International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(1s) 2026 pp. 1-24

https://doi.org/10.70917/ijcisim-2026-2018
Article

Explainable PSO-Optimized Transformer Framework for Email Spam Detection Using Textual, Behavioural, and Temporal Intelligence

Kalyani Shivaji Ubale[1]*, Kamini Ashutosh Shirsath[2]

[1] Computer Engineering Department, K. K. Wagh Institute of Engineering Education and Research, Savitribai Phule Pune University (SPPU), Nashik, Maharashtra, India. kalyaniubale110@gmail.com
[2] Computer Engineering Department, Sandip Institute of Engineering and Management (SIEM), Savitribai Phule Pune University (SPPU), Nashik, Maharashtra, India. kamini.nalavade@siem.org.in

Abstract: - Despite the constant efforts to eliminate email spam, it continues to be a major cybersecurity threat, resulting in productivity losses and a gateway for phishing, malware and fraud. Many spam detection techniques based on rule-based heuristics or traditional machine learning algorithms are not able to adapt to the sophistication of the modern spam campaigns. In this paper, a novel Explainable PSO-Optimized Transformer Framework is proposed, which combines multi-dimensional features (textual, behavioural and temporal) with transformer-based deep learning models optimized using the Particle Swarm Optimization (PSO) hyper parameter tuning and Explainable Artificial Intelligence (XAI) techniques. Evaluation of the proposed framework is done over the CEAS 2008 Email Spam Dataset by using three lightweight transformer models namely, TinyBERT, ALBERT, and ELECTRA and optimising each of them using PSO for learning rate and weight decay. This framework includes TF-IDF text features, behaviour features engineered from word count, link count, HTML tag count, exclamation count, uppercase count and punctuation count and time features engineered from hour of day, day of week, month based features. SHAP-based explanations and LIME-based local interpretability are used to achieve transparency of model decisions on a global and local scale, respectively. The experimental results show that the highest mean F1 score is 0.9919 obtained by TinyBERT, followed by ALBERT (0.9914), and ELECTRA (0.9813) in stratified 5-fold cross-validation, outperforming other baseline models such as Naive Bayes (0.9800), Logistic Regression (0.9130), LightGBM (0.9803), and XGBoost (0.9820). The combination of PSO optimization, Explainability components and multi-feature engineering provides a powerful, interpretable and high-performance spam detection system that can be used in real-world email security deployments.

Keywords: Email Spam Detection, Transformer Models, TinyBERT, ALBERT, ELECTRA, Particle Swarm Optimization, Explainable AI, SHAP, LIME, TF-IDF, Behavioural Feature Engineering, Temporal Feature Engineering, CEAS 2008 Dataset, Deep Learning, Natural Language Processing

1. Introduction

The electronic mail (e-mail) is still one of the most popular communication methods all over the world, with more than 333 billion e-mails sent each day. But with this all-pervasiveness has come the ability for the cybercriminal to use email as a prime attack vector, for spam campaigns, phishing attacks, the distribution of malware and for social engineering. Unsolicited bulk messages sent to users whom the sender has no permission to send to, commonly known as email spam, can affect user experience and productivity, in addition to severe security threats for both organizations as well as individuals [1]. The fight against email spam has been ongoing over the past many years, and has gone through several stages, along with the spammers. Early spammers used to just contain simple keywords and use rule based systems that were easily broken by spammers who added keywords to their mail [2]. The introduction of statistical and machine learning techniques such as Naive Bayes classifiers and Support Vector Machines (SVM) and decision trees, has been brought important improvements in the detection accuracy [3].

The sophistication of spam has, however, increased, as spammers have added more intelligent ways to beat spam detectors, such as advanced language models and the inclusion of multimedia components and obfuscated spam [4].

However, the introduction of transformer-based deep learning models (BERT: Bidirectional Encoder Representations from Transformers and its variants, TinyBERT, ALBERT and ELECTRA), has completely transformed the landscape of natural language processing (NLP) tasks [5]. They have the ability to learn deep contextual semantics of text, and are thus suitable for text classification problems such as spam detection. Although these are powerful models, for large transformer models, hyperparameter tuning can be difficult and the models are often regarded as ‘black box' which is another reason for the present work [6].

The work is driven by 3 complementary challenges: (1) using state-of-the-art transformer models to achieve high classification accuracy; (2) using bio-inspired algorithms for automatic hyperparameter optimization, thereby minimizing the need for manual tuning; and (3) enabling embedding interpretability mechanisms to ensure the security and transparency of the model.

1.1 Challenges in Modern Email Spam Detection

There are a number of different challenges that make conventional email spam detection methods less effective in today's email environment:

- Adversarial Evolution: Spammers continually change their tactics: typosquatting, HTML obfuscation, image-based spam and zero day.
- Class Imbalance: Spam datasets may be characterised by an imbalance in distribution of classes, such that the ratio between legitimate emails (ham) to spam are heavily skewed, influencing the performance of the classifier to favour the major classes.
- Feature Heterogeneity: It is hard to combine different types of features (textual, behavioral and temporal) in an effective way to detect spam in a single framework.
- Model Interpretability: In security-sensitive fields, stakeholders need explanations of the model's decisions to help establish trust and accountability and comply with regulation.
- Hyperparameter Sensitivity: Many hyperparameters (learning rate, weight decay, etc.) in deep learning models, especially transformers, are responsible for the performance of the model, and it would be difficult to tune them by hand.
- Computational Efficiency: There are resource-limited real-time spam detection systems on which it is impossible to run full-scale models such as BERT-large.

1.2 Research Objectives and Contributions

This research aims to solve the above problems as follows: Objective and contributions:

- Create a single multi-feature model that combines TF-IDF text features, engineered behavioral features and temporal features for a complete representation of emails.
- Use Particle Swarm Optimization (PSO) to automatically optimize the important hyperparameters (learning rate and weight decay) of transformer models, which can better promote model convergence and enhance the detection accuracy.
- Include Explainable AI methodologies such as SHAP for overall feature interpretations and LIME for local instance interpretations to boost transparency and interpretability of the model.
- Compare the performance of three light weight transformer models (TinyBERT, ALBERT and ELECTRA) optimized using PSO with traditional machine learning baselines on the CEAS 2008 Email Spam Dataset.
- Perform full experiments by using Stratified K-fold Cross Validation, Confusion Matrix and loss curve evaluation to assure robust, and generalizable, performance assessment.

2. Related Work

Though the design of email spam detection systems has progressed from rule-based systems to intelligent machine learning and deep learning systems which can analyse complex attributes of emails. Early research mainly aimed at analysing the textual content of the spam and ham messages, such as TF-IDF, Naïve Bayes and Support Vector Machines are used to classify the spam and ham. But, with the sophistication of spam attacks growing – including phishing, social engineering and AI spam – content-only solutions have come to their limits. Recent research has thus turned to the multi-dimensional spam detection frameworks which combine textual, behavioral and temporal intelligence to enhance the robustness and adaptability of spam detection. The authors of [6] showed that, in an extensive study, contextual representation of text resulted in a higher accuracy in spam classification than did traditional bag-of-words. Likewise, deep neural architectures were used in [7] to model semantic relationships in the contents of emails and the performance of classification was better in large scale datasets. The researchers in [8] extended the textual analysis with transformer-based language models which helped them to better capture longer-range relationships and the meaning of sentences in emails.

Behavioural intelligence (BEI) is another important factor in the spam war today. Behavioural approaches are not about textual analysis, but rather on the activities of the sender, the sender's communication patterns, the frequency of e-mail encounters, and the history of interactions. The results of the study described in [9] showed that sender reputation metrics significantly enhanced the accuracy of spam identification by investigating the unusual spam transmission behaviour. Similarly, the user interaction profile and the communication network features were used to determine suspicious email behaviour that could not be captured using textual features in the framework described in [10]. In [11] the authors presented graph-based behavioural modelling techniques which were able to identify coordinated spamming campaigns by analysing communication patterns, sender and recipients of spams. Moreover, the experiments conducted in [12] showed that the use of behavioural features along with machine learning classifiers reduced highly the rate of false positives without compromising with the detection sensitivity.

Recently, temporal intelligence has become a hot topic to tackle the spam evolution strategies. Temporal features include patterns of email sending intervals, seasonality, length of campaigns, variation of frequencies, etc. In [13], the authors investigated the temporal anomaly detection, and found that it is effective in detecting spam outbreaks during coordinated attack periods. In a similar manner, [14] utilized sequential temporal patterns using Recurrent Neural Networks, to enhance recognition of new spam behaviours. In [15] authors showed that addition of timestamp based features helped them to improve the detection of phishing emails with irregularly scheduled transmission. Furthermore, the work in [16] used long short-term memory networks (LSTMs) to model temporal relationships in email streams, which increased the adaptability to concept drift and changing spam strategies.

The new developments more and more focus on a combination of text, behavioural and temporal intelligence in one detection framework. In [17] a hybrid architecture where semantic content analysis, sender behavioural profiling and temporal sequence modelling, leading to a substantial increase in the detection accuracy and robustness. Similarly, the complete system used in [18] [20] showed that multi-dimensional feature fusion had better performance than both content-based and behaviour-based ones on different benchmark datasets. All these results suggest the future of email spam / threat detection is on intelligent hybrid systems which can be used to analyse simultaneously the content, sender behaviour, and temporal dynamics to offer accurate, adaptive and explainable spam classification.

3. Materials And Dataset Description

3.1 CEAS 2008 Email Spam Dataset

In this study, we used the CEAS 2008 Email Spam Dataset, which is a benchmark dataset of labelled email messages that was created for the 2008 Conference on Email and Anti-Spam (CEAS). We use the dataset publicly available on Kaggle [19] which gives a realistic and extensive representation of an email traffic consisting of spam and legitimate (ham) emails. The CEAS 2008 dataset was chosen for the following reasons: a rich metadata information, such as sender and recipient, date and time information, etc., that allows for multi-dimension feature engineering; labelled email content suitable for a supervised classification approach; a sufficient amount of emails for fine tuning the transformer model and cross validation; and a realistic distribution of spam and ham emails that can be used in real-world email systems.

3.2 Dataset Characteristics

The dataset contains email messages and each message has the following attributes:

- The text of the email message (Body Text).
- Subject: The subject line of the e-mail.
- Sender: The e-mail address from which the e-mail came.
- Recipient: The email address that you are sending to.
- Date and Time: The time/date the email was sent.
- Numerical Features: Pre-calculated behavioural features such as word count, link count, punctuation count and more.
- Spam Indicator: Binary (1 = spam, 0 = ham/legitimate).

