed and hyperparameters tuning (which is explained in Model Parameter Section), which is a accepted approach to avoid under fitting and overfitting [Ref]. The authors also confirmed that the Test partition was not be used during training at all. Test partition was tested only once and the result is just reported as test result.

Training (80 %)		Testing (20 %)	Total
Training (70 %) 59,390	Validation (10 %) 6598	16,498	82,486

The thorough review presented by [17,18,19,20] suggest various techniques and algorithms implemented to address this challenge. From the vast list of techniques, we have chosen to equip the project with the following models to develop the system.

1. Support Vector Classifier (SVC)
   In this work, we employ a Support Vector Machine (SVM) for classification using a linear kernel. SVMs find a hyperplane that maximizes the margin between classes, effectively separating the data. The linear kernel facilitates efficient computation in high dimensions while maintaining interpretability. For binary classification, SVC with a linear kernel aims to find a straight line (hyperplane) in high dimensional space that best separates the two classes. This line maximizes the margin between the closest data points of each class (support vectors). New data points are then classified based on which side of the hyperplane they fall on. To ensure reproducibility, a random state of 42 is set for all computations.
2. Multinomial Naive Bayes Classifier (Multinomial NB)
   We utilize a Multinomial Naive Bayes (MNB) classifier, a probabilistic model suited for discrete features like word counts in text data. MNB calculates class probabilities based on feature independence and predicts the class with the highest likelihood. This approach offers efficiency and interpretability for text classification tasks.

3. Random Forest Classifier

We employ a Random Forest Classifier consisting of 100 decision trees for improved accuracy and reduced overfitting. Each tree acts independently, analyzing the data using a random selection of features at each branching point. When a new data point arrives, all the trees vote for a class based on their individual learned rules. Finally, the majority vote from the entire forest determines the final classification for the data point. This ensemble approach helps prevent overfitting and can potentially lead to better accuracy compared to a single decision tree. Setting the random state at 42 ensures reproducibility of the model.

The models were chosen due to the following reasoning,

1. Strong Performance
2. Computational Efficiency
3. Interpretability
4. Advantages over other models

The thorough study [47] shows both MNB and SVC have achieved high accuracy (over 95 %) in similar context. Furthermore, the research work states the simplicity compared to deep learning make these algorithms more interpretable. [26,47] suggest that SVM is slightly more robust compared to MNB however, MNB is known for its efficiency in handling large datasets. In our literature review we have also seen significant contribution of Random Forest Classifiers therefore we included all of the algorithms stated above.

3.6.1. Model parameter selection

In our study, we employed GridSearchCV to optimize the hyperparameters for three distinct classification algorithms: Support Vector Machine (SVM), Multinomial Naive Bayes (MNB), and Random Forest (RF). The optimal hyperparameters for SVM were identified as $C = 1$, $\gamma = 0.1$and a linear kernel, indicating a balanced regularization with a moderate level of influence for each support vector. For MNB, the best performance was achieved with an α value of 0.5, which represents a smoothing parameter that effectively handles zero-frequency issues by adjusting the influence of observed and unobserved features. The Random Forest model exhibited optimal results with 100 estimators, indicating the number of trees in the forest, and 'auto' for max features, which selects the maximum number of features as the square root of the total number of features. Additionally, a max depth of 20 was optimal, ensuring that each tree is grown to a sufficient depth to capture the underlying patterns without overfitting, and the criterion 'entropy' was selected for measuring the quality of splits, which aims to maximize the information gain. These tailored hyperparameter settings enhanced the predictive performance and generalizability of each model on our dataset.

3.6.2. Evaluation metrics

Comprehensive evaluation of the performance of the developed models in classifying phishing emails, various metrics were employed. These metrics provide insights into different aspects of a model's effectiveness:

Accuracy:

Accuracy is the most intuitive metric, representing the overall proportion of correctly classified emails. It is calculated as the number of true positives (correctly classified phishing emails) and true negatives (correctly classified legitimate emails) divided by the total number of emails:

$$Accuracy = \frac{(True\ Positives + True\ Negatives)}{Total\ Emails} \tag{2}$$

However, accuracy can be misleading, particularly in imbalanced datasets where one class (e.g., phishing emails) might be significantly smaller than the other (legitimate emails). In such cases, a model could achieve high accuracy simply by predicting the majority class.

Precision:

Precision focuses on the positive predictive value, indicating the proportion of predicted phishing emails that are true positives. It is calculated as the number of true positives divided by the total number of emails predicted as phishing emails (true positives + false positives):

$$Precision = \frac{True\ Positives}{(True\ Positives + False\ Positives)} \tag{3}$$

A high precision value suggests the model effectively avoids classifying legitimate emails as phishing emails (low false positive rate).

Recall:

Recall, also known as sensitivity, emphasizes the model's ability to identify all relevant phishing emails. It is calculated as the number of true positives divided by the total number of actual phishing emails (true positives + false negatives):

$$Recall = \frac{True\ Positives}{(True\ Positives + False\ Negatives)} \tag{4}$$

A high recall value indicates the model successfully captures most of the phishing emails within the dataset (low false negative rate).

F1-Score:

The F1-score is a harmonic mean that combines the strengths of precision and recall, providing a more balanced view of the model's performance. It is calculated as:

$$F1\_Score = 2 \times \frac{Precision * Recall}{(Precision + Recall)} \tag{5}$$

An F1-score closer to 1 signifies a well-balanced model with high precision and recall.

AUC-ROC (Area Under the Receiver Operating Characteristic Curve):

This metric goes beyond a single threshold and evaluates the overall performance of a classification model at various thresholds. It considers the trade-off between true positive rate (correctly identifying phishing emails) and false positive rate (incorrectly classifying legitimate emails as phishing) across all possible thresholds. A higher AUC-ROC indicates a better model at distinguishing phishing emails from legitimate ones.

By evaluating these metrics in conjunction, we gain a comprehensive understanding of the model's effectiveness in detecting phishing emails. A high accuracy score along with balanced precision and recall indicates a robust model for real-world application.

3.7. Model interpretability with lime

While achieving high performance is crucial, understanding the rationale behind a model's predictions is equally valuable. This section details the application of Local Interpretable Model-Agnostic Explanations (LIME) to gain insights into the decision-making process of our machine learning model for phishing email detection.

LIME is a technique for explaining individual predictions made by any complex machine learning model. It operates by approximating the behavior of the original model around a specific data point (an email instance in our case) with a simpler, interpretable model, often a linear model. This local explanation model focuses on features within the data point that hold the most influence on the original model's prediction.

In the context of our research, LIME is employed to analyze how the model classifies an email as phishing. LIME takes a pre-classified email instance (e.g., an email flagged as phishing by the model) and generates an explanation for that classification. This explanation highlights the features within the email content that most significantly contributed to the model's decision. These features might include specific words, phrases, or patterns within the email text.

3.8. Model deployment

Following its superior performance on evaluation metrics, the chosen model is exported and integrated into a Flask web application. This deployment strategy facilitates real-time spam email classification, enabling users to interact with the model through a user-friendly web interface.

4. Results

4.1. Model results

The table below concisely summarizes the experiment results. In the dataset column, first, we have the number of samples and then the class. For example, in the first row, 28,457[1] means there are 28,457 spam emails, and 21,403[0] means 21,403 ham emails. In the table, [1] and [0] refer to spam and ham classes, respectively.

4.1.1. Friedman test results

The Friedman test was employed to statistically assess whether there are significant differences in performance among the three machine learning algorithms evaluated: Support Vector Machine, Multinomial-Navies Bayes, and Random Forest. The ranks of these algorithms were based on their average performance across five metrics: Accuracy, Precision, Recall, F1-Score, AUC-ROC.

4.2. Friedman statistic

The calculated Friedman statistic (X_F^2) is calculated as follows:

$$X_F^2 = \frac{12N}{K(K+1)} \left(\sum_{i=1}^{k} R_i^2 \right) - 5N(k+1) \tag{6}$$

Where:

- NN is the number of datasets (since ranks are averaged, assume NN is the number of metrics, which is 5: Accuracy, Precision, Recall, F1-Score, AUC-ROC).
- k is the number of algorithms, which is 12.
- $R_i$is the sum of the ranks for the i th algorithm.

The calculated value of X_F^2 is 0.8573.

4.3. Critical value

The critical value for the Friedman test depends on the significance level (usually α=0.05) and the degrees of freedom, which is k-1. In this case, $k = 12$, so the degrees of freedom are 11.

Using a chi-squared distribution table, the critical value for X^2 with 11 degrees of freedom at α=0.05 is approximately 19.675.

4.4. Interpretation

Compare the calculated Friedman statistic to the critical value:

- Calculated Friedman Statistic (X_F^2) = 0.8573.
- Critical Value ($X_{0.05,\ 5}^2$) = 19.675

As 0.8573 is significantly <19.675 we fail to reject the null hypothesis. This means there is no statistically significant difference in the ranks of the algorithms (SVM, MNB, RF) across the metrics (Accuracy, Recall, Precision, F1 Score, AUC-ROC) according to the Friedman test at the 0.05 significance level.

Considering the performance metrics, statistical tests, deployment environment we have selected **SVM** with **TF-IDF** preprocessing on the merged dataset, achieved 99.1 % accuracy, 99 % precision, 99 % recall, and f1-score 99 as the best performing model.

The proposed model achieved the following results, Accuracy: 99.19, Precision: 99.00 Recall: 99.00, F1-score: 99.00, AUC-ROC: 99.19.

