

Real-time Spam Email Filtering with Stacking and Majority Voting Ensembles

Dharmaraj R. Patil

Department of Computer Engineering, R. C. Patel Institute of Technology, Shirpur, India

E-mail: dharmaraj.patil@rcpit.ac.in

ORCID iD: <https://orcid.org/0000-0001-7634-2769>

Received: January 27, 2025; Revised: May 24, 2025; Accepted: September 2, 2025; Published: August 8, 2026

Abstract: Spam emails continue to be a major issue in digital communication, hurting productivity and jeopardizing security. This paper presents a real-time spam email filtering method that combines ensemble learning techniques (stacking and majority voting) with machine learning classifiers such as Logistic Regression (LR), Support Vector Machine (SVM), Stochastic Gradient Descent (SGD), and Decision Trees (DT). The objective is to improve the accuracy and reliability of spam detection systems. The proposed models were tested against Kaggle's Spam Email Classification Dataset on crucial performance parameters such as accuracy, precision, recall, F-measure, false positive rate (FPR), and false negative rate (FNR). The results show that, while the stacking ensemble is competitive, the majority voting ensemble consistently achieved good results. It delivers improved accuracy of 98.59%, precision of 98.59%, recall of 98.59%, and F-measure of 98.59% while keeping a low FPR of 0.019 and FNR of 0.009, making it the most dependable option for real-world use. This study demonstrates the efficiency of ensemble learning, particularly majority voting, in building scalable and practical solutions for real-time spam email filtering. The findings lay a solid foundation for enhancing email security mechanisms and user experience.

Index Terms: Email Spam Detection, Ensemble Learning, Stacking Ensemble, Majority Voting Ensemble, NLP, Machine Learning Models.

1. Introduction

Email is a key communication medium across business, education, healthcare, governance, and personal domains, valued for its accessibility and speed. However, its widespread use makes it vulnerable to spam—unsolicited and often malicious messages—that form a large portion of global email traffic [1, 2]. Beyond cluttering inboxes, spam poses cybersecurity risks such as phishing, malware, and ransomware, highlighting the need for adaptive and reliable spam filters. Early spam detection relied on blocklists, rule-based methods, and heuristics [3, 4], but these static approaches could not counter evolving spam tactics, prompting the use of machine learning (ML) methods like Naïve Bayes, decision trees, and support vector machines, which adaptively learn discriminative patterns.

Deep learning (DL) further enhanced spam detection through models such as CNNs and RNNs, enabling automatic feature extraction and semantic understanding. Challenges remain in scalability, computational efficiency, and resilience to adversarial spam [5]. Real-world environments require processing billions of emails daily while maintaining accuracy, and spammers continuously adapt, causing concept drift. Models must therefore be accurate, adaptive, scalable, and robust across multilingual and heterogeneous datasets. Recent studies confirm the persistent dominance of spam, underscoring the need for hybrid solutions [6, 7, 8]. Ensemble learning, combining multiple classifiers, addresses these challenges by improving robustness and minimizing false positives and negatives while maintaining efficiency.

- In 2023, about 45.6% of emails were spam, slightly lower than previous years.
- Total spam volume increased, exceeding 160 billion emails daily in 2023.
- As of December 2024, China and the U.S. were top spam sources, each generating 7.8 billion emails per day.
- Spam poses serious threats; in 2023, 94% of organizations reported phishing attacks, a 40% increase from the prior year.

Single-model filters struggle to adapt to evolving spam strategies, reducing real-world detection accuracy. Ensemble methods, particularly stacking and majority voting, overcome these limits by leveraging multiple classifiers. Stacking ensembles use outputs from base models as input to a meta-classifier to capture complementary patterns, while

majority voting ensembles rely on collective agreement to reduce bias. This study integrates stacking and majority voting ensembles for real-time spam filtering, achieving a balance between accuracy and computational efficiency. The proposed framework improves precision, reduces false positives, and maintains robust performance under dynamic and adversarial spam conditions. The major contributions of this research are summarized below.

- This paper proposes a real-time spam email filtering framework using ensemble learning (stacking and majority voting) with classifiers including LR, SVM, SGD, and Decision Trees to enhance detection reliability.
- Evaluation on Kaggle's Spam Email Classification Dataset ensures realistic assessment, leveraging diverse features and balanced spam/legitimate email representation.
- Performance is measured via accuracy, precision, recall, F-measure, FPR, and FNR. Results show majority voting outperforms stacking in overall effectiveness, reducing misclassification.
- Combining stacking and majority voting addresses high false positives and adapts to evolving spam patterns, demonstrating the strength of ensemble approaches.
- The framework is scalable and adaptable for real-world email systems, offering practical guidance for next-generation spam filters facing dynamic threats.

The remainder of this paper is organized as follows: Related Works: This section presents a comprehensive review of existing research in spam email detection, ensemble learning techniques, and relevant hybrid approaches. Methodology: We detail the proposed framework, including the design of the stacking and majority voting ensemble, preprocessing and feature extraction methods, and classifier configurations. Experimental Results and Analysis: The performance of the proposed model is evaluated quantitatively and qualitatively on benchmark email datasets, with a discussion of key findings and comparative insights. Limitations: This section highlights the constraints and potential challenges of the proposed approach. Finally, Conclusions summarize the study and its contributions, and suggest directions for future research in real-time spam filtering.