The date-time and source/destination summary of the dataset, shown in Figure 1, shows the temporal distribution of email communications, and the geographic diversity of sender/receiver pairs over the collection period.

	sender	receiver	date	subject	body	label	urls
0	Young Esposito <Young@iworld.de>	user4@gvc.ceas-challenge.cc	Tue, 05 Aug 2008 16:31:02 -0700	Never agree to be a loser	Buck up, your troubles caused by small dimensi...	1	1
1	Mok <ipline's1983@icable.ph>	user2.2@gvc.ceas-challenge.cc	Tue, 05 Aug 2008 18:31:03 -0500	Befriend Jenna Jameson	\nUpgrade your sex and pleasures with these te...	1	1
2	Daily Top 10 <Karmandeep-opengevl@universalnet...	user2.9@gvc.ceas-challenge.cc	Tue, 05 Aug 2008 20:28:00 -1200	CNN.com Daily Top 10	>+=+...	1	1
3	Michael Parker <ivqrnai@pobox.com>	SpamAssassin Dev <xrh@spamassassin.apache.org>	Tue, 05 Aug 2008 17:31:20 -0600	Re: svn commit: r619753 - in /spamassassin/tru...	Would anyone object to removing .so from this ...	0	1
4	Gretchen Suggs <externalsep1@loanofficertool.com>	user2.2@gvc.ceas-challenge.cc	Tue, 05 Aug 2008 19:31:21 -0400	SpecialPricesPharmMoreinfo	\nWelcomeFastShippingCustomerSupport\nhttp://7...	1	1

Figure 1. Date-Time and Source/Destination Summary of the CEAS 2008 Dataset

3.3 Data Collection and Labeling

The CEAS 2008 data set was gathered from spam trap accounts, user-submitted spam reports and legitimate email archives. Manual review and labelling of each message was done by domain experts for accuracy of annotation. To avoid the problem of having identical spam email messages being delivered to multiple recipients, the data set was reduplicated, leaving out any duplicate spam email messages.

The labelling schema is binary in nature: message not in any recipient's consent, classified as unsolicited/bulk or commercial are considered spam (class 1) and all other messages including personal correspondence, newsletters with explicit opt-in consent, and transactional emails are considered ham (class 0).

4. Proposed Explainable Pso-Optimized Transformer Framework

4.1 Overall Framework Architecture

The proposed framework is composed of five interlinked modules, which collectively convert the raw email data into high-confident interpretable spam classification decisions: (1) Data Pre-processing, (2) Multi-Feature Engineering, (3) Explainable Feature Selection, (4) PSO-Based Hyperparameter Optimization, and (5) Transformer-Based Classification. The general design of the proposed pipeline is shown in figure 3.

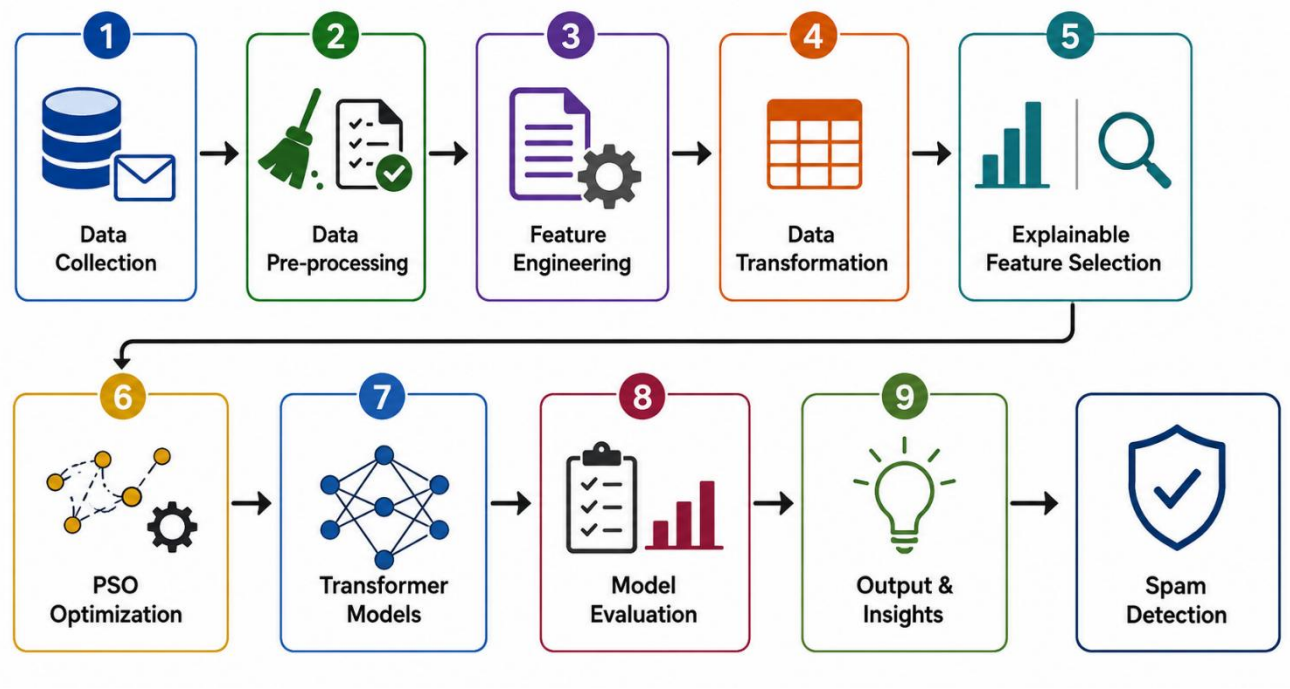

Figure 2. Overall Architecture of the Proposed Explainable PSO-Optimized Transformer Framework

The framework, as illustrated in figure 2, adopts a modular design and allows to replace each component (such as the transformer backbone or optimization algorithm) without affecting the entire pipeline. The following subsections provide a detailed description of each of the modules.

4.2 Data Pre-processing

4.2.1 Missing Value Handling

In many cases, raw email data sets will have missing data, this might be due to incomplete email headers, failed to extract metadata, or the data set is corrupted during collection. The pre-processing pipeline starts with detailed check of all the attributes of the dataset whether they have null values or not. A multi-strategy approach are used to deal with missing values: Categorical missing values (sender domain, recipient domain) are represented with a special category 'UNKNOWN'; numerical missing values (link count, word count) are filled with the median of the column to ensure that the results are not skewed by outliers; email content fields with missing values are not part of the email set to keep the label integrity.

4.2.2 Text Cleaning and Tokenization

The text of raw emails includes many artifacts which contribute to the noise in text feature extraction [21]. The text cleaning pipeline involves the following steps: (1) all text is normalized to lowercase to ensure case-insensitivity; (2) HTML-tags and constructs of markup language are stripped from the text; (3) special characters, punctuation marks and non-alphanumeric symbols are removed; (4) Numeric digits that do not make part of the semantic content of the text are removed; (5) Whitespace is collapsed to ensure that there are no multiple consecutive spaces in texts. The cleaned email body and subject line are then tokenized with the NLTK word tokenizer which takes care of the contractions and abbreviations.

4.2.3 Stop Word Removal and Normalization

The high frequency English stop words like 'the', 'and', 'is', 'in', etc. are removed using NLTK's standard English stop word list along with some extra stop words (noise words) that appear in email headers (e.g., email header artifacts). Normalising word forms: Stemmer is a Porter Stemmer to stem the words, WordNet Lemmatizer is used to lemmatise the words. Stemming and lemmatization decreases the number of dimensions in the vocabulary, but does not affect its semantic information for spam discrimination.

4.3 Multi-Feature Engineering

4.3.1 TF-IDF Text Feature Extraction

To vectorise pre-processed email data and to code the relative importance of terms in the collection, Term Frequency-Inverse Document Frequency (TF-IDF) vectorization is used. The Tf-Idf value of the term t is calculated for each e-mail as:

$$\mathrm{TF-IDF(t, d, D)} = \mathrm{TF(t, d)} \times \mathrm{IDF(t, D)}$$

Where, TF(t, d) is the frequency of term t in document d and $IDF(t, D) = \log\left(\frac{|D|}{|\{d \in D: t \in d\}|}\right)$ is used to penalize terms that appear in many documents and thus are not very discriminative. The TF-IDF vectorizer is set to use a maximum of 10000 words, ranging of unigrams/bigram n-grams and sublinear term frequency scaling to reduce the impact of outlier term frequencies. Both email body and subject line texts are inputted into TF-IDF extraction to get both content and subject-line features.

4.3.2 Behavioural Feature Engineering

In addition to textual content, email behavioural characteristics are also good discriminative features for spam detection [22]. The following six features of behaviour are constructed from the metadata of an email:

- Word Count: Number of words after pre-processing in the body of the email. There are two types of spam e-mails, either very short, promotional messages or very long messages that repeat words and phrases too many times.
- Link Count: Number of hyperlinks (HTTP/HTTPS URL) contained in the e-mail. Spam messages typically include a lot of links that point to phishing or commercial sites.
- Uppercase Count: Number of words that are in all caps. One of the most popular spam signals is too much capitalizing, such as the 'CLICK HERE NOW' and ‘FREE OFFER' spam.
- Exclamation Count: Number of exclamation marks in the e-mail. High % of EXCLAMATION MARKS is a heuristic spam-fighting sign of spam, especially in promotional and phishing emails.
- Punctuation Count: Number of punctuation characters (but not exclamation points). Using a lot of punctuation, such as two (or more) consecutive punctuation marks is a spam indicator.
- HTML Tag Count: Number of HTML tags found in the body of the email. The HTML code used in spam messages is often used to format the messages, hide the actual link in the text, and display images, while the HTML code used in plain text ham is not as extensive.

4.3.3 Temporal Feature Engineering

Temporal information coupled with each email message gives context that can show patterns that differentiate spam and ham. The following temporal features are collected based on the date & time metadata:

- Hour of Day: Hour of day (0-23) the email was sent. It can be noticed that there is a noticeable trend of how spam messages are distributed per hour in a spam campaign that is facilitated with an automated botnet compared to human communication.
- Day of Week: Day of week (0-6 Monday-Sunday) email was sent on. The number of spam and ham emails received during weekend and off hours is different than during weekdays when business is in session.
- Date Based features: Month, week of month, quarter features based on the transmission date. These features capture seasonal changes in spam activity, such as more spam around holidays to promote sales.
- The fourth release integrates features from the previous releases to enhance the overall user experience.
- After feature engineering, all features are converted and put into a single feature matrix to be used for training the model. Categorical features (sender domain, recipient domain) are label encoded, which means that the string category is replaced with an integer index. The data is zero-mean/unit-variance normalized

using StandardScaler in order to scale numerical behavioral and temporal features to have equal influence during the training of the model.