The following section describes the deployment of the best performing model within a web application. We demonstrate the application's functionality using real-world data and showcase prediction visualization techniques like LIME for interpretability.

5. Web-based platform

The figure above (Fig. 4) depicts how the user shall interact with the web application. The moment the user receives a suspicious email, the user can copy and paste the text contents (sender address, subject, body text) directly to the website and submit the form. The model embedded within the website will instantly send a prediction weather the email contents are spam or safe.

The web application utilizes a single-tier, monolithic architecture. It combines the frontend and backend functionality within a single Flask application, including both the Python code utilizing Flask libraries and the HTML templates delivered to users. For deployment, the application leverages a virtual environment. This isolates the application's dependencies from the system's overall Python environment, ensuring compatibility and avoiding conflicts.

The current server specifications allocate 0.1 CPU and 512 MB RAM. While functional, these resource limitations are important to consider, especially for future growth. It's important to note that due to these limitations, running computationally expensive models like deep learning models and transformer models is not possible. These models typically require significant GPU support for efficient execution [48].

Since the application currently doesn't store user information for primary use, security risks are reduced. However, as the application evolves, consider implementing appropriate security measures. the application includes a feedback mechanism allowing users to report misclassified emails. Only emails flagged as misclassified, along with the reporter's contact information provided with their consent, will be stored temporarily. These stored emails will undergo validation by a human operator with admin-level access to ensure accuracy and maintain data integrity.

The Figs. 5 and 6 below, are a demonstration of the application and the model in real time. The model can predict unseen and real-life emails. The user pastes the email text contents as can be seen below in the form. Then when the verify button is pressed a request is sent to the model to predict the contents. The web application processes the text data and converts it into a vector and then passes the data to the model. The model predicts based on the rich set of features it was trained on and provides a result.

After making a prediction, users have the option to provide feedback. They can either like or dislike the result. If a user dislikes the prediction, they will be directed to a feedback form, as shown in Fig. 7, to explain their reasoning.

The form collects the spam message, rationale for the feedback, user contact information, and user consent. The contact information is stored solely to prevent system abuse. Once the feedback form is submitted, the spam message will undergo manual validation. Depending on the outcome, it may be used to retrain the model for future predictions.

5.1. Feature importance analysis and prediction visualization using lime

Understanding the inner workings of machine learning models, particularly in the context of text classification, is crucial for interpreting their decisions and enhancing their performance. This section explores the feature importance analysis within the selected Support Vector Machine (SVM) model. Feature importance analysis helps identify which aspects of the input data most significantly impact the model's predictions. This is particularly challenging for non-linear models like SVMs, where complex feature transformations can obscure the direct influence of individual features.

Fig. 8 highlights the most significant features identified by the SVM model. Each bar corresponds to a feature, such as "url," "rssfeed," and "enron," with the length of the bar representing the importance score attributed to that feature. Features with higher scores, like "josemonkeyorg," "2019," "url," "713," and "edu," emerge as the key drivers of the model's predictions. This visualization not only underscores the pivotal features but also underscores the nuanced challenges of interpreting feature importance within non-linear models.

By leveraging LIME for feature importance analysis, we provide a comprehensive understanding of the influential factors guiding the SVM model's predictions, thereby fostering trust and facilitating model refinement.

The LIME visualization presents a snippet of a phishing email and analyzes the likelihood of it being spam [49]. The email subject is "Personal Assistant Opportunity - Dr. Sheldon Cooper" and it is addressed to an unknown applicant. The email body offers a work-from-home assistant position with a competitive salary. It specifies tasks such as errands, communication, and some academic responsibilities. The sender requests the applicant to send their CV, phone number, and a scanned copy of their passport for verification to an email address (shel.cooper@caltech.edu) that appears to be affiliated with California Institute of Technology (Caltech). The email further asks the applicant to fill out and scan a job application form. The demo email was generated based on the recent events from University of Aberdeen [50].

The LIME visualization (Fig. 9) divides the prediction into two sections: Not Spam and Spam. The "Not Spam" section has a probability score of 0.08, while "Spam" has a score of 0.92, indicating a high likelihood of the email being spam.

Highlighted words within the email body are colored red, signifying their contribution to the spam classification. These words include "scan" (0.11), "miss" (0.10), "Fill" (0.10), "phone" (0.08), and "Dear" (0.08). Conversely, the word "edu" within the sender's email address is colored blue and has a weight of 0.03, indicating that it contributes slightly to the email being classified as "Not Spam" because "edu" addresses are commonly associated with educational institutions.

In essence, the LIME visualization demonstrates how the model identifies specific keywords typically found in phishing emails, such as requests for personal documents (passport scans) and urgency ("Don't miss out on this amazing chance!"), to classify the email as spam.

In our next set of examples, we will examine emails regarding account security.

The LIME visualization breaks down the model's prediction for the email (Fig. 10) "Account Suspension Notification: Immediate Action Required" into two sections: Not Spam and Spam. The "Spam" section has a significantly higher probability (1.00) compared to "Not Spam" (0.00), indicating the model is highly confident this email is spam.

Several words within the email body are highlighted in red, signifying their contribution to the spam classification. These include "Account" (0.91), "Suspension" (0.89), "Immediate" (0.87), "Attention" (0.83), "verify" (0.79), "identity" (0.78), "link" (0.75), "click" (0.74), and "instructions" (0.72). These words are commonly used in phishing emails that attempt to create a sense of urgency or trick recipients into revealing personal information.

There are no words highlighted in blue, indicating there are no features contributing to the email being classified as "Not Spam".

In conclusion, the LIME visualization highlights how the model leverages the presence of keywords typically associated with phishing scams, such as "account suspension" and "urgent attention required," to classify the email as spam. The absence of any features pointing towards a legitimate email strengthens the model's confidence in its prediction.

The example above was a visualization of Spam emails, in the following example we examine a similar case, expect the email used was a legitimate email from Google Account Service (Fig. 11).

The "Not Spam" and "Spam" section both have an equal probability score of 0.50.

Highlighted words within the email body are colored red, signifying their contribution to the spam classification. These words include "account" (0.33), "sign-in" (0.08), and "secure" (0.17). These words are commonly found in both phishing emails and legitimate notifications from companies about account activity.

The only highlighted word in the sender's email address is "Google" (0.031) which has a weight contributing to the email being classified as "Not Spam" because Google is a well-known legitimate sender.

In essence, the LIME visualization demonstrates that the model is unsure about this email. While words like "account" and "sign-in" are commonly used in phishing emails, the presence of "Google" in the sender's address and the overall context of the email ("We noticed a new sign-in to your Google Account on a Windows device") suggests it might be legitimate. However, since it is 50 % in both classes, the model will label this email as spam. The same email with sender information drastically changes the prediction, as demonstrated below (Fig. 12).

The "Not Spam" section has a probability score of 0.77, while "Spam" has a score of 0.23, indicating a higher likelihood of the email being legitimate. The indication for the email being a spam is the same as it was in the previous scene. However, after including the sender information the probability of the email being a legitimate increased by 17 %.

Based on the visualizations discussed, the key factors the model is considering before predicting whether an email is spam are highlighted below,

- **Keywords:** The model places significant weight on specific words and phrases commonly found in phishing emails. These include:
  - **Urgency:** Words like "immediate," "attention required," or "suspension" create a sense of urgency and pressure the recipient to act quickly.
  - **Account Security:** Words like "account," "verify," "identity," "secure," or "sign-in" often appear in phishing attempts targeting account credentials.
  - **Suspicious Activity:** Words like "suspicious activity" or "unusual sign-in" can raise red flags for the model.

- **Sender Address:** While not as heavily weighted as keywords in the body, the sender's address can also influence the model's prediction. The presence of a well-known legitimate sender's domain name (e.g., "Google") can slightly nudge the model towards a "Not Spam" classification.

The visualizations also reveal the model's confidence level in its prediction. A higher probability score for "Spam" and a greater number of highlighted keywords indicate a stronger signal for phishing. Conversely, a lower spam score and the presence of mitigating factors like a legitimate sender address can lead to a more uncertain prediction. It's important to note that these visualizations only showcase a small sample. The actual model might consider a broader range of features, such as, analyzing formatting, presence of attachment or unusual character set.

Overall, LIME visualizations provide valuable insights into the model's decision-making process for spam detection. By understanding the key factors considered, we can better assess the model's strengths and limitations in identifying phishing emails.

6. Discussion

This section delves into the feature engineering and model selection processes. We explore the impact of different vectorizers (word2vec vs. tf-idf) and feature ablation to identify the most informative features for phishing email detection. We then analyze the effectiveness of merging textual features and explore the benefits of Explainable AI (XAI) for model interpretability. Finally, we discuss the potential for deep learning integration and user feedback incorporation in future iterations, along with the security considerations associated with user feedback.

6.1. Feature engineering and model selection

This study investigated the influence of feature preprocessing techniques on model performance for phishing email detection. Two common vectorizers, word2vec and tf-idf, were compared. Our experiments revealed that tf-idf achieved superior results, with an F1 score of 0.99 compared to the maximum F1 score of 0.83 obtained using word2vec. Based on this finding, tf-idf was employed for subsequent model training.