2. Related Works

Spam email filtering has been the focus of substantial study over the past two decades, indicating the rising importance of combatting unwanted emails. To solve this issue, a wide range of solutions have been developed and deployed, from basic rule-based systems to powerful machine learning models. This section examines some of the more significant approaches and their contributions to the discipline.

Tusher et al. highlight the central role of email in business, education, and other domains. The rapid increase in spam wastes resources, lowers productivity, and raises risks like phishing and data breaches. Traditional methods such as blocklists and blackhole listings struggle to keep up with evolving spam. While ML and DL improve

detection, challenges remain in real-time adaptability, efficiency, and scalability. The study emphasizes the need for lightweight models that adapt dynamically to emerging spam behaviors [1].

Emmanuel Gbenga Dada et al. underline the rising importance of effective anti-spam filters given the increasing volume of spam emails. They provide a comprehensive survey of ML-based spam filtering methods used by major service providers such as Gmail and Outlook. While these approaches show promise, the study highlights persistent issues related to scalability, adaptability to new spam patterns, and real-time detection. The authors propose that deep learning and adversarial learning could help address these shortcomings, but stress that further research is needed to balance detection accuracy with computational cost [2].

Francisco Jáñez-Martino et al. focus on the growing sophistication of spam, which increasingly carries phishing, malware, and fraudulent content. They identify two pressing challenges: (a) dataset shifts caused by the dynamic evolution of spam, and (b) adversarial techniques used by spammers to evade detection. Their experiments show that ignoring dataset shifts can result in performance drops as high as 48.81%. The authors highlight the need for adaptive and resilient spam filters capable of maintaining high accuracy under adversarial and rapidly evolving conditions [3].

Naeem Ahmed et al. investigate the dual challenges of spam in both email systems and IoT-based communication. They analyze ML and DL models such as Naïve Bayes, decision trees, and neural networks, noting that these algorithms achieve reasonable accuracy but are constrained by scalability and resource limitations. Importantly, they stress that IoT platforms introduce unique constraints due to limited computational power. The identified research gap is the lack of lightweight yet accurate spam filtering frameworks designed specifically for IoT integration [4].

Panem Charanarur et al. present a unique angle by linking spam detection with browser forensics. Their study examines artifacts like cookies, caches, and super cookies in popular browsers to identify spam-related activities. Using ML algorithms such as K-Nearest Neighbor (KNN) and Naïve Bayes (NB), they achieve promising accuracy. However, the authors acknowledge limitations in addressing false positives and in scaling their approach for diverse datasets. The research gap lies in integrating forensic analysis with robust ML-based spam filtering systems that can handle real-time detection effectively [5].

Chira N. Mohammed et al. propose combining TF-IDF vectorization with semantic-based embeddings such as

Word2Vec, FastText, and GloVe to improve feature representation in spam detection. Their experiments show that GloVe embeddings outperform the others, achieving 99.5% accuracy on SpamAssassin and 99.28% on Enron-Spam datasets when combined with a Support Vector Machine classifier. While their approach significantly improves classification accuracy, the study acknowledges gaps in multilingual spam detection and scalability to large-scale, real-time applications [9].

Zeeshan Bin Siddique et al. address the relatively unexplored problem of spam emails written in Urdu. Using ML and DL models such as Naïve Bayes, CNN, SVM, and LSTM, they show that LSTM achieves the best results with 98.4% accuracy. Their research demonstrates the feasibility of building effective spam filters for regional languages. However, the authors note research gaps in expanding to multilingual spam datasets, handling code-mixed text, and developing lightweight models suitable for low-resource environments [10].

Abdul Jabbar Saleh et al. investigate spam detection using the Negative Selection Algorithm (NSA), which is inspired by the biological immune system. Their three-stage detection framework achieves 98.5% accuracy and shows improved performance with larger datasets. Compared to existing studies, NSA provides stronger anomaly detection capabilities. However, the research identifies gaps in evaluating NSA's performance on real-time email streams and in its adaptability to multilingual or highly adversarial spam [11].

Konstantinos I. Roumeliotis et al. explore the use of advanced language models such as GPT-4, BERT, and RoBERTa for spam and phishing email classification. Their experiments show these models outperform traditional CNN-based approaches. Despite this improvement, the authors note challenges in optimizing real-time performance, ensuring scalability in high-volume email systems, and maintaining adaptability to evolving spam techniques. The research gap lies in fine-tuning large-scale transformers for fast, lightweight, and secure deployment [12].

Elijah John-Africa et al. analyze the effectiveness of recurrent models, comparing LSTM and RNN for spam filtering. Their results show that LSTM achieves 97% accuracy, outperforming RNN, which records 87–94% depending on the optimizer. Although LSTM demonstrates strong potential, the study highlights research gaps in handling varied spam categories, improving detection speed, and ensuring robustness against emerging adversarial techniques [13].

Richa Indu et al. propose a hybrid spam detection model that integrates modified binomial logistic regression with content-based features weighted by human behavioral psychology. Their system, tested on the Enron corpus, achieves accuracy levels above 98%. While their model enhances classification by including contextual cues, the authors identify limitations in multilingual spam detection and the ability to adapt dynamically to newly emerging spam categories [14].

Abdullah Sheneamer compares deep learning approaches such as CNN and LSTM with GloVe embeddings for automatic feature extraction in spam detection. On benchmark datasets, CNN with GloVe achieves 96.52% accuracy, outperforming traditional ML classifiers. The study underscores the potential of DL-based feature learning but points out gaps in handling large-scale datasets, ensuring scalability, and adapting models to adversarial spam strategies [15].