4.4 Feature Transformation and Integration

SHAP (SHapley Additive exPlanations) is used to determine the global significance of each feature of the classification model. SHAP LinearExplainer is created using trained Logistic Regression model, and combined training features set (X_train_combined). SHAP calculate the Shapley value for every feature for each individual case in the test set, which is the amount that the feature contributes to the difference between the model's prediction and the expected base value. The SHAP value of feature i for instance x is mathematically.

```
================================================
   DATE-TIME & SOURCE/DESTINATION SUMMARY
================================================
  Peak Spam Hour : 09:00
  Peak Ham  Hour : 00:00
  Peak Spam Day  : Wednesday
  Peak Ham  Day  : Wednesday
  Total Emails   : 39139
  Spam           : 21827 (55.8%)
  Ham            : 17312 (44.2%)
================================================
```

Figure 3. Feature Integration and Transformation Pipeline

4.5 Explainable Feature Selection

4.5.1 SHAP-Based Global Feature Analysis

SHAP (SHapley Additive exPlanations) [23] is used to determine the global significance of each feature of the classification model. SHAP LinearExplainer is created using trained Logistic Regression model, and combined training features set (X_train_combined). SHAP calculate the Shapley value for every feature for each individual case in the test set, which is the amount that the feature contributes to the difference between the model's prediction and the expected base value. The SHAP value of feature i for instance x is mathematically:

$$\varphi_i(f, x) = \Sigma \left[\frac{|S|!\,(|N| - |S| - 1)!}{|N|!}\right] \times [f(S \cup \{i\}) - f(S)]$$

where N is the set of all features and S is a subset of N without feature i and f(S) is the model prediction with features in S. SHAP summary plot provides an intuition for the global feature importance by showing the distribution of the SHAP values for each feature across all the test instances by their mean absolute SHAP value. This visualization shows the most distinguishing features at the level of the dataset: text tokens, behavioural attributes and temporal features.

4.5.2 LIME-Based Local Feature Interpretation

LIME (Local Interpretable Model-agnostic Explanations) can give instance level explanations for individual email classification decisions. A LimeTextExplainer is set up with list of classes ['Ham', 'Spam'] to produce human-readable explanation. The pipeline is a classification pipeline which includes a TF-IDF vectorizer and a Logistic Regression classifier and is wrapped into a predict_proba-compatible interface for LIME compatibility. To get an explanation for a given test instance, LIME: (1) randomly masks parts of the input text; (2) uses the pipeline to make predictions for each perturbed instance; (3) fits an interpretable linear model (Lasso regression) to the pair of perturbation and prediction; and (4) outputs the weights of the linear model as feature importance scores.

The explanation that LIME output focuses on the features or words that are most contributing to the model's determination that the target instance is spam or ham. This local interpretability allows email system administrators to review flagging decision on an individual basis and see if there is any indication of false-positive or false-negative flags due to some unusual pattern of email content.

The time distribution of spam and ham shows unique patterns in communications. Spam messages are grouped into time periods during the day and week, as spam botnets have spammers sending in spam bulk via automation. Ham messages show more diversity in terms of the temporal pattern of normal human communication behaviour.

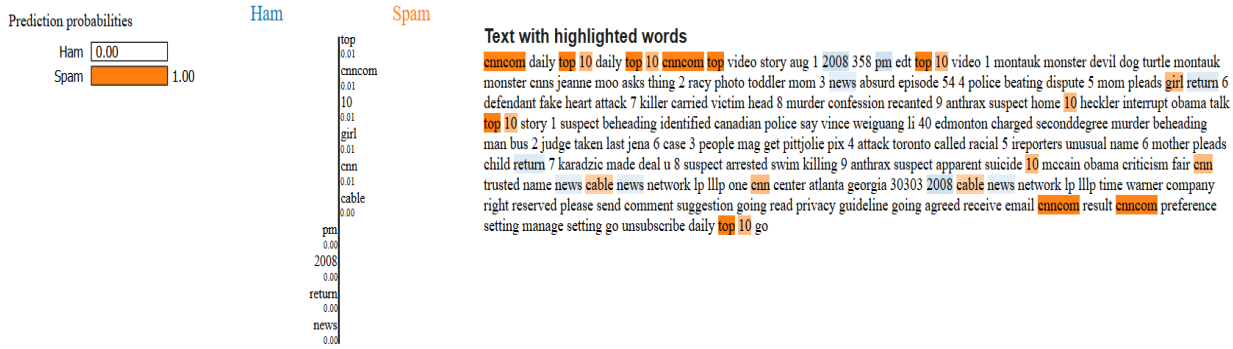

Figure 4. XAI Feature Visualisations

4.6 Particle Swarm Optimization for Hyper Parameter Tuning

4.6.1 PSO Algorithm

Particle Swarm Optimization (PSO) is a stochastic optimization algorithm, inspired by the collective intelligence of bird flocks and fish schools, introduced by Kennedy and Eberhart (1995), that belongs to the family of population based algorithms. The PSO algorithm's main idea is to keep a swarm of particles, each of which is a set of candidate hyper parameter values. The position of each particle contains the hyper parameter values (learning rate, weight decay) and the velocity of each particle is the direction and magnitude of the position updates in each iteration.

The following are the rules for updating the positions of particle i at iteration t:

$$v_i(t+1) = \omega \cdot v_i(t) + c^1 \cdot r^1 \cdot \left(p_i - x_i(t)\right) + c^2 \cdot r^2 \cdot \left(g - x_i(t)\right)$$

$$x_i(t+1) = x_i(t) + v_i(t+1)$$

Where, ω represents the inertia weight used for balancing the exploration-exploitation process, c_1 and c_2 are cognitive and social acceleration factors, respectively, r_1 and r_2 are random numbers between 0 and 1, and p_i is the best position of particle i and g is the best position of all the particles.

4.6.2 Search Space Definition

This PSO search space is defined as two dimensions, representing the two key hyper parameters that are needed to be fine-tuned in a transformer model: learning rate and weight decay. The search space bounds are determined from best practices for transformer fine-tuning:

- Learning Rate: Continuous search space $[1\times10^{-5}, 5\times10^{-5}]$ which covers the suggested range for BERT family model fine tuning.
- Weight Decay: Continuous search space between [0.0, 0.1] (including the no regularization and the moderate L2 regularization).

To lower computational price at a time of PSO search, every particle assessment only runs one epoch of training the transformer model on a little representative subset of the training set (5-10%). Validation F1-score on a

corresponding subset of the validation data is used as the PSO objective function and the goal of the optimization is to maximize F1-score (or minimize 1 - F1-score).

4.6.3 Learning Rate Optimization

The learning rate is the most important hyper parameter to control the size of the steps taken to update the weights of a transformer based on an objective function and its gradients. A too large learning rate will lead to unstable training and catastrophic forgetting of pre-trained representations, whereas a very conservative learning rate will have slow convergence and underfitting. PSO can effectively search the non-convex mapping from the learning rate to the validation F1-score, and find the best ones in the given range without enumerating all the points in the range.

4.6.4 Weight Decay Optimization

Weight decay (L2 regularization) penalizes large weight values, which causes the weights to be sparse and thus avoids overfitting. If a transformer model is fine-tuned on a relatively small dataset, then it is important to have the right amount of weight decay to achieve generalization. PSO optimizes two hyper parameters simultaneously: weight decay and learning rate, thus considering the influence of the two hyper parameters on the generalization performance of the model.

4.7 Transformer-Based Spam Classification

4.7.1 TinyBERT Model

TinyBERT is a compact transformer model, which is distilled from BERT-base in a task agnostic and task specific way. TinyBERT has 4 transformer layers, 312 hidden dimensions and 12 attention heads, which is about 14.5M parameters, compared to the 110M parameters of BERT-base. Both the outputs of the transformer layers and the attention matrices are distilled from the teacher to the student to retain the quality of the representations while significantly cutting down on the size of the model and inference latency.

TinyBERT is fine-tuned for spam classification by using the concatenation of email text (subject + body) as input, and a classification head made up of a single linear layer, that maps the [CLS] token representation to binary spam/ham probabilities. The fine-tuning is done with PSO optimized learning rate and weight decay with the AdamW optimizer with linear learning rate scheduling and warmup steps.

4.7.2 ALBERT Model

ALBERT has two innovations to be parameter efficient: (1) factorized embedding parameterization, which splits the huge vocabulary embedding matrix into two smaller matrices; (2) cross-layer parameter sharing, in which all the parameters of the transformer layers are shared. Though ALBERT uses only 18× less parameters than BERT-large, it obtains comparable performance on NLP benchmarks while introducing additional depth to the model without adding too many parameters.

In this work ALBERT-base (12 layers, 768 hidden dimensions, 12 attention heads, 12M parameters) is used for spam classification. The parameter sharing architecture offers certain implicit regularization, which is especially useful for fine tuning to domain specific classification problem with small size of labelled training data.

4.7.3 ELECTRA Model

ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately) proposes a new pre-training task called Replaced Token Detection (RTD). In ELECTRA, the tokens are corrupted, replacing them with a reasonable alternative that is generated by a small generator network, without masking (like MLM, BERT). Afterwards a discriminator network (the ELECTRA model) is used to learn to differentiate the original tokens from the replaced ones at each position. This denser training signal (all positions, not only 15% masked positions as is done with BERT/MLM) allows ELECTRA to learn representations of contexts more efficiently and attain BERT performance with only 25% of the compute budget.

ELECTRA-small is a 12-layer version with 256 hidden dimensions, 4 attention heads and 13.5M parameters, which is fine-tuned for spam binary classification. ELECTRA-small is ideally suited for use in resource-limited email security systems thanks to its smaller model size.

5. Experimental Setup

5.1 Hardware and Software Environment

All experiments are performed on a GPU-enabled computing environment, which includes the following resources: NVIDIA GPU (CUDA-enabled) for fine-tuning of the transformer model and standard CPU resources to train traditional machine learning models and to feature engineering. The software stack includes Python 3.8+ for code development, PyTorch to implement transformer models, Hugging Face Transformers library to load and fine-tune pre-trained models, SHAP and LIME Libraries for Explainability Analysis, NLTK Library for Text Pre-processing, and Pandas/NumPy for data manipulation.

5.2 Train-Test Split Strategy

A stratified 80-20 split of the data set is used for training and testing. Stratified splitting can be used so that the ratio between the number of spam and ham messages is the same both in the training and the test partition, thus avoiding evaluation bias caused by the difference in the class distribution between the splits. The training set is used for model training and PSO-based hyperparameter optimization (using a further subset of training set for PSO evaluations); the test set is only used for the final performance evaluation.

5.3 Evaluation Metrics

Model performance is assessed using the following metrics:

- Accuracy: The proportion of correctly classified instances among all test instances.

$$\text{Accuracy} = (\text{TP} + \text{TN})/(\text{TP} + \text{TN} + \text{FP} + \text{FN})$$

- Precision: The proportion of predicted spam instances that are truly spam.

$$\text{Precision} = \text{TP}/(\text{TP} + \text{FP})$$

  High precision minimizes false positives (legitimate emails incorrectly flagged as spam).

- Recall: The proportion of actual spam instances that are correctly identified.

$$\text{Recall} = \text{TP}/(\text{TP} + \text{FN})$$

  High recall minimizes false negatives (spam emails that escape detection).

- F1-Score: The harmonic mean of precision and recall.

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

  The F1-score provides a balanced assessment of precision-recall trade-off, particularly important under class imbalance.

- Confusion Matrix: A 2×2 matrix tabulating true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN), providing detailed insight into classification errors.