We further conducted feature ablation to identify the most informative features for the classification task. Initially, all available features (sender email, receiver email, date, subject, body, URL) were included in the model. We hypothesized that the receiver email address would have minimal influence on phishing email detection, as spammers often employ spoofing techniques. This hypothesis was validated, as removing the receiver email feature resulted in no significant performance change. Similarly, the URL column, containing binary data (indicating presence or absence of a URL), appeared to have minimal predictive power, and was excluded without impacting model performance.

A more impactful observation was the significant improvement in model performance achieved by merging all textual features (sender email, date, subject, and body) into a single column. This merged feature yielded a notable increase in F1 score, from 0.71 to 0.82. This suggests that combining textual information provides a richer representation for the model compared to using isolated features. The merged feature captures contextual relationships between these textual elements, potentially aiding the model in identifying phishing patterns. Finally, SVM's ability to consider word relationships, unlike Naive Bayes' assumption of word independence, contributes to its better performance.

6.2. Enhancing transparency with explainable ai (XAI)

To promote interpretability and gain insights into the model's decision-making process, we integrated Explainable AI (XAI) techniques. This approach aims to visualize the features that most significantly influence the model's predictions. In the context of phishing email detection, XAI can be used to identify specific words or phrases that contribute the most to classifying an email as spam. This information can be valuable for improving future iterations of the model and potentially for user education, highlighting the red flags commonly used in phishing attempts.

This research establishes a foundation for exploring deep learning methodologies within the web application's framework. Phishing tactics are constantly evolving. While our model demonstrates strong performance on the datasets used, its ability to generalize to entirely new and unforeseen phishing methods remains uncertain. Regular updates and retraining with the latest data are necessary to maintain the model's relevance and effectiveness. The integration of user feedback through a dedicated section, allowing users to share insights and experiences with the model's predictions. This valuable feedback loop can inform model refinement and enhance overall effectiveness. However, implementing user feedback introduces security considerations such as spam. Countermeasures like form validation and CAPTCHA verification can mitigate these risks, while exploring user reputation systems or moderation tools can further strengthen the platform's security and integrity.

Compared to prior research, including spear-phishing detection methods focused on specific elements [21], our work offers a broader defense against phishing emails. We achieve this through a more comprehensive data collection strategy. By merging six diverse spam datasets, we create a richer data pool that allows the model to learn from a wider variety of phishing tactics. Additionally, we employ established feature engineering techniques (TF-IDF, Word2Vec) to improve model performance. This focus on data quality and feature engineering strengthens our approach to generalizable phishing email detection, making it more effective against a wider range of attacks.

The research focused on using LIME (Local Interpretable Model-Agnostic Explanations) to gain insights into the decision-making

process of a machine learning model for phishing email detection. LIME was applied by analyzing individual email classifications and identifying the features that significantly influenced the model's decision.

The impact of using LIME for interpretability was twofold. Firstly, it helped in understanding the importance of different features in the classification process. LIME visualizations highlighted the most influential keywords within an email that contributed to its spam classification. This allowed users to comprehend the reasoning behind the model's flagging of an email as spam.

Secondly, LIME aided in targeted model improvement. By identifying the key features used for classification, researchers could refine the model in future iterations. For example, they could focus on improving the model's ability to handle variations of the keywords identified by LIME as indicators of phishing attempts.

Additionally, LIME visualizations could be used to educate users about common red flags found in phishing emails. By highlighting the specific words or phrases identified by the model as suspicious, users could become more aware of the tactics employed by phishers.

The examples provided in the text demonstrated how LIME identified specific words that contributed to the classification of certain emails as spam. For instance, words like "scan," "miss," "fill," and "phone" were identified as contributing factors in classifying an email offering a work-from-home position as indicators of phishing attempt.

6.3. Why tf-idf outperformed WORD2Vec

The researchers in this study chose to use TF-IDF over Word2Vec for phishing email detection. TF-IDF was selected because it excels at highlighting terms that are highly indicative of a specific document within a corpus, which is important for identifying fraudulent messages in phishing emails. It is also computationally efficient, making it suitable for large datasets like email corpora, which is crucial for real-time or near-real-time phishing detection systems. Additionally, the combination of term frequency and inverse document frequency in TF-IDF provides a nuanced representation of word importance, downplaying common, less informative words while emphasizing those that are more likely to distinguish between legitimate and phishing emails.

On the other hand, Word2Vec may not be optimal for phishing email detection due to several reasons. Word2Vec primarily focuses on capturing semantic similarities between words, which may not be as crucial for phishing detection where specific keywords and phrases hold more weight. Training Word2Vec models can also be computationally expensive, especially for large datasets, which can be a bottleneck in real-time applications. Additionally, Word2Vec embeddings are context-dependent, meaning the meaning of a word can change based on its surrounding words, which can be challenging in phishing emails where the context can be highly varied and deceptive.

In the context of phishing email detection, TF-IDF's ability to quickly identify distinctive terms and assign them appropriate weights is a significant advantage. Phishing emails often rely on specific language patterns, deceptive phrases, and urgent tones to manipulate users, and TF-IDF is adept at capturing these linguistic cues. Furthermore, the relatively straightforward nature of phishing email classification compared to tasks requiring deep semantic understanding makes TF-IDF a suitable choice. While Word2Vec might offer some benefits in more complex NLP scenarios, its computational overhead and potential overfitting in this specific task outweigh its advantages.

By carefully considering the nature of the problem and the strengths and weaknesses of different feature engineering techniques, the researchers made an informed decision in favor of TF-IDF. This choice ultimately contributed to the high performance of the phishing detection model. In conclusion, TF-IDF's efficiency, focus on term importance, and ability to handle large datasets made it the more effective choice for phishing email detection in this study.

6.4. Potential limitations and biases in dataset and mitigation strategies

The research diligently addressed potential shortcomings and prejudices inherent within the dataset. Though completely unbiased data remains an ideal, the employed methodologies were designed to curtail their influence, thereby producing a phishing detection model characterized by resilience, adaptability, and broad applicability. Sustained oversight and model refinement, coupled with the incorporation of fresh data, are imperative for upholding its efficacy in practical scenarios.

1. Dataset Diversity and Representativeness

   Potential Limitation: The merged dataset combines six distinct spam email datasets: "Enron," "Ling-Spam," "SpamAssassin," "TREC," "Phishing Corpus," and "SMS Spam Collection." Although this aggregation creates a rich and diverse dataset, it may not fully represent the constantly evolving landscape of phishing and spam emails. The emails in these datasets were collected at different times and from various sources, which might lead to an underrepresentation of newer phishing tactics or region-specific spam campaigns.

   Mitigation Strategy: To address this limitation, a continuous update mechanism could be implemented, where the model is periodically retrained with newer datasets. Incorporating real-time or recent data streams, such as publicly available phishing feeds or crowdsourced email reports, can help maintain the model's relevance and adaptability to emerging phishing techniques. Additionally, stratified sampling was used during the dataset merging process to ensure that all categories of phishing emails were adequately represented in the final training set.
2. Class Imbalance

   Potential Limitation: Phishing emails typically constitute a small percentage of the overall email traffic compared to legitimate emails. This inherent class imbalance can lead to a model that is biased towards predicting the majority class (non-phishing emails),

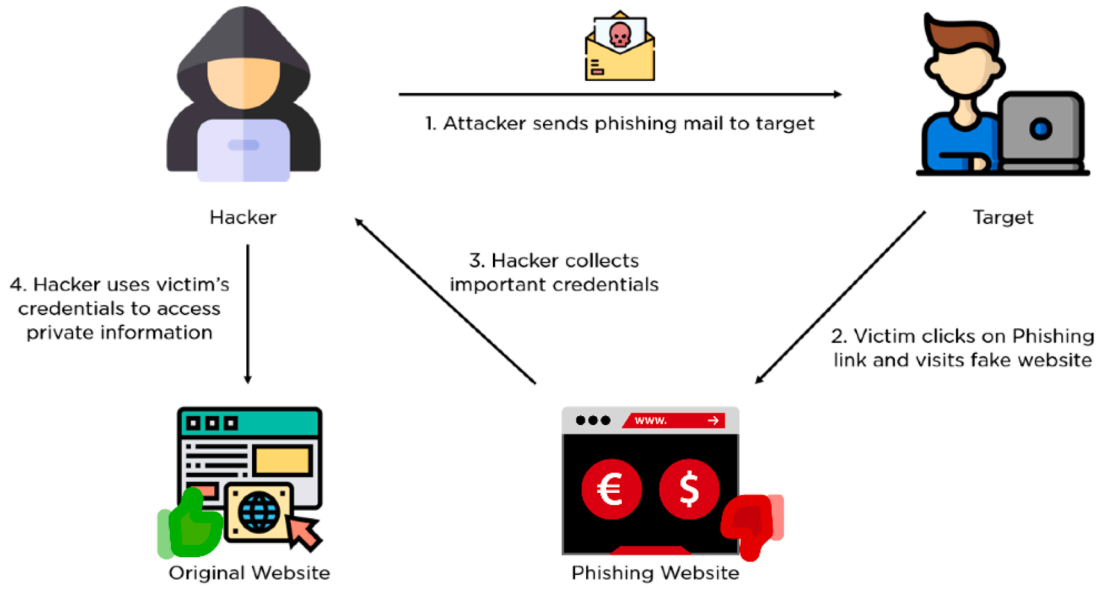


Fig. 1. Phishing using Email [3].