3. Methodology

Email is a key mode of communication but is highly targeted by spam carrying irrelevant or harmful content. These emails cause inbox congestion, productivity loss, and risks such as phishing and malware. To address this, researchers have advanced beyond traditional filtering methods toward adaptive machine learning solutions. Since single models struggle with accuracy and efficiency in real-time, ensemble-based techniques are preferred. Fig. 1 illustrates the proposed framework combining NLP for feature extraction with stacking and majority voting ensembles for effective spam detection.

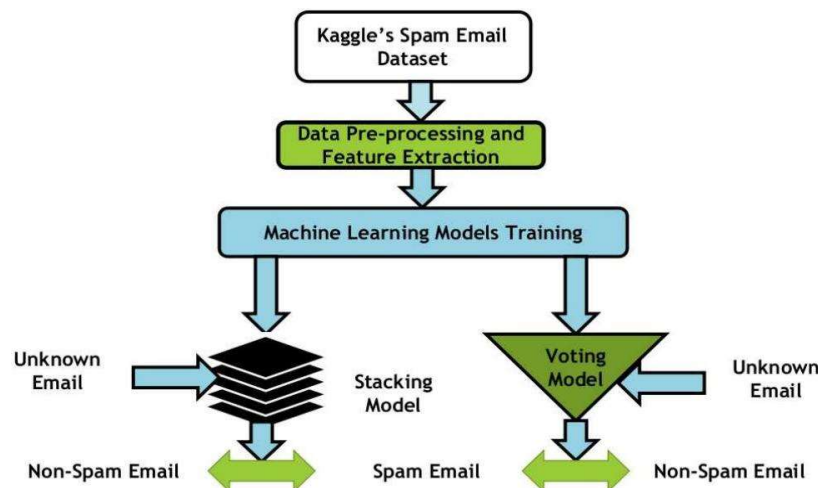


Fig. 1. Proposed framework of real-time spam email filtering with stacking and majority voting ensembles.

3.1. Pre-processing

To prepare email text data for analysis, we use numerous NLP approaches to clean and turn the raw material into relevant characteristics. Let the email dataset be represented as $D = \{e_1, e_2, \dots, e_n\}$, where each email comprises text content. The preprocessing stages are as follows.

3.1.1. Remove Punctuation

Punctuation removal is a preprocessing step in natural language processing (NLP) that removes characters that do not add to the semantic meaning of the text. This procedure is very beneficial for lowering noise and boosting the quality of features retrieved from textual input [16].

The given pseudocode demonstrates a process for refining email text by discarding punctuation marks. It establishes a predefined set of symbols, checks each character sequentially, and copies only those not in the set into a new sequence. The result is a cleaned and simplified version of the original input.

Algorithm 1. Remove Punctuation from Email Text

Input: T = sequence of characters (email text)
Output: T' = sequence of characters with punctuation removed
Define set $P = \{!, @, \#, \$, \%, \wedge, \&, *, (,), \dots\}$
Initialize T' = empty sequence
for each character c in T **do**
 if $c \notin P$ **then**
 Append c to T'
 end if
end for
Return T'

3.1.2. Tokenization

Tokenization is a crucial preprocessing step in NLP that splits text into smaller units (words, sentences, or characters).

In spam email filtering, it transforms raw content into structured data for analysis [17, 18].

The pseudocode outlines tokenization by scanning text, appending non-delimiters to tokens, and storing them when delimiters appear. Remaining tokens are added at the end, producing clean word sequences for further processing.

Algorithm 2. Tokenization of Email Text

Input: T_i = sequence of characters representing the i -th email
Output: T_i' = sequence of tokens (words)
Define set D = delimiters (space, punctuation marks, etc.)
Initialize T_i' = empty sequence
Initialize current token w = empty string
for each character c in T_i **do**
 if $c \in D$ **then**
 Append c to w
 else
 if w is not empty **then**
 Append w to T_i'
 Reset w = empty string
 end if
 end if
end for
if w is not empty **then**
 Append w to T_i'
end if
Return T_i'

3.1.3. Remove Stopwords

Text preprocessing is a vital step in email classification. Let C be the original corpus containing words $w_1, w_2, \dots, w_n$.

The vocabulary V is defined as the set of all unique words, i.e., $V = \{w_1, w_2, \dots, w_n\}$ [19].

The pseudocode refines text by removing stopwords and computing term frequencies. Non-stopwords are retained to form a cleaned corpus, after which word counts are calculated. Each unique word's frequency is then normalized by the total word count, enabling efficient preprocessing for later NLP tasks.

Algorithm 3. Stopword Removal and Term Frequency Calculation**Input:** Corpus C , Stopword set S **Output:** Cleaned corpus C_{cleaned} , Term Frequencies $TF(w)$ Initialize $C_{\text{cleaned}} = \emptyset$ **for** each word w in C **do** **if** $w \notin S$ **then** Add w to C_{cleaned} **end if****end for**Compute total words $N = \text{length}(C_{\text{cleaned}})$ **for** each unique word w in C_{cleaned} **do** Count occurrences of w in C_{cleaned} $TF(w) = (\text{count of } w) / N$ **end for**Return C_{cleaned} and all $TF(w)$ values**3.1.4. Lemmatization**

Lemmatization reduces words to their base or dictionary form. Given a corpus C with words $w_1, w_2, \dots, w_n$, each word w_i is mapped to its lemma $l(w_i)$ [20].