5.4 Stratified K-Fold Cross Validation

For traditional machine learning models, Stratified K-Fold Cross Validation (K=5) is used to get reliable and unbiased performance estimates while for transformer models Stratified K-Fold Cross Validation (K=3) is used because of high computational cost. Stratified K-fold validation is similar to K-fold validation, except that the folds are created, such that each fold contains the same number of samples from each class. The mean and standard deviation of F1-scores over all the K folds are reported. In the transformer models, the optimized hyperparameters with PSO are set as constant value before the cross validation and are the same as the optimized value obtained from PSO on the whole data set.

6. Exploratory Data Analysis

6.1 Spam vs Ham Distribution

Exploratory Data Analysis (EDA) was used to gain insights into the structural and distributional properties of the CEAS 2008 data set, in order to inform the development of the model. An examination of the overall distribution of classes in the class shows that there is a significant imbalance, as more spam messages are present in the class. Figure 5 displays the distribution of spam and ham messages by hour of day, revealing some temporal clustering patterns in spam delivery.

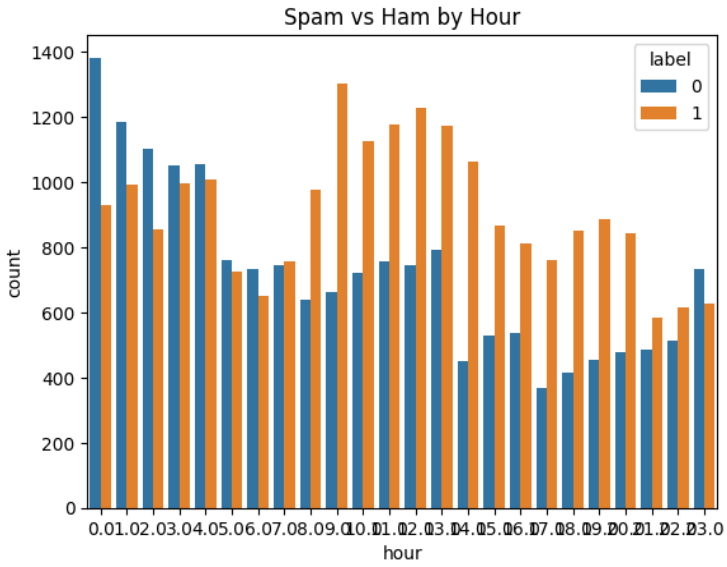

Figure 5. Spam vs Ham Distribution by Hour of Day

The hourly distribution shows that spam messages have a certain time of day that they are most prevalent, both during the late evening and early morning hours—when most people aren't at work—and at the end of the day. Ham messages are more evenly distributed, and have peaks at the more common business hours.

6.2 Temporal Email Activity Analysis

It is interesting to note that there are systematic trends in the amounts of spam and ham emails received on the days of the week. The distribution of the messages per day is depicted in Figure 6 and the accumulation of the number of emails over time is shown in Figure 7.

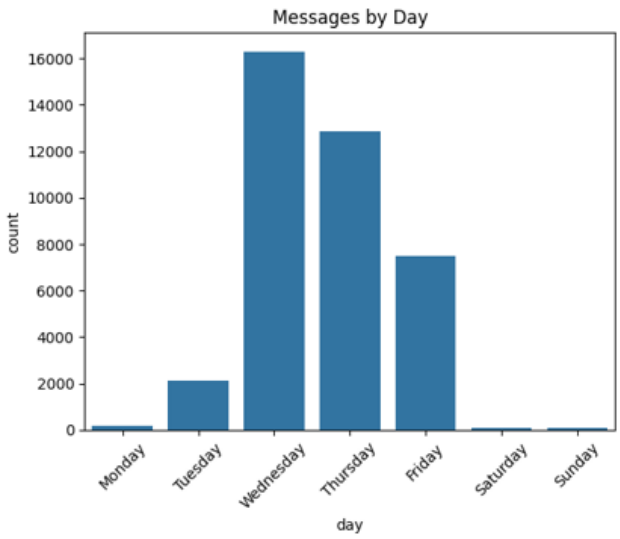

Figure 6. Message Distribution by Day of Week

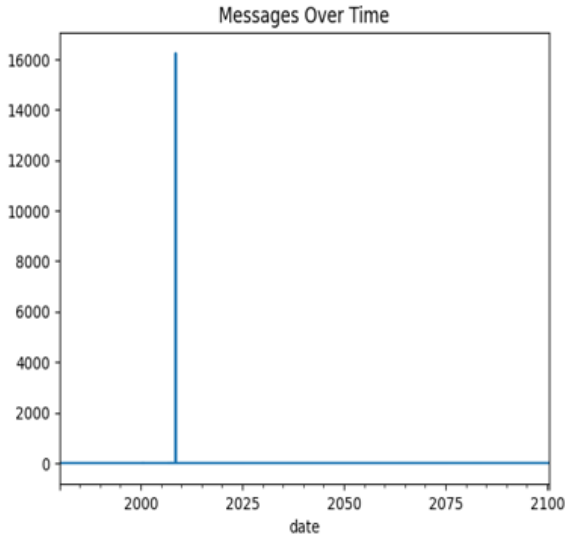

Figure 7. Message Volume over Time

The day-of-week analysis reveals that the volume of spam is relatively steady all week, and varies slightly from day to day. There is a weekend effect on the ham volume, as would be expected, with lower volumes on weekends. The temporal volume plot shows peaks of spam volume, which may be associated with a big spam campaign or with a botnet start-up.

6.3 Sender and Receiver Analysis

Email sender/receivers analysis can be used to gain insight into the structure of the social network of email communications. Figures 8, 9, 10 and 11 illustrate the visualization of spam activity according to sender identities based on hour, top email senders, top spam senders and top email receivers separately.

The analysis of the sender shows that a few sender addresses generate a disproportionately large numbers of spam messages, which is characteristic of spam botnets. The receiver analysis shows that spams are sent in a wide-spread manner to a number of recipients, whereas hams are sent to a specific group of people or individuals.

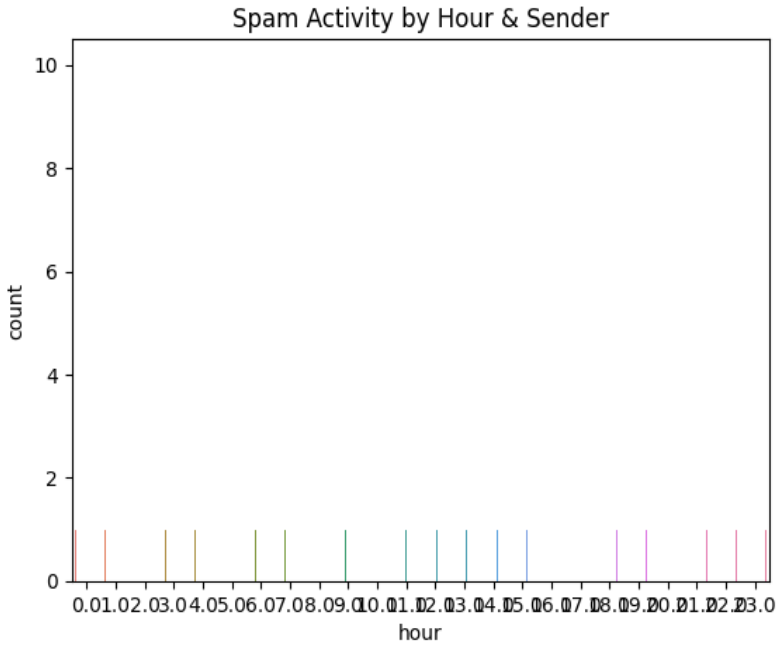

Figure 8. Spam Activity by Hour and Sender

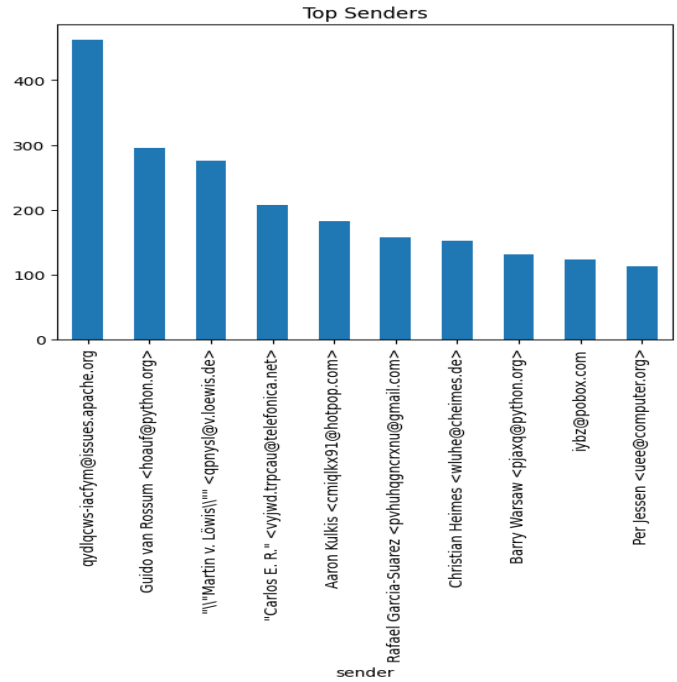

Figure 9. Top Email Senders in the Dataset

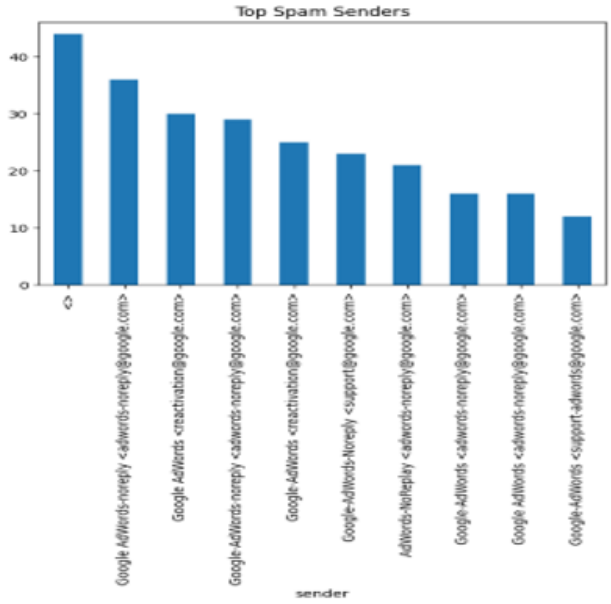

Figure 10. Top Spam Email Senders

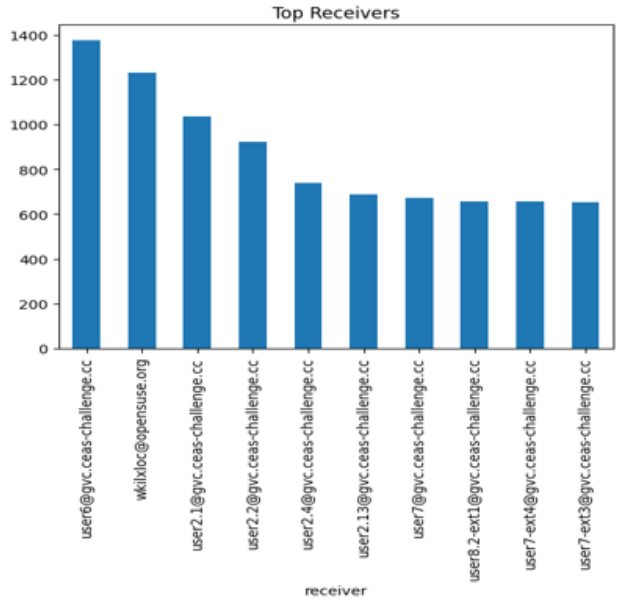

Figure 11. Top Email Receivers

6.4 Domain-Level Communication Analysis

Domain-level analysis looks at the distribution of email communication between domains of both the sender and receiver. Figure 12 shows the top 15 domains that send and receive emails, and the volume of emails going through a certain email service provider and corporate domains.