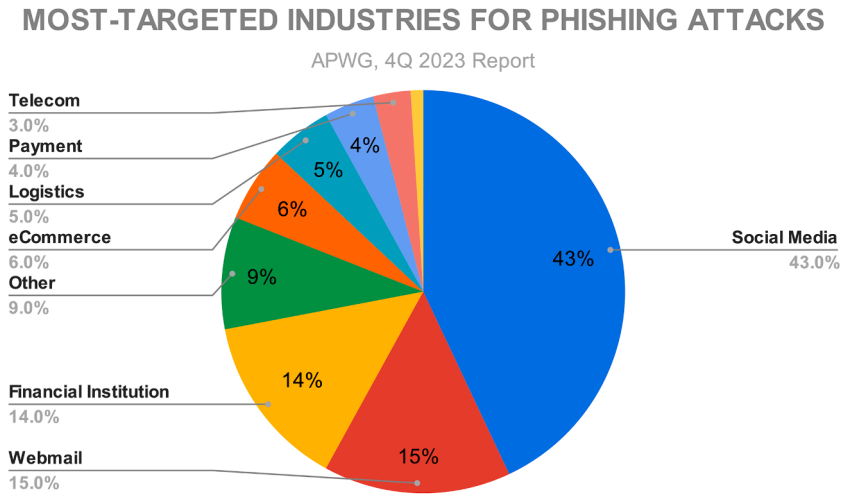


Fig. 2. Most-Targeted Industries for Phishing Attacks. Image generated from APWG 4Q 2023 Report Data.

potentially resulting in a high number of false negatives (phishing emails misclassified as legitimate).

Mitigation Strategy: Several techniques were employed to mitigate the effects of class imbalance:

- Over-sampling and Under-sampling: Techniques like SMOTE (Synthetic Minority Over-sampling Technique) were applied to balance the classes by creating synthetic samples of the minority class. Conversely, under-sampling of the majority class was also considered to prevent overfitting to the majority class.
- Use of Performance Metrics: Instead of focusing solely on accuracy, which could be misleading in imbalanced datasets, the F1 score was used as a primary evaluation metric. The F1 score balances precision and recall, providing a more meaningful measure of the model's performance on the minority class (phishing emails).

3. Feature Bias

Potential Limitation: The reliance on certain textual features could introduce bias, especially if these features are not consistently representative of phishing content. For example, certain words or phrases that are commonly used in phishing emails in one dataset might be less common in another, leading to potential feature bias.

Mitigation Strategy:

- Feature Engineering: Careful feature engineering was conducted to select features that are robust across different datasets. The use of TF-IDF helps mitigate feature bias by weighing terms based on their importance across all documents, rather than their frequency in a single dataset.

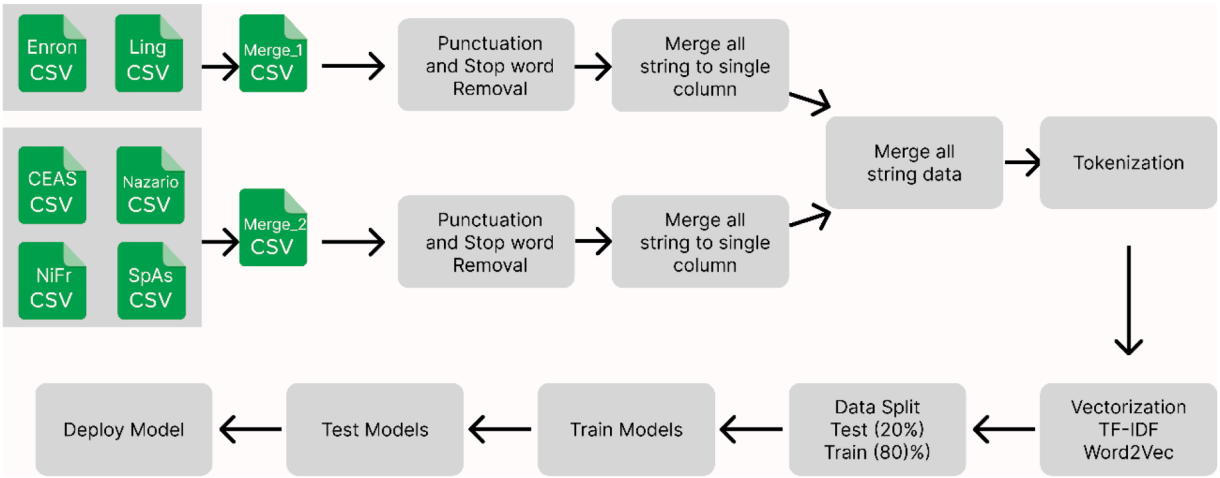


Fig. 3. Methodology.

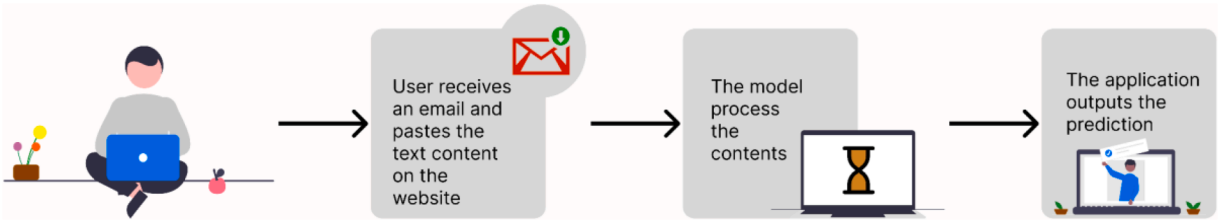


Fig. 4. Web-application process flow.

Fig. 5. Real world email before prediction.

- Cross-validation: Cross-validation was used extensively during model training to ensure that the model generalizes well to different subsets of the data, reducing the risk of overfitting to biased features.

4. Source Bias

Potential Limitation: Each of the six datasets used in the project comes from different sources, such as corporate environments (e.g., Enron) or spam traps (e.g., SpamAssassin). These sources may have inherent biases, such as corporate email patterns in Enron or spam that targets specific types of users in SpamAssassin. These biases might not fully reflect the general population of email users.

Mitigation Strategy:

- Dataset Merging and Normalization: By merging multiple datasets, the project aimed to dilute the biases inherent in any single source. Additionally, normalization techniques were applied during preprocessing to ensure that features like email structure or language, which might be source-specific, were standardized across the combined dataset.
- Source-agnostic Feature Selection: Features that are less likely to be tied to a specific source (e.g., content-based features rather than metadata) were prioritized during feature selection to create a more generalized model.

Fig. 6. Real world email after prediction.

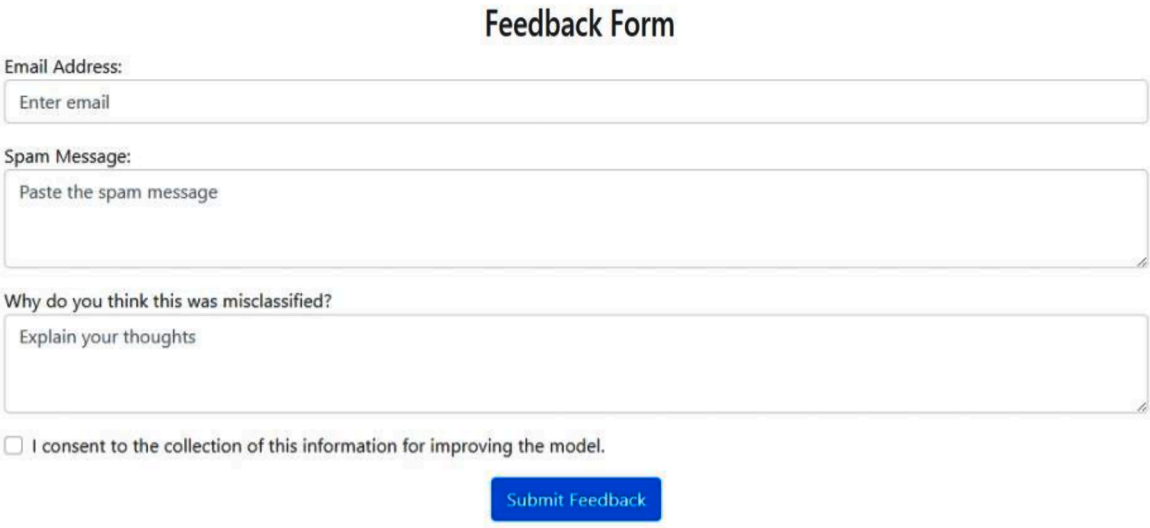


Fig. 7. Web application prediction feedback form.

5. Temporal Bias

   Potential Limitation: The emails in the dataset span different time periods. Some phishing tactics that were prevalent in earlier years might no longer be relevant, while new tactics might not be well represented in older datasets. This temporal bias can affect the model's ability to detect current phishing strategies.

   Mitigation Strategy:

   - Temporal Validation: The dataset was split into training and testing sets based on different time periods to evaluate how well the model performs on more recent data. If a significant drop in performance was observed on newer data, this indicated temporal bias, which was then addressed by updating the model with more recent samples.
   - Continuous Learning: Implementing a continuous learning approach, where the model is periodically retrained with new data, was considered as a future enhancement. This strategy helps the model stay updated with the latest phishing tactics, thus mitigating temporal bias.

6. Language and Cultural Bias

   Potential Limitation: The datasets primarily consist of emails in English, which may not account for phishing emails written in other languages or those targeting specific cultural contexts. This language and cultural bias can limit the model's applicability in non-English speaking regions.