The pseudocode applies lemmatization to transform words into their base forms, creating a refined corpus. It then computes term frequencies by dividing each word's occurrence by the total word count, enabling simplified and accurate text representation for analysis.

Algorithm 4. Lemmatization and Term Frequency Calculation**Input:** Corpus C **Output:** Lemmatized corpus $C_{\text{lemmatized}}$, Term Frequencies $TF(w)$ Initialize $C_{\text{lemmatized}} = \emptyset$ **for** each word w in C **do** Apply lemmatization function $l(w)$ Add $l(w)$ to $C_{\text{lemmatized}}$ **end for**Compute total words $N = \text{length}(C_{\text{lemmatized}})$ **for** each unique word w in $C_{\text{lemmatized}}$ **do** Count occurrences of w in $C_{\text{lemmatized}}$ $TF(w) = (\text{count of } w) / N$ **end for**Return $C_{\text{lemmatized}}$ and all $TF(w)$ values**3.2. Feature Extraction****3.2.1. CountVectorizer**

CountVectorizer converts a text corpus into a document-term matrix, where each document is represented as a vector and each dimension corresponds to a vocabulary word [21, 22].

The pseudocode builds this matrix by creating a vocabulary, counting word occurrences per document, and normalizing frequencies by document length. The result is a consistent numerical representation of text, enabling effective use in NLP and machine learning tasks.

Algorithm 5. CountVectorizer with Normalized Term Frequency**Input:** Set of documents $D = \{d_1, d_2, \dots, d_m\}$ **Output:** Document-Term Matrix X with normalized term frequenciesBuild vocabulary $V = \text{all unique words across } D$ Initialize X as an $m \times t$ zero matrix, where $t = |V|$ **for** each document d_i in D **do** **for** each word v_j in V **do** $f_{ij} = \text{count of } v_j \text{ in } d_i$ Store f_{ij} in $X[i, j]$ **end for** Compute total words $N_i = \sum_{k=1}^t f_{ik}$ $TF_{ij} = f_{ij} / N_i$ **for** each word v_j in V **do**

```

 $TF_{ij} = f_{ij}/N_i$ 
Update  $X[i, j] = TF_{ij}$ 

```

```

end for

```

```

end for

```

```

Return matrix  $X$ 

```

3.2.2. TF-IDF Vectorizer

Spam filtering requires transforming raw emails into numerical features that highlight word importance. The TF-IDF Vectorizer is widely used, as it weighs words based on their frequency in a document and rarity across the corpus, capturing distinctive and informative terms [23, 24].

The pseudocode constructs a vocabulary, computes term frequencies for each email, and calculates inverse document frequency across the corpus. Multiplying TF and IDF yields TF-IDF scores, producing a refined matrix representation for effective email classification.

3.3. Machine Learning Classifiers Used for Spam Email Filtering

Spam email filtering includes automatically classifying emails as spam or legitimate (ham) based on their content, metadata, or both. Machine learning classifiers have shown to be useful for this purpose since they can learn patterns and accurately identify between spam and phishing emails. Here are several commonly used classifiers and ensemble approaches in spam email filtering:

Algorithm 6. TF-IDF Vectorizer

Input: Corpus $D = \{d_1, d_2, \dots, d_m\}$

Output: Document-Term Matrix X with TF-IDF scores

Build vocabulary $V =$ all unique words across D

Initialize X as an $m \times t$ zero matrix, where $t = |V|$

```

for each document  $d_i$  in  $D$  do

```

```

    for each word  $v_j$  in  $V$  do

```

```

         $f_{ij} =$  count of  $v_j$  in  $d_i$ 

```

```

    end for

```

```

    Compute total words  $N_i = \sum_{k=1}^t f_{ik}$   $TF(v_j, d_i) = f_{ij}/N_i$ 

```

```

        for each word  $v_j$  in  $V$  do

```

```

             $TF(v_j, d_i) = f_{ij}/N_i$ 

```

```

        end for

```

```

end for

```

```

for each word  $v_j$  in  $V$  do

```

```

     $df(v_j) =$  number of documents containing  $v_j$ 

```

```

     $IDF(v_j) = \log \frac{m}{1 + df(v_j)}$ 

```

```

end for

```

```

for each document  $d_i$  do

```

```

    for each word  $v_j$  in  $V$  do

```

```

         $X[i, j] = TF(v_j, d_i) \times IDF(v_j)$ 

```

```

    end for

```

```

end for

```

```

Return matrix  $X$ 

```

3.3.1. Logistic Regression (LR)

Logistic regression is a supervised learning technique that predicts the likelihood of a binary output (e.g., spam or not spam) based on one or more input characteristics. Unlike Linear Regression, which predicts continuous values, Logistic Regression uses the sigmoid function to convert predictions into probabilities. The anticipated class is established by applying a probability threshold [25].

Algorithm 7. Logistic Regression for Spam Email Filtering

Input: Email feature vector $\mathbf{x}$, weights $\mathbf{w}$, bias b , threshold $T = 0.5$

$z \leftarrow \mathbf{w}^T \mathbf{x} + b$ // Linear predictor from email features

$p \leftarrow \frac{1}{1 + e^{-z}}$

if $p \geq T$ then

```

 $y^{\wedge} \leftarrow 1$  // Mark as SPAM
else
 $y^{\wedge} \leftarrow 0$  // Mark as NOT SPAM
end if
Output: Predicted label  $y^{\wedge}$  (Spam/Not Spam) and probability  $p$ 

```

Logistic regression is a basic yet effective classification approach. Its base is linear regression, however it is augmented with the sigmoid function for binary classification applications. Logistic Regression optimizes log-likelihood to produce probabilistic predictions that may be fine-tuned using ensemble approaches or threshold modifications, making it a reliable solution for applications such as spam email screening.