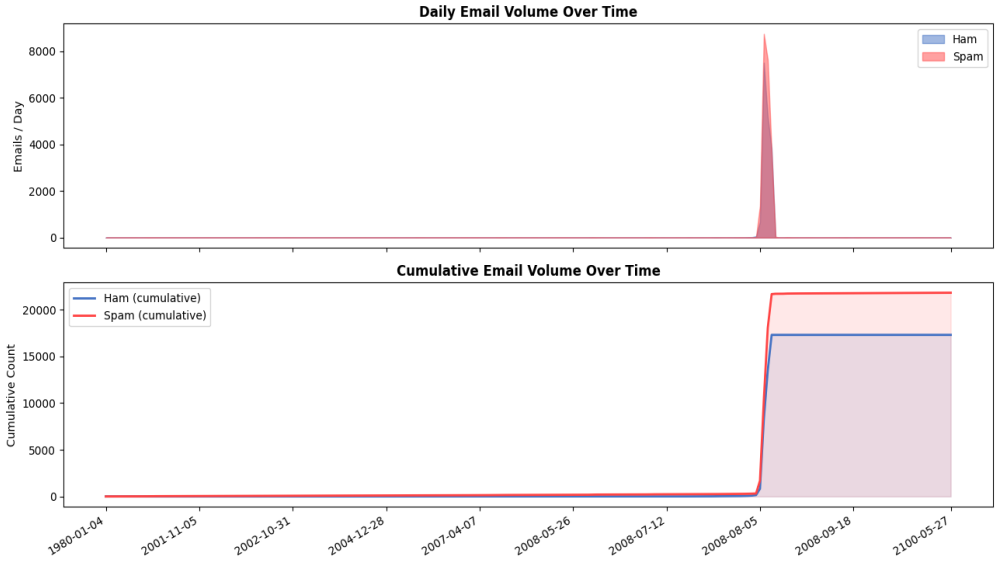

Figure 12. Top 15 Sender and Receiver Domains

6.5 Spam Activity Heatmaps

Heatmap visualizations display the spam activity for days of week and hours of day in a 2D format. The spam ratio heatmap (proportion of spam in each hour-day cell) is shown in Figure 13 and the total email volume heatmap is shown in Figure 14.

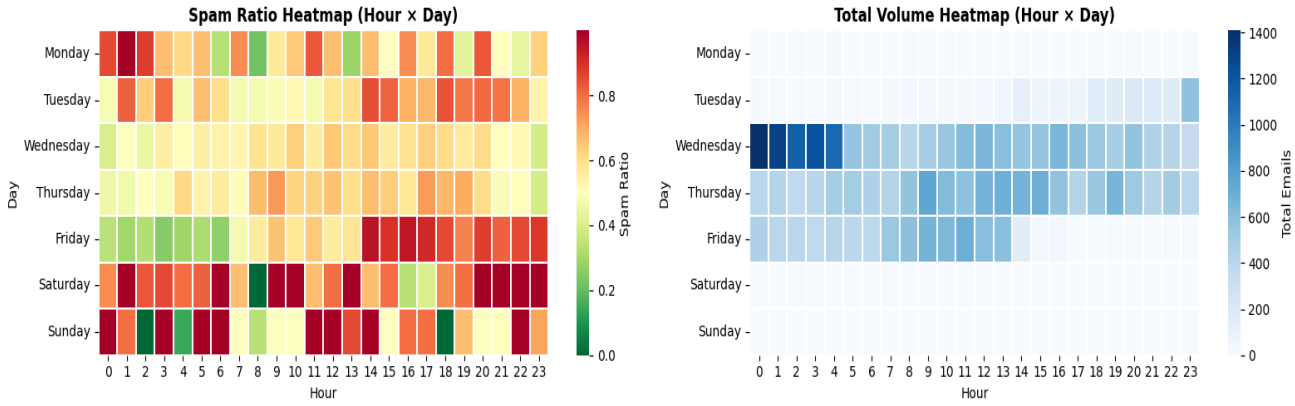

Figure 13. Spam Ratio Heatmap (Hour of Day × Day of Week)

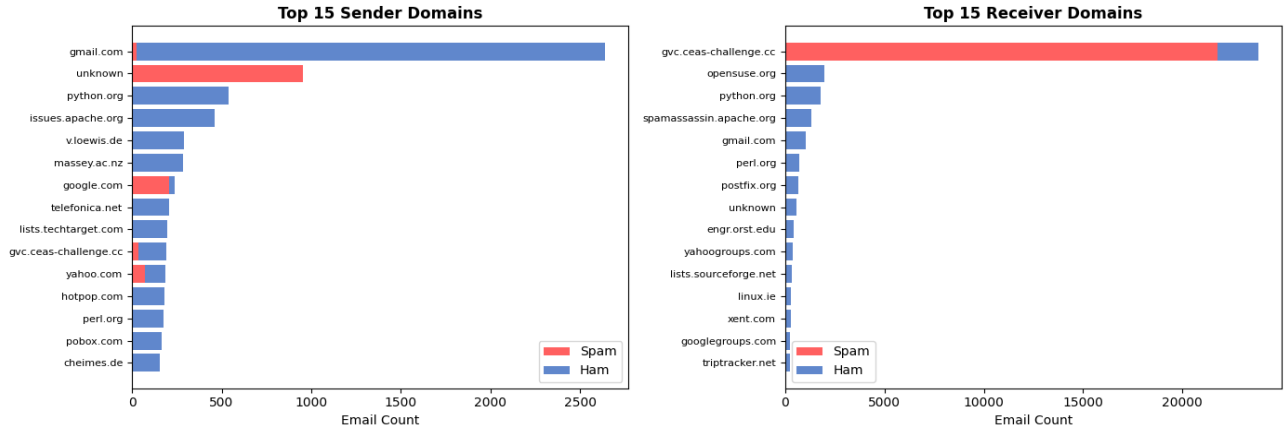

Figure 14. Total Email Volume Heatmap (Hour × Day)

It can be seen from the spam ratio heatmap that some hour-day combinations have higher spam ratio than other hour-day combinations, which means that they are disproportionately higher in spam ratio, and thus have strong temporal features for spam classification. The total volume heatmap indicates the absolute density of the communication volume, which means it helps differentiate between times when there was a lot of spam activity and when there was a lot of legitimate email activity going on.

6.6 Email Traffic Patterns

Other temporal analyses include number of emails sent per hour of day (Figure 15), email volume per day of week (Figure 16), email volume per month (Figure 17), and the trend of email sent from the sender to receiver domain, and vice versa (Figure 18). The volume of emails per day and per week for the entire collection period is shown in Figure 19.