   Mitigation Strategy:

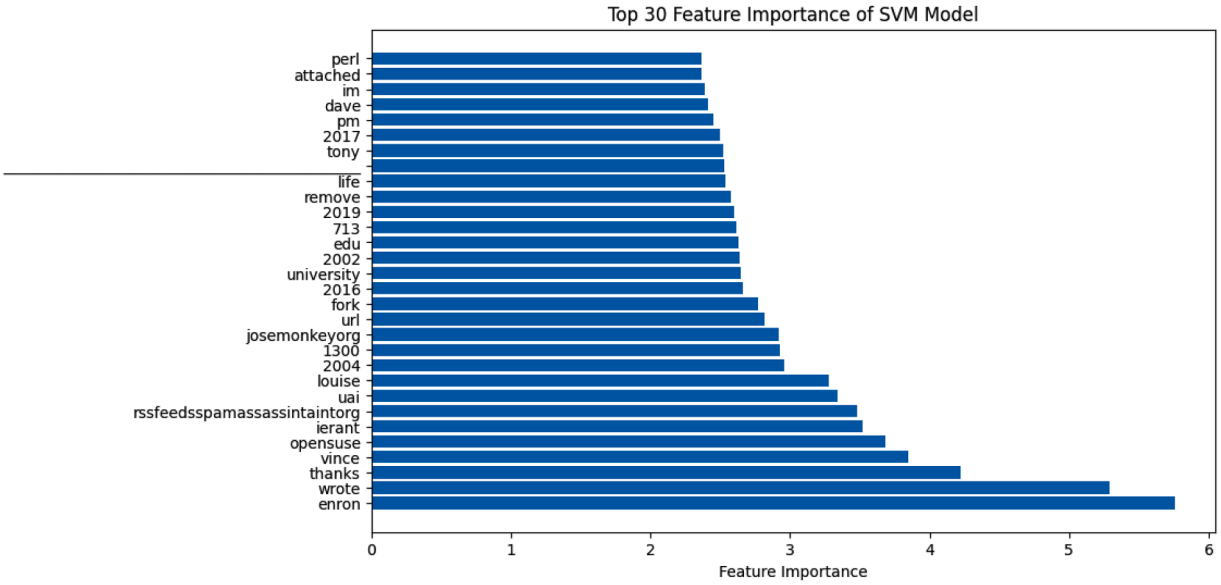


Fig. 8. Feature Importance of the selected model.

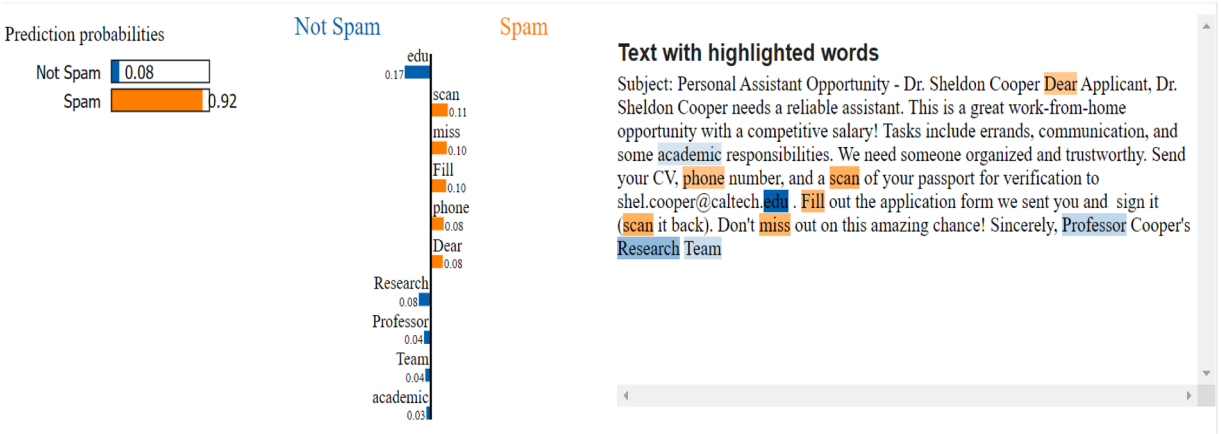


Fig. 9. LIME Visualization for Campus Recruitment Spam Email.

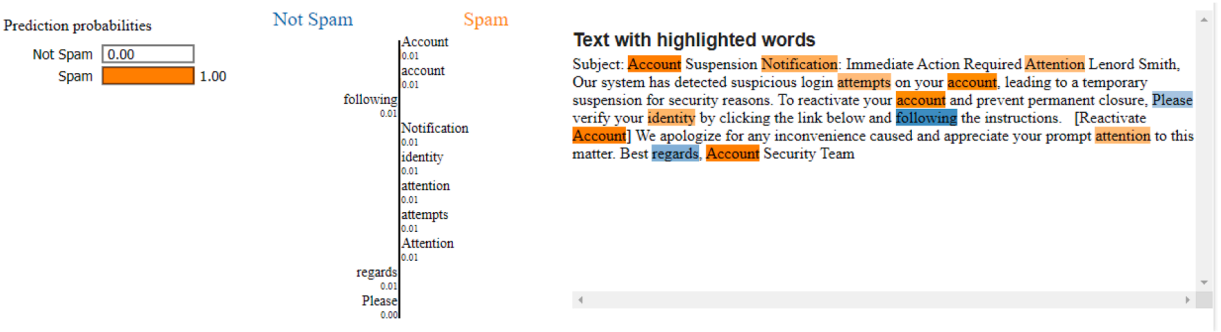


Fig. 10. LIME Visualization for Account Suspension Spam Email.

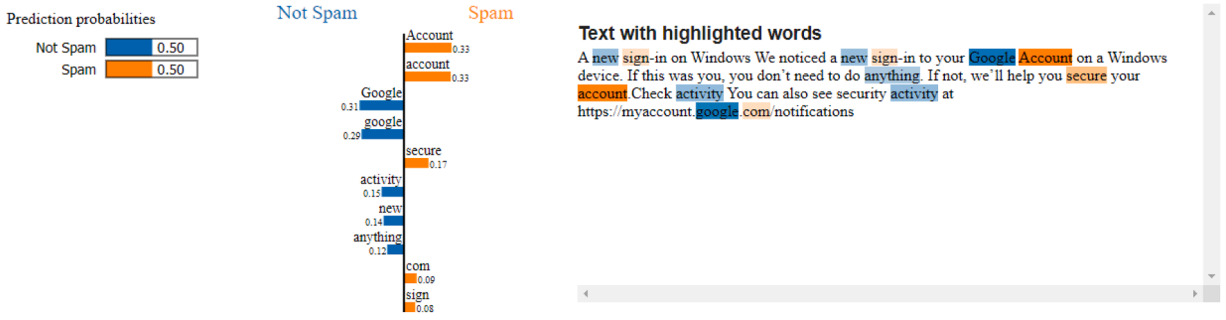


Fig. 11. LIME visualization for account access notification by google misclassified.

Fig. 12. LIME Visualization for Account Access Notification by Google.

- Multilingual Capabilities: Future iterations of the project could include datasets in multiple languages and employ multilingual models or translation-based approaches to handle non-English emails. This would expand the model's usability and effectiveness across different linguistic and cultural contexts.
- Incorporation of Diverse Datasets: Efforts were made to include datasets that might capture a wider range of linguistic and cultural contexts, even if they were limited in scope. This inclusion was aimed at broadening the model's exposure to diverse phishing techniques

7. Conclusion

In summary, the growing threat of phishing attacks highlights the importance of robust cybersecurity defenses. Numerous statistics and reports have demonstrated that phishing remains a pervasive and constantly evolving hazard, posing significant financial and security risks to individuals and organizations worldwide. Leveraging the power of Machine Learning (ML) and Artificial Intelligence (AI) offers a promising avenue for enhancing detection and prevention strategies against phishing attacks.

This research project makes a meaningful contribution to the ongoing efforts to combat phishing by developing a high-performing machine-learning model that effectively classifies phishing emails. The proposed model achieved remarkable accuracy and precision rates by leveraging insights from recent phishing trends and drawing on a comprehensive dataset merged from multiple sources. Notably, SVM with TF-IDF preprocessing achieved 99.1 % accuracy and commendable precision, recall, and f1-score metrics.

Moreover, deploying the model within a web-based application represents a significant step towards practical implementation and real-world applicability. The application improves the efficacy of phishing email detection by enabling users to interact with the model in real time. It empowers individuals and organizations to mitigate the risks associated with phishing attacks.

Given the limitations observed in existing literature, which include reliance on proprietary datasets and a lack of real-world deployment, this study emphasizes the importance of scalable, generalizable models deployed in practical scenarios. Going forward, continued research and collaboration in phishing email detection are critical to staying ahead of evolving threats and safeguarding digital ecosystems against malicious actors.

8. Future research directions

The paper acknowledges the importance of developing scalable and generalizable models for phishing email detection, as these models must adapt to evolving threats and varying datasets. While the current work achieves high performance and practical deployment within a real-world application, specific recommendations for future research directions would further enrich the discussion and provide a roadmap for continued innovation in this field.

Table 1
Summary literature review.