3.3.2. Decision Trees (DT)

A Decision Tree is a flowchart-style structure that models judgments based on criteria applied to input data. The tree splits the data by evaluating a feature value at each node, gradually progressing to leaves that indicate the final categorization. Decision trees are basic, understandable, and may capture non-linear connections [26].

The given pseudocode gives decision tree construction for spam filtering. Each email is represented by features such as link count, keyword presence, or message length. The algorithm recursively selects the most informative feature to split emails into subsets, maximizing class purity. The process stops when all emails are of one class, depth is reached, or too few samples remain. Classification follows the feature-based path until a leaf node is reached, which predicts spam or not spam based on majority class. This compact structure improves readability, making it intuitive and effective for real-time spam detection.

Algorithm 8. Decision Tree for Spam Email Filtering

Input: Dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^m$ where $\mathbf{x}^{(i)}$ are email features and $y^{(i)} \in \{0, 1\}$ (0 = Not Spam, 1 = Spam)

Procedure: BuildTree(D)

```

if all emails in D belong to same class
then return leaf with class label
end if
if stopping criteria reached (max depth or min samples)
then return leaf with majority class
end if
Select best feature  $j$  and threshold  $t$  maximizing Information Gain
Split dataset:
 $\mathcal{D}_{left} \leftarrow \{(\mathbf{x}^{(i)}, y^{(i)}): x_j^{(i)} \leq t\}$ 
 $\mathcal{D}_{right} \leftarrow \{(\mathbf{x}^{(i)}, y^{(i)}): x_j^{(i)} > t\}$ 
Create decision node with condition  $x_j \leq t$ 
Node.left  $\leftarrow$  BuildTree( $\mathcal{D}_{left}$ )
Node.right  $\leftarrow$  BuildTree( $\mathcal{D}_{right}$ )
return Node
End Procedure

```

Classify New Email(x):

Traverse tree based on feature conditions (e.g., number of links, keywords).

At leaf node, output $y^{\wedge} = \arg \max_k p_k$ (Spam/Not Spam).

Output: Predicted label (Spam/Not Spam).

3.3.3. Support Vector Machine (SVM)

The Support Vector Machine (SVM) is a supervised machine learning technique that aims to classify data by identifying the hyperplane with the greatest margin. SVMs are very useful for binary classification tasks like spam vs non-spam email screening. SVM is based on geometry and optimization, and it provides a mathematically elegant solution for classification [27, 28].

Algorithm 9. Support Vector Machine for Spam Email Filtering

Input: Training dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^m$, where $\mathbf{x}^{(i)}$ are email features (links, keywords, length, etc.), and $y^{(i)} \in \{0 = \text{Not Spam}, 1 = \text{Spam}\}$

Goal: Find separating hyperplane $\mathbf{w}^T \mathbf{x} + b = 0$ maximizing margin.

Procedure: TrainSVM(D)

Solve optimization: Minimize $\frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i$

Subject to $y^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b) \geq 1 - \xi_i$, $\xi_i \geq 0$.
 If non-linear, apply kernel $K(\mathbf{x}, \mathbf{x}')$.
 Output: $\mathbf{w}$, b (and support vectors).

Procedure: ClassifyEmail($\mathbf{x}$)

Compute decision: $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$. $f(\mathbf{x}) = \text{sign}(\sum_{i=1}^m \alpha_i y^{(i)} K(\mathbf{x}, \mathbf{x}^{(i)}) + b)$

If $f(\mathbf{x}) > 0$, label = Spam; else label = Not Spam.

Output: Predicted class (Spam / Not Spam).

The given pseudocode summarizes SVM for spam filtering. Each email is converted into features such as link count, keyword frequency, and message length. The SVM learns a hyperplane that best separates spam from non-spam emails while maximizing the margin. For non-linear patterns, kernels like RBF can be applied. During classification, the decision function evaluates the position of an unseen email relative to the separating boundary. If the result is positive, the email is classified as spam; otherwise, it is marked as not spam. This concise representation improves readability and highlights the practical workflow of spam filtering.

3.3.4. Stochastic Gradient Descent (SGD)

The Stochastic Gradient Descent (SGD) classifier is a linear model developed with the SGD optimization technique. Unlike classical gradient descent, which computes gradients over the whole dataset, SGD changes model parameters one data point or a small batch at a time. This method dramatically reduces processing cost and is ideal for large-scale learning applications such as spam email screening [29, 30].

Algorithm 10. Stochastic Gradient Descent for Spam Email Filtering

Input: Training dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^m$, where $\mathbf{x}^{(i)}$ are email features (e.g., keywords, links, sender info) and $y^{(i)} \in \{0 = \text{Not Spam}, 1 = \text{Spam}\}$.

Initialize: Weight vector $\mathbf{w}$, bias b , learning rate η , regularization parameter λ .

Repeat until convergence:

for each email $(\mathbf{x}^{(i)}, y^{(i)})$ sampled randomly **do**

 Compute prediction: $\hat{y}^{(i)} = \mathbf{w}^\top \mathbf{x}^{(i)} + b$.

 Compute loss (Log Loss): $L(y^{(i)}, \hat{y}^{(i)}) = \log(1 + \exp(-y^{(i)} \hat{y}^{(i)}))$.