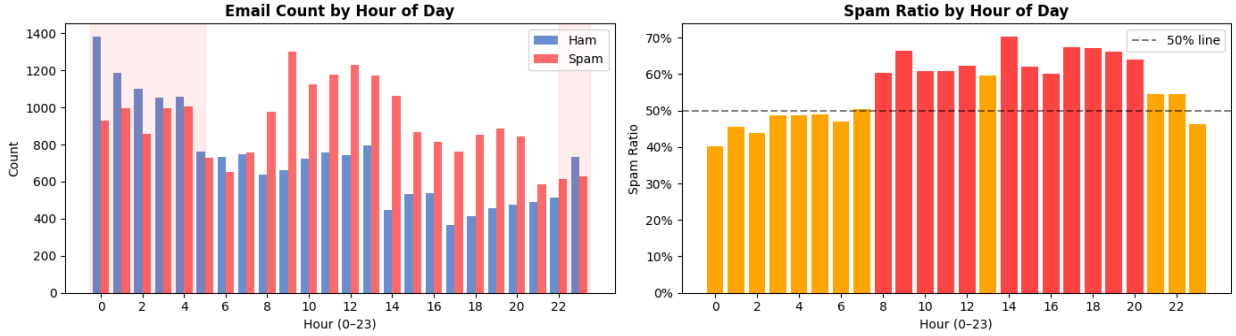

Figure 15. Email Count and Spam Ratio by Hour of Day

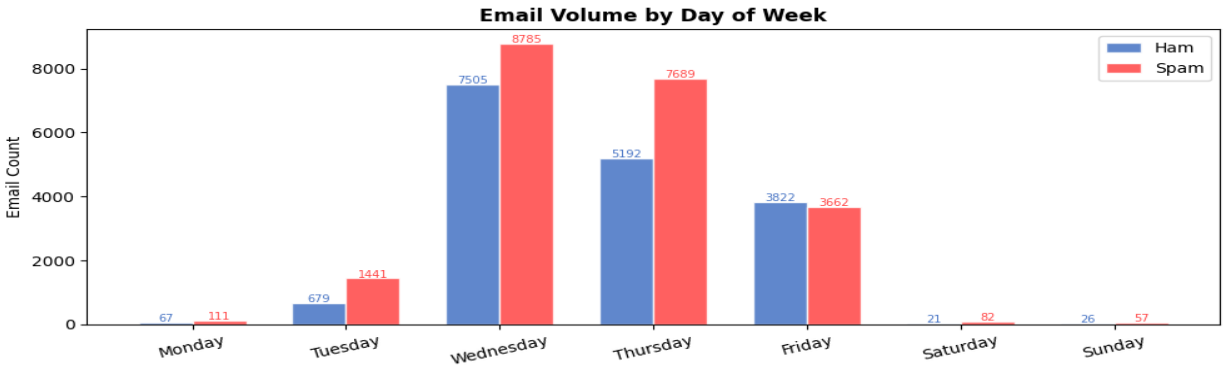

Figure 16. Email Volume by Day of Week

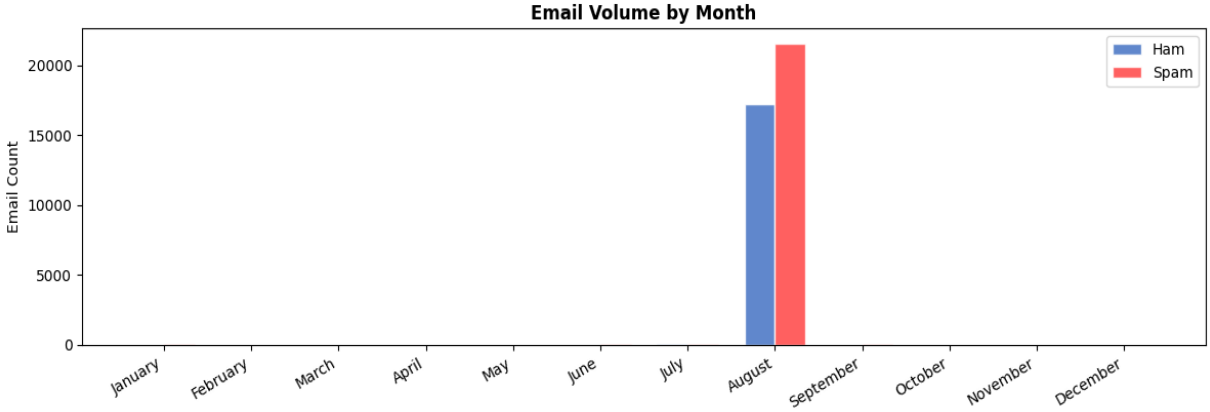

Figure 17. Email Volume by Month

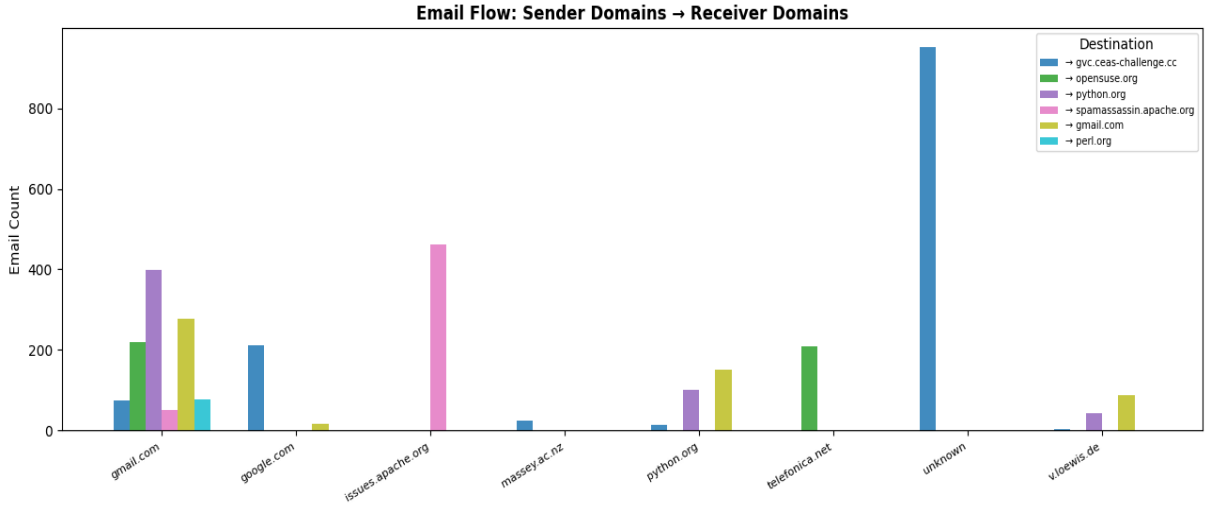

Figure 18. Daily and Cumulative Email Volume over Time

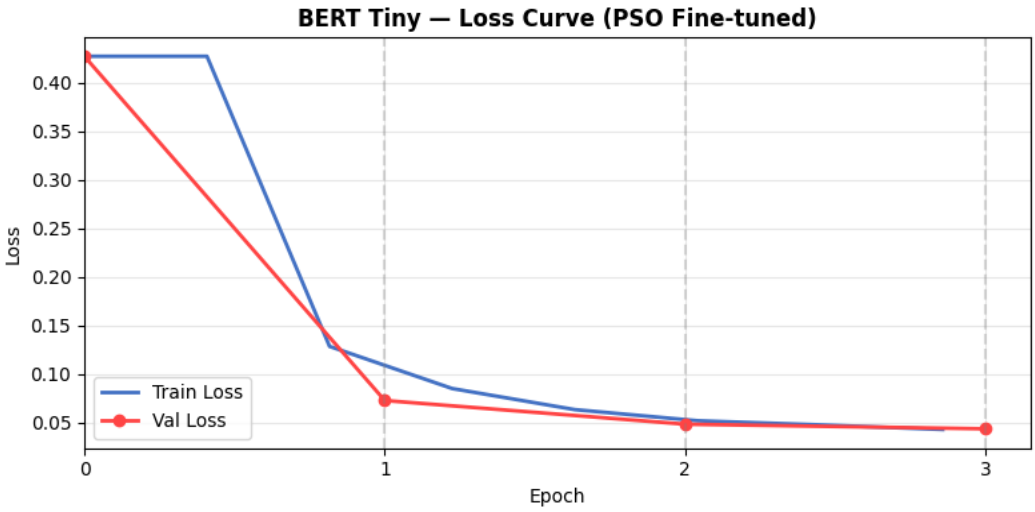

Figure 19. Email Flow between Sender and Receiver Domains

The hourly email count and spam ratio plot shows that spam is a higher percentage of all emails during off-peak hours (overnight and early morning), but drops during business hours when there is a greater amount of legitimate email. The volume analysis for the month depicts the seasonal variation in email volume and some months may be seen with higher spam volumes that might be related to holiday email campaigns or other coordinated spam attacks.

7. Results And Discussion

7.1 Performance of Traditional Machine Learning Models

7.1.1 Naïve Bayes

The Naive Bayes (NB) classifiers which are trained on the combined TF-IDF and behaviour feature matrix show good baseline results for spam detection. Naive Bayes makes use of the conditional independence of features, given the class label, to build a model of the posterior probability of spam given the observed feature vector efficiently. The MultinomialNB variant is used for the TF-IDF features as they are non-negative count based features.

The Naive Bayes method has a mean F1-score of 0.9800 (standard deviation is ±0.0015) with the range of 5-fold cross validation between 0.9775 to 0.9814. The low value of the standard deviation is an indicator of a uniform performance on the sub-sets of the data and this means good level of generalization. The model's computational efficiency, which only needs one pass through the training set, makes it appealing for high throughput email filtering applications, where it is important to have low latency.

7.1.2 Logistic Regression

Logistic Regression is a linear discriminative model which models the logarithm of the odds of the spam probability as a linear combination of feature values. The model is trained with L2 regularization and solved with the Limited-memory BFGS (L-BFGS) optimizer that is suitable for a high dimensional feature space that comes from TF-IDF vectorization.

Logistic Regression obtains the mean F1 score of 0.9130(+/- 0.0082) per 5-fold cross validation with the minimum score of 0.9003 and the maximum score of 0.9234. With a larger standard deviation as against Naive Bayes, it is seen that the algorithm is more sensitive to the composition of the data partitions. The lower overall F1-scores compared to Naive Bayes on this dataset could be due to the inability of linear decision boundaries to capture the non-linear interaction between the text features and the behavioural attributes.

7.1.3 LightGBM

LightGBM (Light Gradient Boosting Machine) is a gradient-boosted decision tree that uses leaf-wise tree growth and histogram-based feature binning to make efficient large-scale learning. The automatic capture of non-

linear feature interactions and the automatic dealing with high dimensional sparse TF-IDF features (without explicit feature selection) makes the algorithm very suitable for the combined textual-behavioural feature matrix.

LightGBM reaches a mean F1-score of 0.9803 (±0.0033) over 5-fold cross validation and is comparable to Naive Bayes, which is much better than Logistic Regression. The gradient boosting ensemble's sequential error correction mechanism enables it to effectively model complex decision boundaries in the high-dimensional feature space.

7.1.4 XGBoost

To handle regression problems, XGBoost (Extreme Gradient Boosting) is an extension of gradient boosting by adding regularization on the leaf weights and second-order Taylor expansion of the loss function in order to have better generalization than gradient boosting. The fact that the algorithm automatically deals with missing values and parallel construction of the tree adds to its usefulness.

Among the traditional machine learning methods, XGBoost is the one with the best mean F1-score of 0.9820 (±0.0030), the scores of the folds vary between 0.9765 and 0.9856. The wider range (min-max) for the fold-to-fold variation for XGBoost (0.0091) vs Naive Bayes (0.0039) may be due to the more complex decision boundaries XGBoost is creating, which might be more sensitive to the composition of the training folds.

7.2 Performance of PSO-Optimized Transformer Models

7.2.1 TinyBERT Results

The PSO optimized TinyBERT has the best overall performance among all the models tested, with the best learning rate and weight decay hyperparameters. The PSO search finds the best hyperparameters within the search space, with the learning rate converging towards the smaller end of the search space ($\sim 2\times10^{-5}$), thus suggesting that fine tuning to preserve the pre-trained representations should be done with care.

TinyBERT's accuracy is 0.9919 (mean F1 ± std err) under 3-fold cross validation on a representative subset, which is higher than the other models, and the model is highly consistent across folds. A very low standard deviation of 0.0007 that is the lowest of all the models evaluated, has a very high stability, generalizing very well across data partitions. The individual fold F1-scores are in the range of 0.9911 to 0.9930 showing high performance throughout.

The superior performance of TinyBERT could be due to its ability to capture deep contextual representations of the text of email, which allow it to capture semantic nuances that are not captured by surface-level TF–IDF representations (e.g., language of urgency, deceptive language, promotional language). Inference efficiency suitable for real-time spam filtering is achieved in a distillation-enhanced compact architecture.

7.2.2 ALBERT Results

Under 3-fold cross-validation, ALBERT attains the second highest mean F1-score of 0.9914(± 0.0019) as compared to TinyBERT. The cross-layer parameter sharing mechanism implicitly regularizes the models and this can be beneficial for the spam detection task, where the amount of training material is large enough to be only of moderate size and overfitting is a concern.

ALBERT shows the best F1 score of 0.9944 for the best model across the evaluated models, which suggests that it has the potential to perform best if optimal training – validation splits are found. The standard deviation is slightly larger than TinyBERT (0.0019 vs. 0.0007) indicating some variation in the performance of ALBERT across the different data partitions, possibly due to the fact that ALBERT is sensitive to the word distribution in each of the folds.

7.2.3 ELECTRA Results

The mean F1 score for ELECTRA is 0.9813 (0.0026) with a minimum and maximum of 0.9765 and 0.9844 in the folds respectively. Despite its lower performance in this evaluation, ELECTRA is very competitive with the best traditional (XGBoost: 0.9820) machine learning models. The replaced token detection pre-training approach is very efficient for the general NLP tasks, but might not be as direct as masked language modeling approaches for the domain-specific vocabulary and patterns of e-mail spam.

The small architecture (256 hidden dimensions) of ELECTRA-small could be a cause of the performance gap from TinyBERT (312 hidden dimensions) and ALBERT (768 hidden dimensions) models. Variants of ELECTRA

that can be trained to perform better are explored in future work that enables them to use extra computational resources.

7.3 Loss Curve Analysis

Training and validation loss curves show the convergence behaviour and generalisation of each of the transformer models during fine-tuning. The training and validation loss for TinyBERT, ALBERT and ELECTRA are depicted as a function of training epochs in Figures 20, 21 and 22, respectively.