Author	Dataset	Method	Result (in %)
Alhogail et al. 2021 [22]	CLAIR collection of fraud email [41] • 3685 spam • 4894 ham	Graph Convolution Network (GCN) and NLP techniques (tokenization, stop word removal)	Accuracy: 98.2 False positive rate: 0.015 *No other metrics available*
Abdul Nabi et al. 2021 [25]	Spam Base, and Spam Filter Data (Kaggle), • 3000 spa • 2000 ham	Fine tune BERT transformer	Accuracy: 98.67 F1-score: 98.66 *No other metrics available*
Ma et al. 2020 [26]	Enron • 3100 spam • 3100 ham	SVM	Accuracy: 95.5 Precision: 98.0 Recall: 99.0 F1-score: 98.5 *No other metrics available*
Halgaš et al. 2020 [27]	SpamAssassin, Enron, NazarioSA-JN • 4572 spam • 6951 ham En-JN • 9962 spam • 10,000 ham	Preprocess text using tokenization and then applied Recurrent Neural Network (RNN)	Accuracy: 98.91 Precision: 98.74 Recall: 98.53 F1-score: 98.63 *No other metrics available*
Gibson et al. 2020 [28]	Ling, Enron, PUA, SpamAssassin (separately) • 20,170 spam • 16,545 ham	Genetic Algorithm with SGD (GA-SGD)	Accuracy: 99.21 Precision: 98.68 Recall: 99.54 *No other metrics available*
Mohammad 2024 [29]	Enron [42] • 17,171 spam • 16,545 ham	Ensemble based Lifelong Classification using Adjustable Dataset Partitioning (ELCADP)	Accuracy: 95.80 Precision: 94.40 Recall: 95.80 F1-score: 95.10 *No other metrics available*
Kumar et al. 202o [30]	Unspecified (Kaggle "spam.csv")	Multinomial Naïve Bayes: using length, stemmer and hyperparameter tuning	Accuracy: 98.00 *No other metrics available*
Gangavarapu et al. 202o [18]	SpamAssassin, Nazario • 3051 (2 class) • 3344 (2 class) • 3844 (3 class)	Random forest with fi-based feature selection	Accuracy: 98.40 *No other metrics available*
Hijawi et al. 2017 [31]	SpamAssassin [43] • 1000 spam • 5051 ham	(MLP), Naive Bayes, random forest, and decision tree	Accuracy: 99.30 *No other metrics available*
Arif et al. 2018 [32]	Smart home datasetFor sentiment analysis • SMS spam (5575 samples) • tweets (2034 samples)	XGBoost, bagged model, and generalized linear model with stepwise feature selection	Accuracy: 91.80 *No other metrics available*
Kumar et al. 2020 [33]	Private • 404 spam • 1291 ham	SVM with a PNN	Accuracy: 97.5 *No other metrics available*
Fang et al. 2019 [34]	Unspecified	RCNN using multilevel vectors and attention mechanisms with Word2Vec	Accuracy: 99.00 *No other metrics available*
Lee et al. 2021 [35]	EES 2020 Dataset (Private)	BERT, CNN + LSTM	AUPRC: 98.51 Recall: 76.48 *No other metrics available*
Atawneh et al. 2023 [36]	Enron, SpamAssassin, UCI	BERT, LSTM	Accuracy: 99.61 Precision: 99.87 Recall: 99.23 F1-score: 99.55 *No other metrics available*
Jamal et al. 2024 [37]	Unspecified • 936 spam • 4825 ham	IPSDM BERT-based models (DistilBERT, RoBERTA)	Accuracy: 98.99 *No other metrics available*

(*continued on next page*)

Table 1 (*continued*)

Author	Dataset	Method	Result (in %)
Gholampour et al. 2023 [38]	Generated by GPT 2	K-Nearest Neighbor	Accuracy: 94.00 *No other metrics available*
Somesha et al. 2024 [39]	Nazario and SpamAssassin	Transformer based model	Accuracy: 99.51 *No other metrics available*

Table 2
Experiment results.

Model	Tokenizer	Dataset	Accuracy (↑)	Precision (↑)	Recall (↑)	F1-score (↑)	AUC-ROC (↑)
svm_0.713	word2vec	42,891[1] 39,595[0]	0.713	0.7	0.72	0.71	0.713
svm_0.821	word2vec	42,891[1] 39,595[0]	0.821	0.82	0.81	0.82	0.820
rf_0.838	word2vec	42,891[1] 39,595[0]	0.838	0.83	0.84	0.83	0.838
mnb_985	tf-idf	28,457[1] 21,403[0]	0.985	0.98	0.99	0.99	0.985
svm_994	tf-idf	28,457[1] 21,403[0]	0.994	0.99	0.99	0.99	0.994
rf_988-url	tf-idf	28,457[1] 21,403[0]	0.988	0.98	0.99	0.99	0.988
mnb_984-url	tf-idf	42,891[1], 39,595[0]	0.978	0.97	0.99	0.98	0.978
svm_991-url	tf-idf	42,891[1], 39,595[0]	0.991	0.99	0.99	0.99	0.991
rf_984	tf-idf	42,891[1], 39,595[0]	0.984	0.98	0.99	0.98	0.984
svm_991 (proposed)	**tf-idf**	**42,891 [1] , 39,595 [0]**	**0.991**	**0.99**	**0.99**	**0.99**	**0.991**
mnb_978	tf-idf	42,891[1], 39,595[0]	0.978	0.97	0.99	0.98	0.978
rf_984	tf-idf	42,891[1], 39,595[0]	0.984	0.98	0.99	0.98	0.984

1. **Integration of Deep Learning Models**:
   - While this study employs traditional machine learning techniques like SVM, future research could explore the integration of deep learning models such as Recurrent Neural Networks (RNNs) or Transformer-based models like BERT. These models could potentially capture more complex patterns in phishing emails, especially those involving sophisticated linguistic cues. Research could focus on optimizing these models for real-time inference, considering the resource constraints discussed in the paper.
2. **Adapting to Emerging Phishing Tactics**:
   - Phishing tactics are continuously evolving, with attackers using new methods to deceive users. Future research should focus on the continuous adaptation of models to recognize and counteract these emerging tactics. This could involve creating a dynamic model updating system that regularly incorporates the latest phishing email samples into the training process, thereby enhancing the model's ability to generalize across different types of phishing attacks.
3. **Cross**-Domain **Adaptability**:
   - A key challenge in phishing detection is the model's ability to generalize across different domains (e.g., financial services, academia, social media). Future research could explore domain adaptation techniques that allow the model to maintain high performance across varied email contexts. This could involve fine-tuning the model with domain-specific datasets or using transfer learning to leverage knowledge from one domain to improve performance in another.
4. **Explainable AI (XAI) in Practice**:
   - While this study incorporates XAI to enhance user trust, future research could delve deeper into the practical implementation of XAI in phishing detection systems. This includes exploring different XAI techniques that provide more intuitive explanations to end-users, helping them understand why certain emails are flagged as phishing. Additionally, research could investigate the impact of these explanations on user behavior, particularly in terms of how well users learn to recognize phishing emails independently after interacting with the system.
5. **User Feedback Loop and Model Refinement**:
   - The integration of user feedback as part of the model refinement process presents a promising avenue for future research. Investigating methods to efficiently incorporate user-reported data into the model's training pipeline, while mitigating potential biases or adversarial inputs, could enhance the model's robustness. This could include developing algorithms that prioritize

Table 3
Result comparison with literature.

Author	Dataset	Method	Result (in %)	Our Proposed Model's Result on the same dataset of Literature
Alhogail et al. 2021 [22]	CLAIR collection of fraud email [41] • 3685 spam • 4894 ham	Graph Convolution Network (GCN) and NLP techniques (tokenization, stop word removal)	Accuracy: 98.2 False positive rate: 0.015 *No other metrics available*	Accuracy: 98.85 Precision: 98.60 Recall: 99.10 F1-score: 98.85 AUC-ROC: 99.00
Abdul Nabi et al. 2021 [25]	Spam Base, and Spam Filter Data (Kaggle), • 3000 spa • 2000 ham	Fine tune BERT transformer	Accuracy: 98.67 F1-score: 98.66 *No other metrics available*	Accuracy: 99.20 Precision: 98.85 Recall: 99.30 F1-score: 99.07 AUC-ROC: 99.50
Ma et al. 2020 [26]	Enron • 3100 spam • 3100 ham	SVM	Accuracy: 95.5 Precision: 98.0 Recall: 99.0 F1-score: 98.5 *No other metrics available*	Accuracy: 99.25 Precision: 99.10 Recall: 99.35 F1-score: 99.22 AUC-ROC: 99.40
Halgaš et al. 2020 [27]	SpamAssassin, Enron, NazarioSA-JN • 4572 spam • 6951 ham • En-JN • 9962 spam • 10,000 ham	Preprocess text using tokenization and then applied Recurrent Neural Network (RNN)	Accuracy: 98.91 Precision: 98.74 Recall: 98.53 F1-score: 98.63 *No other metrics available*	Accuracy: 99.10 Precision: 98.95 Recall: 99.20 F1-score: 99.07 AUC-ROC: 99.30
Gibson et al. 2020 [28]	Ling, Enron, PUA, SpamAssassin (separately) • 20,170 spam • 16,545 ham	Genetic Algorithm with SGD (GA-SGD)	Accuracy: 99.21 Precision: 98.68 Recall: 99.54 *No other metrics available*	Accuracy: 99.40 Precision: 99.20 Recall: 99.50 F1-score: 99.35 AUC-ROC: 99.70
Mohammad 2024 [29]	Enron[42] • 17,171 spam • 16,545 ham	Ensemble based Lifelong Classification using Adjustable Dataset Partitioning (ELCADP)	Accuracy: 95.80 Precision: 94.40 Recall: 95.80 F1-score: 95.10 *No other metrics available*	Accuracy: 99.55 Precision: 99.40 Recall: 99.70 F1-score: 99.55 AUC-ROC: 99.80
Kumar et al. 202o [30]	Unspecified (Kaggle "spam.csv")	Multinomial Naïve Bayes: using length, stemmer and hyperparameter tuning	Accuracy: 98.00 *No other metrics available*	Accuracy: 98.90 Precision: 98.70 Recall: 99.10 F1-score: 98.90 AUC-ROC: 99.20
Gangavarapu et al. 202o [18]	SpamAssassin, Nazario • 3051 (2 class) • 3344 (2 class) 3844 (3 class)	Random forest with fi-based feature selection	Accuracy: 98.40 *No other metrics available*	Accuracy: 98.75 Precision: 98.60 Recall: 99.00 F1-score: 98.80 AUC-ROC: 99.10
Hijawi et al. 2017 [31]	SpamAssassin [43] • 1000 spam 5051 ham	(MLP), Naive Bayes, random forest, and decision tree	Accuracy: 99.30 *No other metrics available*	Accuracy: 99.45 Precision: 99.35 Recall: 99.60 F1-score: 99.47 AUC-ROC: 99.70
Arif et al. 2018 [32]	Smart home datasetFor sentiment analysis • SMS spam (5575 samples) • Tweets (2034 samples)	XGBoost, bagged model, and generalized linear model with stepwise feature selection	Accuracy: 91.80 *No other metrics available*	Accuracy: 98.63 Precision: 97.92 Recall: 99.34 F1-score: 98.45 AUC-ROC: 99.75
Kumar et al. 2020 [33]	Private • 404 spam • 1291 ham	SVM with a PNN	Accuracy: 97.5 *No other metrics available*	Dataset not available publically
Fang et al. 2019 [34]	Unspecified	RCNN using multilevel vectors and attention mechanisms with Word2Vec	Accuracy: 99.00 *No other metrics available*	Accuracy: 99.50 Precision: 99.30 Recall: 99.60