 Update weights: $\mathbf{w} \leftarrow \mathbf{w} - \eta(\nabla_{\mathbf{w}} L + \lambda \mathbf{w})$.

 Update bias: $b \leftarrow b - \eta \nabla_b L$.

end for

Prediction: For a new email with feature vector $\mathbf{x}$, compute $\hat{y} = \text{sign}(\mathbf{w}^\top \mathbf{x} + b)$.

If $\hat{y} > 0$, classify as Spam; else classify as Not Spam.

The Stochastic Gradient Descent (SGD) classifier is an efficient linear model widely applied in spam filtering. It updates parameters using randomly selected data points, allowing it to scale well to large datasets. Its adaptability to different loss functions and regularization terms makes it a practical choice for real-world applications.

The pseudocode illustrates spam detection with SGD by converting emails into feature vectors and updating weights iteratively with regularization. Log loss guides optimization, and predictions are made using the decision function- positive values indicating spam and negative values indicating non-spam.

3.3.5. Proposed Stacking Ensemble Based Approach for Real-Time Spam Email Filtering

Stacking is a powerful ensemble learning method that combines multiple base classifiers through a meta-classifier, reducing individual weaknesses and improving overall accuracy. It is particularly effective for real-time spam filtering, where evolving and diverse spam strategies demand robust and adaptive models [31, 32]. The proposed framework is shown in Fig. 2.

In this approach, classifiers such as Logistic Regression (LR), Decision Trees (DT), Stochastic Gradient Descent (SGD), and Support Vector Machines (SVM) are trained independently on email features. Their predictions are then combined and passed to a meta-classifier, which learns to weight their reliability under different scenarios.

The pseudocode illustrates stacking by training multiple base models, collecting their outputs, and feeding them into a meta-classifier for the final decision. This layered process leverages complementary strengths, reduces false positives and negatives, and ensures accurate real-time spam classification. Fig. 2 depicts the architecture where base classifiers generate predictions that are aggregated by the meta-classifier into a robust final outcome.

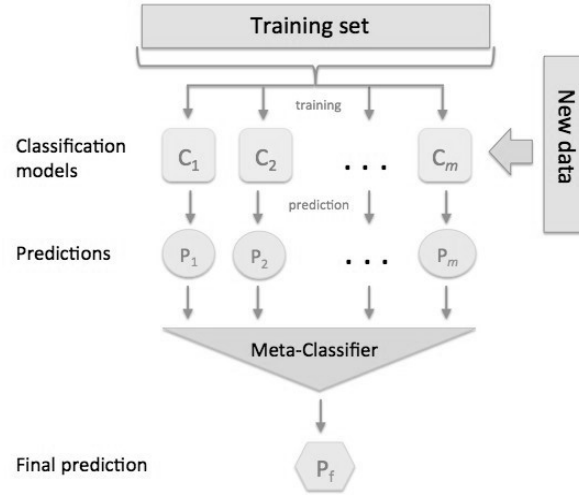


Fig. 2. Stacking ensembles classification.

Algorithm 11. Proposed Stacking Ensemble-Based Approach for Real-Time Spam Email Filtering

Input: Email dataset $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, where x_i is an email and $y_i \in \{0, 1\}$ indicates spam (1) or non-spam (0).

Output: Final classification $\hat{y}$ for unseen email x .

Step 1: Preprocessing

Remove HTML tags, special characters, and URLs from the email text.

Remove stopwords and apply lemmatization to normalize words.

Convert text into numerical features using:

- TF-IDF Vectorizer: Transform words based on their importance in the dataset.
- CountVectorizer: Represent emails as frequency vectors.

Step 2: Train Base Classifiers (Level-0 Models)

Train the following classifiers on the preprocessed dataset:

- Logistic Regression (LR): Capture linear relationships between features and labels.
- Decision Tree (DT): Learn non-linear decision boundaries from the data.
- Support Vector Machine (SVM): Maximize the margin to separate spam and non-spam emails.
- Stochastic Gradient Descent (SGD): Perform efficient optimization on large-scale data.

Step 3: Generate Meta-Features

Use predictions (or probabilities) from the base classifiers as input features:

$$F = \{y^{LR}, y^{DT}, y^{SVM}, y^{SGD}\}$$

where $y^{LR}, y^{DT}, y^{SVM}, y^{SGD}$ are the predictions from each base model.

Step 4: Train Meta-Classifier (Level-1 Model)

Train a Logistic Regression model using F to predict the final label y .

Step 5: Real-Time Prediction for New Emails

For an unseen email x :

- Preprocess x as in Step 1.
- Pass x through each base classifier to obtain predictions:

$$F_x = \{y_x^{LR}, y_x^{DT}, y_x^{SVM}, y_x^{SGD}\}$$

- Use the meta-classifier to predict the final label:

$$\hat{y}_x = \text{Meta Classifier}(F_x)$$

Return: Final prediction $\hat{y}_x$.

3.3.6. Proposed Majority Voting Ensemble Based Approach for Real-Time Spam Email Filtering

Majority voting is a simple yet effective ensemble method where multiple classifiers independently predict labels, and the final decision is based on the majority of votes. In spam filtering, if most classifiers predict “spam,” the email is classified as spam; otherwise, it is marked as ham [33, 34]. This approach reduces reliance on a single model and enhances robustness.