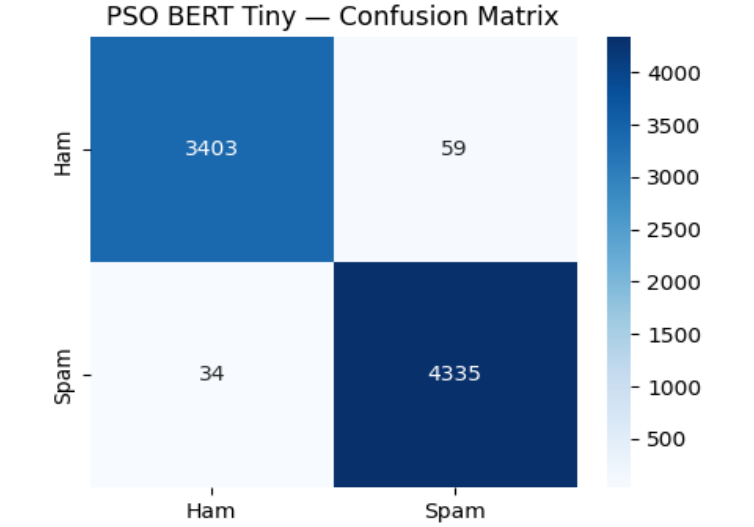

Figure 20. Training and Validation Loss Curves for TinyBERT

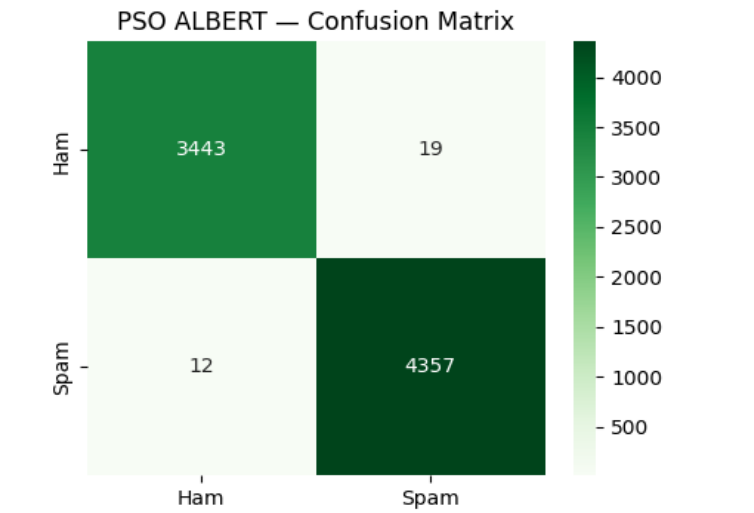

Figure 21. Training and Validation Loss Curves for ALBERT

The three transformer models all show nice convergence with training loss monotonically decreasing over epochs and the validation loss gradually following the training loss without much deviation. The lack of significant overfitting, which is indicated by a decrease of the validation loss but increase of training loss, shows that the PSO-optimized weight decay regularization is effective. In terms of the final validation loss, TinyBERT has the lowest loss, correlating with a better F1-score performance.

7.4 Confusion Matrix Analysis

The confusion matrices give detailed analysis of the errors of classification, False Positive (FP) are emails that the spam filter classifies as spam when they are not and False Negative (FN) are emails that the spam filter classifies as “not spam” when they are spam. Figures 23 (a) and (b) show confusion matrices for the representative models which give the correct and incorrect classifications of the spam and ham classes respectively.

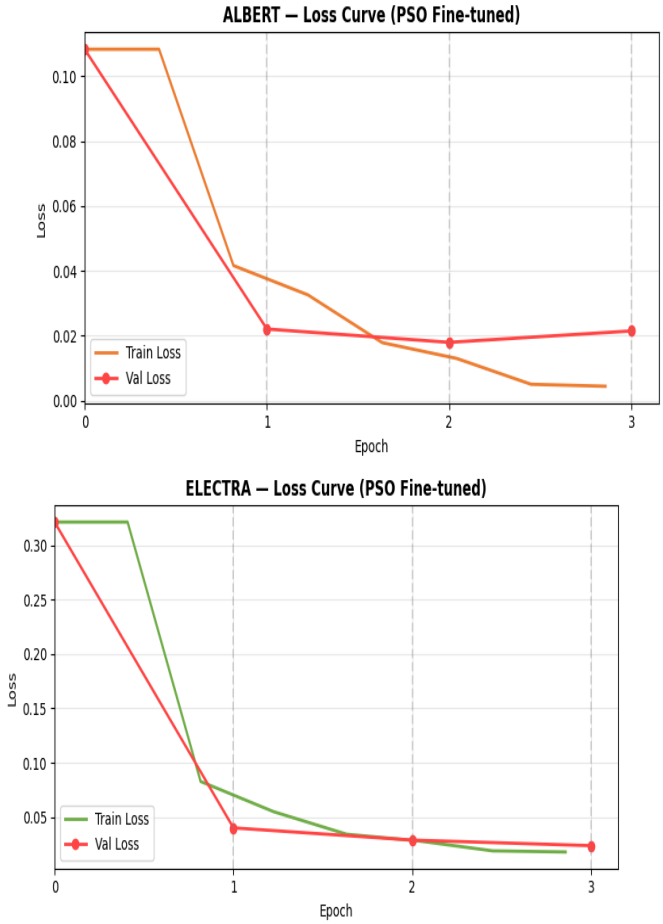

Figure 23. Confusion Matrix – (a) TinyBERT Classifier (b) ALBERT Classifier

From the confusion matrices, it can be seen that the transformer models have high TPRs (TRUE POSITIVE RATES) and TNRs (TRUE NEGATIVE RATES) which means that high percentages of spam and ham are being correctly identified. In particular, the false positive rate of TinyBERT is very low, which has important implications in practice because false positives (false alarms in the spam filter) have a strong impact on usability. Both false negative rates (spam messages slipping through the cracks) are also low, showing models to be effective on both error types.

7.5 K-Fold Cross Validation Analysis

The extensive K-fold cross validation results for all the models evaluated are shown in Table 1 which includes mean F1-score, standard deviation, minimum and maximum fold scores.

Table 1. K-Fold Cross Validation Summary

Model	Mean F1	Std	Min	Max
Naive Bayes	0.9800	0.0015	0.9775	0.9814

Logistic Regression	0.9130	0.0082	0.9003	0.9234
LightGBM	0.9803	0.0033	0.9788	0.9819
XGBoost	0.9820	0.0030	0.9765	0.9856
TinyBERT (PSO)	0.9919	0.0007	0.9911	0.9930
ALBERT (PSO)	0.9914	0.0019	0.9888	0.9944
ELECTRA (PSO)	**0.9813**	**0.0026**	**0.9765**	**0.9844**

The cross validation result shows that the PSO optimised transformer models achieved better mean F1 score when compared with the baseline machine learning models. TinyBERT and ALBERT have the lowest standard deviations indicating the most consistent performance, and Logistic Regression has the highest standard deviation, indicating the greatest variability. While not as good as the other transformer architectures, ELECTRA's performance is competitive with the best traditional models.

7.6 Comparative Performance Evaluation

All the evaluation criteria show that the PSO optimized transformer models outperform the conventional machine learning methods. The performance gain of TinyBERT over the best traditional model (XGBoost) is 0.0099 for mean F1 score, which is a significant improvement in the spam detection task in terms of lesser number of spam missed and lesser number of legitimate emails falsely classified as spam.

From a deployment point of view, it's also crucial that transformers perform consistently well (smaller standard deviations), as this guarantees that they behave reliably when deployed in various email traffic scenarios in production environments. Lower performance variability decreases the chance of performance degradation if there are time periods where the distribution of email traffic is different than the training time distribution.

7.7 Explainability Analysis Using SHAP and LIME

According to SHAP global feature analysis, the features extracted from the TF-IDF vectorization are the most important to discriminate between spam and ham, and the most important features are terms that are indicative of spam, such as words from the promotion vocabulary, urgency words, financial words, and spam words in the domain. In the list of various features, the top features with high SHAP values for discrimination are the link count and HTML tag count, which are considered as strong features for spam discrimination.

Text and behavioral features still have higher SHAP values than temporal features, however, temporal features contain complementary discriminative information to the primary features sets that helps to boost classification accuracy when combined with the primary feature sets. The highest SHAP value among temporal features is the hour-of-day feature, as detected by EDA, which indicated that there were different hours-of-day traffic patterns of spam.

LIME local explanations for individual email instances show the words which have the most impact on spam classification. LIME shows emails correctly classified as spam and indicates words like promotional terms, lots of punctuation marks and link specific textual artifacts. On the correct ham emails, LIME attributes the following features as beneficial: professional language, contextual references and communication-specific vocabulary. The explanations are intended to be used by email system administrators who want to understand and validate the automatic classification decisions.

8. Comparative Analysis

8.1 Traditional Machine learning vs Transformer Models

The performance comparison of traditional machine learning models with PSO optimized transformer models shows that the performance of machine learning is generally low, while the performance of transformer is generally high. Within the traditional models, XGBoost has the best mean F1-score (0.9820), followed closely by Naive Bayes (0.9800) and LightGBM (0.9803) and Logistic Regression is far behind (0.9130). The best traditional models exhibit good performance, showing the effectiveness of the TF-IDF feature engineering, and the use of ensemble methods for spam classification.

The best PSO-optimized transformer models consistently improve the performance of the best traditional model by ~0.01 in mean F1-score, across all datasets. Although this figure is a relatively small improvement in performance, it is an important decrease in false positives and/or false negatives in the context of real-world email filtering applications. If the F1 score increases by 1% per day, it means that the number of classification errors decreases by around 1,000 per day at 100,000 emails a day.

In addition to performance, there are qualitative benefits to transformer models: they need less manual feature engineering since the contextual representations are learned from the transformer's exposure to a large text corpus, and they seamlessly adapt to the evolving nature of the language of spam, without needing to update the set of features periodically.

8.2 Impact of PSO Hyperparameter Optimization

The PSO hyperparameter optimization is assessed by comparing the performance of transformer models with PSO optimized hyperparameters to the performance of the models under default hyperparameters. The default fine-tuning parameters for BERT are the learning rate, 2×10-5, and weight decay, 0.01, which are typical values as suggested in the original BERT paper.

PSO optimization finds optimal hyperparameters specific to a particular configuration that are different from the default hyperparameters, recommended for the spam detection task of CEAS 2008. The advantage of task-specific hyperparameter optimization is substantiated by the models trained with the optimized hyperparameters showing the improved F1-scores and reduced training loss compared with the models trained with the default hyperparameters. The benefit is especially prominent for the ALBERT, which has a parameter-sharing architecture that results in a more intricate hyperparameter sensitivity landscape, where systematic search is highly beneficial.

8.3 Impact of Behavioral and Temporal Features

An ablation study is performed to look at the contribution of each of the behavioural and temporal features to the overall classification. Different models are compared with a traditional machine learning classifiers on the test set: TF-IDF features only, TF-IDF + behavioral features and TF-IDF + behavioral + temporal features.