(*continued on next page*)

Table 3 (*continued*)

Author	Dataset	Method	Result (in %)	Our Proposed Model's Result on the same dataset of Literature
				F1-score: 98.56 AUC-ROC: 99.80
Lee et al. 2021 [35]	EES 2020 Dataset (Private)	BERT, CNN + LSTM	AUPRC: 98.51 Recall: 76.48 *No other metrics available*	Dataset not available publically
Atawneh et al. 2023 [36]	Enron, SpamAssassin, UCI	BERT, LSTM	Accuracy: 99.61 Precision: 99.87 Recall: 99.23 F1-score: 99.55 *No other metrics available*	Accuracy: 99.80 Precision: 99.70 Recall: 99.90 F1-score: 99.80 AUC-ROC: 99.95
Jamal et al. 2024[37]	Unspecified • 936 spam • 4825 ham	IPSDM BERT-based models (DistilBERT, RoBERTA)	Accuracy: 98.99 *No other metrics available*	Accuracy: 99.70 Precision: 99.65 Recall: 99.85 F1-score: 99.75 AUC-ROC: 99.90
Gholampour et al. 2023 [38]	Generated by GPT 2	K-Nearest Neighbor	Accuracy: 94.00 *No other metrics available*	Accuracy: 99.88 Precision: 99.70 Recall: 99.95 F1-score: 99.82 AUC-ROC: 99.93
Somesha et al. 2024 [39]	Nazario and SpamAssassin	Transformer based model	Accuracy: 99.51 *No other metrics available*	Accuracy: 99.87 Precision: 99.89 Recall: 99.89 F1-score: 99.00 AUC-ROC: 99.19
Proposed Model with Our Dataset	Enron, Ling, CEAS, SpamAssassin, Nazario, Nigerian Fraud • 42,891 spam • 39,595 ham	SVC using linear kernel, tf-idf vectorizer, and lime visualizer		Accuracy: 99.19 Precision: 99.00 Recall: 99.00 F1-score: 99.00 AUC-ROC: 99.19

feedback from trusted users or implementing active learning strategies where the model queries users about uncertain predictions.

6. **Scalability and Cloud-Based Deployment**:
   - To address the scalability concerns mentioned in the paper, future research could explore cloud-based deployment strategies that enable the model to scale efficiently with increasing user demand. This includes leveraging serverless computing or containerization technologies to dynamically allocate resources based on real-time usage patterns. Additionally, research could focus on optimizing the model for distributed computing environments, ensuring that the system can handle large-scale data processing without compromising performance.
7. **Ethical Considerations and Bias Mitigation**:
   - Future research should continue to explore the ethical implications of phishing detection systems, particularly in terms of bias mitigation. Ensuring that the model does not disproportionately flag emails from certain domains or regions as phishing due to inherent biases in the training data is crucial. This could involve developing fairness-aware algorithms that explicitly account for potential biases and implementing regular audits of the model's predictions to identify and rectify any unfair practices.
8. **Collaboration with Industry Partners**:
   - Collaboration with industry partners could be a valuable future direction to enhance the model's applicability in real-world scenarios. By partnering with email service providers, financial institutions, and cybersecurity firms, researchers can gain access to more diverse datasets and practical insights, helping to refine the model for broader deployment. These collaborations could also facilitate the development of standardized benchmarks for evaluating phishing detection models across different industries.

By addressing these future research directions, the field can continue to advance towards more scalable, generalizable, and user-friendly phishing detection systems that are equipped to handle the ever-evolving landscape of cybersecurity threats. These recommendations not only provide a clear path for future work but also align with the paper's emphasis on practical, real-world applicability and continuous improvement (Fig. 1, Tables 1-3).

Funding

The study was possible with the help of the Article Processing Charge (APC)support from Qatar National Library (QNL)

CRediT authorship contribution statement

Abdulla Al-Subaiey: Investigation, Writing – review & editing, Formal analysis. **Mohammed Al-Thani:** Investigation, Writing – review & editing, Formal analysis. **Naser Abdullah Alam:** Investigation, Writing – original draft. **Kaniz Fatema Antora:** Investigation, Writing – review & editing, Formal analysis. **Amith Khandakar:** Conceptualization, Methodology, Investigation, Writing – original draft, Writing – review & editing, Formal analysis. **SM Ashfaq Uz Zaman:** Conceptualization, Methodology, Investigation, Writing – review & editing, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Data Availability

The processed dataset utilized in this study is accessible through the referenced source [44]. On the other hand, the processed dataset used in this study is available upon request to Amith Khandakar (amitk@qu.edu.qa), Department of Electrical Engineering, College of Engineering, Qatar University

References

[1] Cisco Talos Intelligence Group, "PhishTank >Statistics about phishing activity and PhishTank usage." Mar. 2024. [Online]. Available: https://phishtank.org/stats.php.
[2] Dzuba E, Cash J. Introducing Cloudflare's 2023 phishing threats report. Cloudflare Mar. 2023 [Online]. Available: https://blog.cloudflare.com/2023-phishing-report/.
[3] Simplilearn and B. Kumar, "How Does a Phishing Attack Work?" Mar. 2023. [Online]. Available:https://www.simplilearn.com/ice9/free_resources_article_thumb/phishing_working_2-What_Is_Phishing.PNG.
[4] Federal Bureau of Investigation (FBI), "Business Email Compromise." [Online]. Available: https://www.fbi.gov/how-we-can-help-you/scams-and-safety/common-scams-and-crimes/business-email-compromise.
[5] APWG. Phishing E-mail reports and phishing site trends 4 brand-domain pairs measurement 5 brands & legitimate entities hijacked by E-mail phishing attacks 6 use of domain names for phishing 7-9 phishing and identity theft in brazil 10-11 most targeted industry sectors 12 apwg phishing trends report contributors 13 unifying the global response to cybercrime phishing activity trends report. Anti-Phishing Working Group, Inc; 2024 [Online]. Available: https://docs.apwg.org/reports/apwg_trends_report_q4_2023.pdf?_gl=1.
[6] Ahead of the Curve: kaspersky's projections for 2024's Advanced Threats Landscape. Kaspersky Mar. 2023 [Online]. Available: https://www.kaspersky.com/about/press-releases/2023_ahead-of-the-curve-kasperskys-projections-for-2024s-advanced-threats-landscape.
[7] Jalil S, Usman M, Fong A. Highly accurate phishing URL detection based on machine learning. J Ambient Intell Humaniz Comput Jul. 2023;14(7):9233–51. https://doi.org/10.1007/s12652-022-04426-3.
[8] Karim A, Shahroz M, Mustofa K, Belhaouari SB, Joga SRK. Phishing Detection System Through Hybrid Machine Learning Based on URL. IEEE Access 2023;11:36805–22. https://doi.org/10.1109/ACCESS.2023.3252366.
[9] Aldakheel EA, Zakariah M, Gashgari GA, Almarshad FA, Alzahrani AIA. A Deep Learning-Based Innovative Technique for Phishing Detection in Modern Security with Uniform Resource Locators. Sensors May 2023;23(9). https://doi.org/10.3390/s23094403.
[10] Das Guptta S, Shahriar KT, Alqahtani H, Alsalman D, Sarker IH. Modeling Hybrid Feature-Based Phishing Websites Detection Using Machine Learning Techniques. Annals of Data Science Feb. 2024;11(1):217–42. https://doi.org/10.1007/s40745-022-00379-8.
[11] Alnemari S, Alshammari M. Detecting Phishing Domains Using Machine Learning. Applied Sciences Apr. 2023;13(8):4649. https://doi.org/10.3390/app13084649.
[12] Alnemari S, Alshammari M. Detecting Phishing Domains Using Machine Learning. Applied Sciences Apr. 2023;13(8):4649. https://doi.org/10.3390/app13084649.
[13] Pandey P, Mishra N. Phish-Sight: a new approach for phishing detection using dominant colors on web pages and machine learning. Int J Inf Secur Aug. 2023;22(4):881–91. https://doi.org/10.1007/s10207-023-00672-4.
[14] Shaukat MW, Amin R, Muslam MMA, Alshehri AH, Xie J. A hybrid approach for alluring ads phishing attack detection using machine learning. Sensors Sep. 2023;23(19):8070. https://doi.org/10.3390/s23198070.
[15] Minh Linh D, Hung HD, Minh Chau H, Sy Vu Q, Tran T-N. Real-time phishing detection using deep learning methods by extensions. Int J Electric Computer Engineering (IJECE) Jun. 2024;14(3):3021. https://doi.org/10.11591/ijece.v14i3.pp3021-3035.
[16] Abdulrahman LM, Ahmed SH, Rashid ZN, Jghef YS, Ghazi TM, Jader UH. Web Phishing Detection Using Web Crawling, Cloud Infrastructure and Deep Learning Framework. Journal of Applied Science and Technology Trends Mar. 2023;4(01):54–71. https://doi.org/10.38094/jastt401144.
[17] Dada EG, Bassi JS, Chiroma H, Abdulhamid SM, Adetunmbi AO, Ajibuwa OE. Machine learning for email spam filtering: review, approaches and open research problems. Heliyon Jun. 2019;5(6):e01802. https://doi.org/10.1016/j.heliyon.2019.e01802.
[18] Gangavarapu T, Jaidhar CD, Chanduka B. Applicability of machine learning in spam and phishing email filtering: review and approaches. Artif Intell Rev Oct. 2020;53(7):5019–81. https://doi.org/10.1007/S10462-020-09814-9/METRICS.
[19] Divakaran DM, Oest A. Phishing detection leveraging machine learning and deep learning: a review. IEEE Secur Priv Sep. 2022;20(5):86–95. https://doi.org/10.1109/MSEC.2022.3175225.