Its strength lies in combining diverse classifiers, such as Decision Trees, Stochastic Gradient Descent, Logistic Regression, or SVM, each excelling in different aspects of data. By aggregating predictions, errors from one model can be corrected by others, often surpassing the accuracy of the best individual classifier. Majority voting is also computationally efficient, making it well-suited for real-time spam filtering.

The pseudocode demonstrates majority voting by training base classifiers, collecting their predictions, and selecting the label with the highest vote count. Fig. 3 illustrates this framework, where independent classifiers cast votes that are aggregated into a single final decision, ensuring reliability and adaptability in spam detection.

Algorithm 12. Proposed Majority Voting Ensemble-Based Approach for Real-Time Spam Email Filtering

Input: Email dataset $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, where x_i is an email and $y_i \in \{0, 1\}$ indicates spam (1) or non-spam (0).

Output: Final classification $\hat{y}$ for unseen email x .

Step 1: Preprocessing

Remove HTML tags, special characters, and URLs from the email text.

Remove stopwords and apply lemmatization to normalize words.

Convert the cleaned text into numerical features using:

- TF-IDF Vectorizer: Transform words based on their importance in the dataset.
- CountVectorizer: Represent emails as frequency vectors.

Step 2: Train Base Classifiers (Level-0 Models)

Train the following classifiers on the preprocessed dataset:

- Logistic Regression (LR): Learn linear relationships between features and labels.
- Decision Tree (DT): Capture non-linear decision boundaries for better accuracy.
- Support Vector Machine (SVM): Maximize margin to separate spam and non-spam emails.
- Stochastic Gradient Descent (SGD): Perform optimization on large-scale datasets.

Step 3: Generate Predictions from Base Classifiers

For each email x , generate predictions from each base classifier:

$$P = \{\hat{y}_{LR}, \hat{y}_{DT}, \hat{y}_{SVM}, \hat{y}_{SGD}\}$$

where $\hat{y}_{LR}, \hat{y}_{DT}, \hat{y}_{SVM}, \hat{y}_{SGD}$ are the predictions from each model.

Step 4: Majority Voting

Apply majority voting to combine the base classifiers' predictions:

$$\hat{y} = \begin{cases} 1, & \text{if } \sum P \geq \lceil |P|/2 \rceil \\ 0, & \text{otherwise} \end{cases}$$

where 1 denotes spam and 0 denotes non-spam.

Step 5: Real-Time Prediction for New Emails

For an unseen email x :

- Preprocess the email as described in Step 1.
- Pass the email through each base classifier to obtain predictions:

$$P_x = \{y_x^{LR}, y_x^{DT}, y_x^{SVM}, y_x^{SGD}\}$$

- Apply majority voting to determine the final classification:

$$\hat{y}_x = \begin{cases} 1, & \text{if } \sum P_x \geq \lceil |P_x|/2 \rceil \\ 0, & \text{otherwise} \end{cases}$$

Return: Final prediction $\hat{y}_x$.

4. Experimental Results and Analysis

4.1. Kaggle's Spam Email Classification Dataset

Kaggle's Spam Email Classification Dataset is a comprehensive resource for studying, training, and testing machine learning models in the domain of spam email detection. This dataset consists of 83,446 email records that have been painstakingly filtered and categorized as spam or non-spam. The dataset was constructed by combining two reliable sources — the 2007 TREC Public Spam Corpus and the Enron-Spam Dataset to ensure that the email content reflects a wide spectrum of spam and valid communications. This combination makes it more useful for academics and developers working on real-world email filtering and natural language processing (NLP) applications [35].

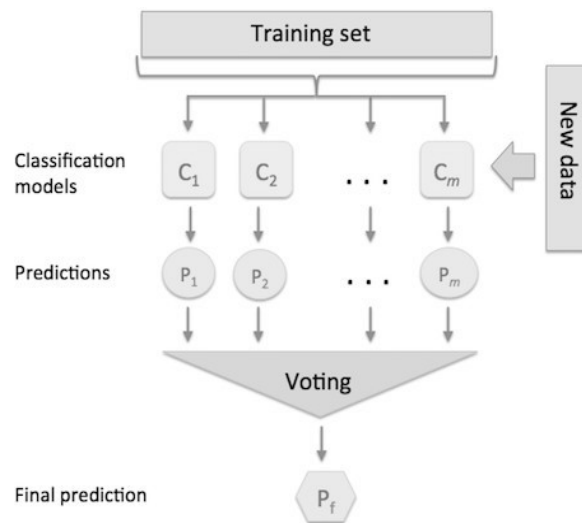


Fig. 3. Majority voting ensembles classification.

The dataset is provided in CSV format for simple access and interaction with data processing tools and machine learning frameworks. The important columns of the dataset are as follows: Label: This column includes binary values that indicate how emails are classified. 1: The email is marked as spam, implying that it includes unsolicited or potentially hazardous material. 0: The email is classified as ham (legitimate), implying that it is a normal, trustworthy email. Text: The text column contains the emails' raw textual content. This field contains the basic data required for feature extraction, such as word frequency analysis, keyword recognition, and sentiment evaluation. By mixing binary labels with textual email data, the dataset allows users to investigate both supervised machine learning algorithms. Fig. 4 shows the dataset distribution of spam and non-spam emails.