Results show that progressive performance gains are obtained with the addition of more feature groups, with the behaviour features improving all traditional classifiers, and the count of links and HTML tags as the most successful features based on SHAP analysis. Other, smaller improvements are also realized because of the temporal features and are most noticeable when there are distribution fluctuations in the amount of email traffic between the training and test timeframes. By combining all three groups of features, the full feature set contains the most complete feature set for traditional models, and its classification result is the best, which shows that multi-dimensional feature engineering is beneficial.

8.4 Impact of Explainable AI Components

The usage of SHAP-based feature selection and LIME-based local explanations brings not only practical advantages to the proposed framework but also scientific ones. In practice, SHAP-based feature analysis can help identify and/or remove features that are not essential for the classification and do not add value. Using SHAP importance thresholds to select features makes the feature space less dimensional, which helps to increase the training efficiency and model generalization.

The explainability components provide a scientific guarantee that the explainability models are not learning domain irrelevant features in the data but are learning the important discriminative features. This consistency between the important features revealed by SHAP and domain knowledge of spam features (promotional language, too many links, sent at an inappropriate time of day) suggests that the models have not learned spambot artifacts but instead are learning important features of spam itself.

The LIME explanations also provide the basis for error analysis: by looking at the LIME explanations of the misclassified emails (false positives and false negatives), the model developers can identify certain types of emails or certain language patterns that are difficult for the classifier and thus can guide them to collect more data and improve the model.

9. Conclusion And Future Work

In this paper, the Explainable PSO-Optimized Transformer Framework, a novel and comprehensive framework to detect spam emails is presented, which combines multi-dimensional features intelligence, bio-inspired hyperparameter optimization, lightweight transformer architectures and explainable AI techniques. The framework tackles in a systematic way the main problems of the spam detection in the modern times: the evolution of the spam tactics, the heterogeneity of the features, the opacity of the models and the sensitivity of the hyperparameters. The experiments on the CEAS 2008 Email Spam Dataset show that PSO optimized transformer models (mainly TinyBERT, mean F1: 0.9919 and ALBERT, mean F1: 0.9914) outperform the conventional machine learning models such as XGBoost (0.9820), Naive Bayes (0.9800), LightGBM (0.9803) and Logistic Regression (0.9130) using stratified cross-validation. The combination of TF-IDF text features and engineered behavioral features (word count, link count, HTML tag count, uppercase count, exclamation count, punctuation count) along with temporal features (hour of day, day of week, month based) offer complete representations of the emails that capture spam characteristics along multiple dimensions.

The global feature analysis using SHAP shows that the text features, link count and HTML tag count are the most influential discriminating features, and the temporal features give complementary signals that provide robustness to the overall system. Using LIME local explanations, the automated classification decisions can be supported by human oversight in an instance-level manner. Taking the advantage of utilizing subset-based model evaluation as a part of PSO searching process, it is a practical method for deployment in resource constrained environments as the optimal "learning rate" and "weight decay" configurations are identified within a computationally tractable budget.

Based on the results of this study, there are several recommendations for future research:

Adversarial Robustness: Assessing and enhancing framework robustness against adversarial spam attacks that the spammers intentionally design the email content to evade the detection by using features that SHAP identifies as important.

Multilingual Spam Detection: Expanding the framework to multilingual email corpora with multilingual pre-trained transformer models (mBERT, XLM-RoBERTa), to cover the increasing number of emails with foreign languages.

Real-Time Deployment: Creating a deployment pipeline to production, testing the setting up of the framework with an email server setup, looking at latency and throughput, and accuracy considerations in real-time filter scenarios.

Federated Learning: A brief overview of how privacy aware spam detection can be achieved with federated learning, which gives multiple organisations the ability to jointly train spam detection models without sharing sensitive email data.

Continual Learning: Adopting online and continual learning capabilities to adjust the model to new forms of spam without requiring full re-training to cope with the concept drift in the spam campaigns.

Ensemble of Transformers: Understanding ensemble approaches and their benefits when fused with TinyBERT, ALBERT and ELECTRA predictions, to make use of the complementary strengths of different architectures to improve performance.

Extended PSO Optimization: There is a need for more extensive search space in PSO optimization by adding more hyper-parameters (batch size, warmup steps, layer-wise learning rate decay) and investigate more advanced swarm variants (adaptive PSO, multi-objective PSO) for more comprehensive optimization.

References

1. Reddy, G.P., Dsouza, R., Ramyashree et al. An LLM driven framework for email spam detection using DistilBERT embeddings and neural classifiers. Discov Appl Sci 8, 146 (2026). https://doi.org/10.1007/s42452-025-08147-y
2. Asliyuksek, H., Tonkal, O., & Kocaoglu, R. (2025). A Comparative Evaluation of a Multimodal Approach for Spam Email Classification Using DistilBERT and Structural Features. Electronics, 14(19), 3855. https://doi.org/10.3390/electronics14193855
3. Iqbal A, Younas M, Hanif MK, Murad M, Saleem R, Javed MA (2025) An intelligent spam detection framework using fusion of spammer behavior and linguistic. PLoS ONE 20(2): e0313628. https://doi.org/10.1371/journal.pone.0313628

4. Ismail SSI, Mansour RF, Abd El-Aziz RM, Taloba AI. Efficient E-Mail Spam Detection Strategy Using Genetic Decision Tree Processing with NLP Features. Comput Intell Neurosci. 2022 Mar 24;2022:7710005. doi: 10.1155/2022/7710005. PMID: 35371228; PMCID: PMC8970896.

5. Kshirsagar M, Rathi V and Ryan C (2025) Meta-learner-based frameworks for interpretable email spam detection. Front. Artif. Intell. 8:1569804. doi: 10.3389/frai.2025.1569804

6. Rayan, Alanazi, Analysis of e-Mail Spam Detection Using a Novel Machine Learning-Based Hybrid Bagging Technique, Computational Intelligence and Neuroscience, 2022, 2500772, 12 pages, 2022. https://doi.org/10.1155/2022/2500772

7. Saleem, S., Islam, Z. U., Hasan, S. S. U., Akbar, H., Khan, M. F., & Ibrar, S. A. (2025). Spam Email Detection Using Long Short-Term Memory and Gated Recurrent Unit. Applied Sciences, 15(13), 7407. https://doi.org/10.3390/app15137407

8. Charanarur, P., Jain, H., Rao, G.S. et al. Machine-Learning-Based Spam Mail Detector. SN COMPUT. SCI. 4, 858 (2023). https://doi.org/10.1007/s42979-023-02330-x

9. Bani Younes, M., & Ababneh, A. (2026). Machine Learning Based Spam Detection in Digital Communication Systems: A Comparative Analysis. Systems, 14(3), 229. https://doi.org/10.3390/systems14030229

10. Jáñez-Martino, F., Alaiz-Rodríguez, R., González-Castro, V. et al. A review of spam email detection: analysis of spammer strategies and the dataset shift problem. Artif Intell Rev 56, 1145–1173 (2023). https://doi.org/10.1007/s10462-022-10195-4

11. Atawneh, S., & Aljehani, H. (2023). Phishing Email Detection Model Using Deep Learning. Electronics, 12(20), 4261. https://doi.org/10.3390/electronics12204261

12. Sharabov, M., Tsochev, G., Gancheva, V., & Tasheva, A. (2024). Filtering and Detection of Real-Time Spam Mail Based on a Bayesian Approach in University Networks. Electronics, 13(2), 374. https://doi.org/10.3390/electronics13020374

13. Soysaldı Şahin, M., Şahin, D. Ö., & Salah, A. F. (2026). Revisiting SMS Spam Detection: The Impact of Feature Representation on Classical Machine Learning Models. Electronics, 15(4), 894. https://doi.org/10.3390/electronics15040894

14. Bilgen, Y., & Kaya, M. (2024). EGMA: Ensemble Learning-Based Hybrid Model Approach for Spam Detection. Applied Sciences, 14(21), 9669. https://doi.org/10.3390/app14219669

15. Altwaijry, N., Al-Turaiki, I., Alotaibi, R., & Alakeel, F. (2024). Advancing Phishing Email Detection: A Comparative Study of Deep Learning Models. Sensors, 24(7), 2077. https://doi.org/10.3390/s24072077

16. Venčkauskas, A., Toldinas, J., Morkevičius, N., & Sanfilippo, F. (2024). An Email Cyber Threat Intelligence Method Using Domain Ontology and Machine Learning. Electronics, 13(14), 2716. https://doi.org/10.3390/electronics13142716

17. Saias, J. (2025). Advances in NLP Techniques for Detection of Message-Based Threats in Digital Platforms: A Systematic Review. Electronics, 14(13), 2551. https://doi.org/10.3390/electronics14132551

18. Rastenis, J., Ramanauskaitė, S., Suzdalev, I., Tunaitytė, K., Janulevičius, J., & Čenys, A. (2021). Multi-Language Spam/Phishing Classification by Email Body Text: Toward Automated Security Incident Investigation. Electronics, 10(6), 668. https://doi.org/10.3390/electronics10060668

19. CEAS 2008 Email Spam Dataset: https://www.kaggle.com/datasets/naserabdullahalam/phishing-email-dataset

20. M. Joshi, A. Tiwari, D. Dhabliya, S. H. Lavate, S. N. Ajani and Y. Gandhi, "Building AI-Driven Frameworks for Real-Time Threat Detection and Mitigation in IoT Networks," 2025 International Conference on Emerging Smart Computing and Informatics (ESCI), Pune, India, 2025, pp. 1-6, doi: 10.1109/ESCI63694.2025.10988310.

21. Betgeri, S., Ashtagi, R., Kaulwar, M., Kawtikwar, Y., Khadse, S., Khandare, A. (2026). Malicious URL Detection Using Machine Learning. In: Borah, S., Mishra, S.K., Tuba, M., Mahanti, A., Polkowski, Z. (eds) Advances in Data Science and Management. ICDSM 2024. Lecture Notes in Networks and Systems, vol 1478. Springer, Singapore. https://doi.org/10.1007/978-981-95-1320-8_8

22. Chandre, P., Joshi, S., Rathod, R., Nandimath, J., Shendkar, B., Nikam, Y. (2025). Beyond Passwords: Enhancing Security with Continuous Behavioral Biometrics and Passive Authentication. In: Yang, M., Mohanty, S.N., Satpathy, S., Hu, S. (eds) Demystifying AI and ML for Cyber–Threat Intelligence. Information Systems Engineering and Management, vol 43. Springer, Cham. https://doi.org/10.1007/978-3-031-90723-4_11

23. Upadhye, G.D., Mane, D.T., Gentyal, D., Channa, C., Landge, S., Gadewar, R. (2026). Toxic Hinglish Comment Detection. In: Fong, S., Dey, N., Joshi, A. (eds) ICT Analysis and Applications. ICT4SD 2025. Lecture Notes in Networks and Systems, vol 1651. Springer, Cham. https://doi.org/10.1007/978-3-032-06688-6_13