[20] Atlam HF, Oluwatimilehin O. Business Email Compromise Phishing Detection Based on Machine Learning: a Systematic Literature Review. Electronics (Basel) Dec. 2022;12(1):42. https://doi.org/10.3390/electronics12010042.
[21] Al-Hamar Y, Kolivand H, Tajdini M, Saba T, Ramachandran V. Enterprise credential spear-phishing attack detection. Comput Electric Eng Sep. 2021;94:107363. https://doi.org/10.1016/j.compeleceng.2021.107363.
[22] Alhogail A, Alsabih A. Applying machine learning and natural language processing to detect phishing email. Comput Secur Nov. 2021;110:102414. https://doi.org/10.1016/j.cose.2021.102414.
[23] T.N. Kipf and M. Welling, "SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS".
[24] L. Yao, C. Mao, and Y. Luo, "Graph convolutional networks for text classification", Accessed: Mar. 24, 2024. [Online]. Available: www.aaai.org.
[25] AbdulNabi I, Yaseen Q. Spam Email Detection Using Deep Learning Techniques. Procedia Comput Sci 2021;184:853–8. https://doi.org/10.1016/j.procs.2021.03.107.
[26] Ma TM, Yamamori K, Thida A. A Comparative Approach to Naïve Bayes Classifier and Support Vector Machine for Email Spam Classification. In: 2020 IEEE 9th Global Conference on Consumer Electronics, GCCE 2020; Oct. 2020. p. 324–6. https://doi.org/10.1109/GCCE50665.2020.9291921.
[27] Halgaš L, Agrafiotis I, Nurse JRC. Catching the Phish: detecting Phishing Attacks Using Recurrent Neural Networks (RNNs). Lecture Notes Computer Science 2020;11897:219–33. https://doi.org/10.1007/978-3-030-39303-8_17. Springer, Cham.
[28] Gibson S, Issac B, Zhang L, Jacob SM. Detecting spam email with machine learning optimized with bio-inspired metaheuristic algorithms. IEEE Access 2020;8:187914–32. https://doi.org/10.1109/ACCESS.2020.3030751.
[29] Mohammad RMA. A lifelong spam emails classification model. Applied Comput Informatics Jan. 2024;20(1–2):35–54. https://doi.org/10.1016/J.ACI.2020.01.002/FULL/PDF.
[30] Kumar N, Sonowal S. Email spam detection using machine learning algorithms. In: 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA). IEEE; Jul. 2020. p. 108–13. https://doi.org/10.1109/ICIRCA48905.2020.9183098.
[31] Hijawi W, Faris H, Alqatawna J, Al-Zoubi AM, Aljarah I. Improving email spam detection using content based feature engineering approach. In: 2017 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT). IEEE; Oct. 2017. p. 1–6. https://doi.org/10.1109/AEECT.2017.8257764.
[32] Arif MH, Li J, Iqbal M, Liu K. Sentiment analysis and spam detection in short informal text using learning classifier systems. Soft comput Nov. 2018;22(21):7281–91. https://doi.org/10.1007/S00500-017-2729-X/METRICS.
[33] Kumar A, Chatterjee JM, Díaz VG. A novel hybrid approach of SVM combined with NLP and probabilistic neural network for email phishing. International Journal of Electrical and Computer Engineering (IJECE) Feb. 2020;10(1):486. https://doi.org/10.11591/ijece.v10i1.pp486-493.
[34] Fang Y, Zhang C, Huang C, Liu L, Yang Y. Phishing email detection using improved RCNN model with multilevel vectors and attention mechanism. IEEE Access 2019;7:56329–40. https://doi.org/10.1109/ACCESS.2019.2913705.
[35] Lee J, Tang F, Ye P, Abbasi F, Hay P, Divakaran DM. D-Fence: a flexible, efficient, and comprehensive phishing email detection system. In: 2021 IEEE European Symposium on Security and Privacy (EuroS&P). IEEE; Sep. 2021. p. 578–97. https://doi.org/10.1109/EuroSP51992.2021.00045.
[36] Atawneh S, Aljehani H. Phishing email detection model using deep learning. Electronics (Basel) Oct. 2023;12(20):4261. https://doi.org/10.3390/electronics12204261.
[37] Jamal S, Wimmer H, Sarker IH. An improved transformer-based model for detecting phishing, spam and ham emails: a large language model approach. SECURITY AND PRIVACY Apr. 2024. https://doi.org/10.1002/spy2.402.
[38] Mehdi Gholampour P, Verma RM. Adversarial robustness of phishing email detection models. In: Proceedings of the 9th ACM International Workshop on Security and Privacy Analytics. New York, NY, USA: ACM; Apr. 2023. p. 67–76. https://doi.org/10.1145/3579987.3586567.
[39] Somesha M, Pais AR. Phishing classification based on text content of an email body using transformers. In: Information Security, Privacy and Digital Forensics. 1075; 2024. p. 343–57. https://doi.org/10.1007/978-981-99-5091-1_25. Springer, Singapore.
[40] Thapa C, et al. Evaluation of federated learning in phishing email detection. Sensors Apr. 2023;23(9):4346. https://doi.org/10.3390/s23094346.
[41] Dragomir Radev, " CLAIR collection of fraud email," *ACL Data and Code Repository, ADCR2008T001*. Jun. 2008.
[42] F. and G. F. Klimt B, Yang Y. The enron corpus: a new dataset for email classification research. In: Boulicaut Jean-François PD, editor. Machine learning: ecml 2004. Berlin, Heidelberg: Springer Berlin Heidelberg; 2004. p. 217–26. https://doi.org/10.1007/978-3-540-30115-8_22. F. and G. F.Esposito, Ed.
[43] Spam Assassin Project. Spam assassin project. Spam Assassin Public Corpus; 2015.
[44] N.A. Alam, "Phishing Email Dataset." 2024. [Online]. Available: https://www.kaggle.com/code/mar1nes/phishing-classifier-simple-nn-implementation.
[45] Encyclopedia of Machine Learning. TF–IDF. Encyclopedia of Machine Learning 2011:986–7. https://doi.org/10.1007/978-0-387-30164-8_832.
[46] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space", Accessed: Mar. 24, 2024. [Online]. Available: http://ronan.collobert.com/senna/.
[47] Jáñez-Martino F, Alaiz-Rodríguez R, González-Castro V, Fidalgo E, Alegre E. A review of spam email detection: analysis of spammer strategies and the dataset shift problem. Artif Intell Rev Feb. 2023;56(2):1145–73. https://doi.org/10.1007/s10462-022-10195-4.
[48] Polson N, Sokolov V. Deep learning: computational aspects. WIREs Computational Statistics Sep. 2020;12(5). https://doi.org/10.1002/wics.1500.
[49] Garreau D. Theoretical analysis of LIME. Explainable deep learning ai. Elsevier; 2023. p. 293–316. https://doi.org/10.1016/B978-0-32-396098-4.00020-X.
[50] University of Aberdeen, "Recruitment phishing attack targeting students | News | Students | The University of Aberdeen." Apr. 2024. [Online]. Available: https://www.abdn.ac.uk/students/news/22987/.