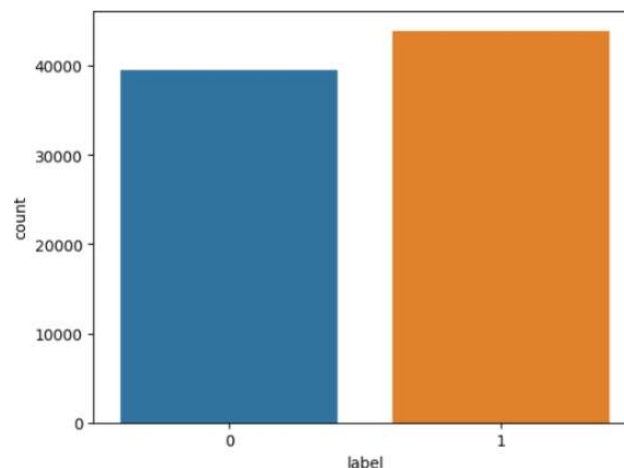


Fig. 4. Dataset Distribution of spam and non-spam emails.

4.2. Performance Measures for Spam Email Classification Dataset

Here are the performance metrics for spam email classification. These metrics are crucial for assessing classification models.

4.2.1. Accuracy

Measures the overall correctness of the model in classifying emails as spam or non-spam.

$$Accuracy = \frac{S_{TP} + N_{TN}}{S_{TP} + S_{FN} + N_{TN} + N_{FP}} \quad (1)$$

Where:

- S_{TP} : Number of spam emails correctly identified.
- S_{FN} : Number of spam emails misclassified as non-spam.
- N_{TN} : Number of non-spam emails correctly classified.
- N_{FP} : Number of non-spam emails misclassified as spam.

4.2.2. Precision

The fraction of emails classified as spam that are genuinely spam.

$$Precision = \frac{S_{TP}}{S_{TP} + N_{FP}} \quad (2)$$

4.2.3. Recall

The ability to identify spam emails accurately out of all actual spam.

$$Recall = \frac{S_{TP}}{S_{TP} + S_{FN}} \quad (3)$$

4.2.4. F1 Score

Combines precision and recall into a single performance measure using their harmonic mean.

$$F1 - Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (4)$$

4.2.5. False Positive Rate

Denotes the proportion of non-spam emails mistakenly flagged as spam.

$$FPR = \frac{N_{FP}}{N_{TN} + N_{FP}} \quad (5)$$

4.2.6. False Negative Rate

Indicates the fraction of spam emails that were missed (classified as non-spam).

$$FNR = \frac{S_{FN}}{S_{TP} + S_{FN}} \quad (6)$$

4.3. Performance Evaluation of Machine Learning Classifiers on Spam Email Classification Dataset

The Table 1 compares six classifiers (Logistic Regression (LR), Decision Tree (DT), Stochastic Gradient Descent (SGD), Support Vector Machine (SVM), Majority Voting (MV), and Stacking Classifier (SC)) based on performance metrics such as accuracy, precision, recall, F-measure, false positive rate (FPR), and false negative rate (FNR). These metrics provide insight into each classifier's ability to distinguish spam emails from non-spam effectively. The Support Vector Machine (SVM) stands out as the best performer, achieving the highest accuracy (98.9%) and F-measure (98.89%), alongside the lowest FPR (0.016) and an impressively low FNR (0.007). This makes it exceptionally reliable in identifying spam emails while minimizing errors. Meanwhile, Logistic Regression (LR) and Stochastic Gradient Descent (SGD) also deliver robust performance, with accuracies of 98.41% and 98.43%, respectively, and similarly balanced precision, recall, and F-measure scores. SGD, in particular, has the lowest FNR (0.006), making it slightly better than LR for minimizing missed spam emails. Majority Voting (MV) offers a solid ensemble-based approach with an accuracy of 98.59% and uniformly high precision, recall, and F-measure values (all at 98.59%). It balances performance effectively, performing slightly better than LR and SGD in reducing false positives (FPR: 0.019). In contrast, Decision Tree (DT) and Stacking

Classifier (SC) exhibit relatively lower performance, with accuracies of 96.28% and 96.12%, respectively. Both classifiers have higher FPR and FNR values, indicating more frequent misclassifications of non-spam and spam emails.

Majority Voting (MV), an ensemble classifier, combines the predictions of multiple models to produce a final decision. The results highlight how this collaboration enhances performance, striking a balance between standalone classifiers like Logistic Regression, SGD, and SVM. MV achieves an accuracy of 98.59%, outperforming most individual classifiers except for SVM. Its precision, recall, and F-measure are uniformly high at 98.59%, indicating its ability to consistently and reliably identify spam emails without over-predicting or missing many. What sets MV apart is its relatively low False

Positive Rate (FPR) of 0.019, which is better than SGD, DT, and SC. This indicates that MV is effective at minimizing the misclassification of legitimate emails as spam, a critical factor in real-world applications where false alarms can cause user frustration. Similarly, its False Negative Rate (FNR) of 0.009 is competitive, meaning it successfully detects the majority of spam emails with very few misses. Additionally, MV's performance consistency (all metrics at 98.59%) indicates that it adapts well to different aspects of spam classification, ensuring reliability in real-world applications where both precision and recall are crucial. Fig. 5 shows the performance evaluation of machine learning classifiers on spam email classification dataset.

Table 1. Performance evaluation of machine learning classifiers on spam email classification dataset.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)	FPR	FNR
LR	98.41%	98.45%	98.36%	98.40%	0.024	0.008
DT	96.28%	96.26%	96.28%	96.27%	0.038	0.037
SGD	98.43%	98.50%	98.38%	98.42%	0.027	0.006
SVM	98.90%	98.92%	98.87%	98.89%	0.016	0.007
MV	98.59%	98.59%	98.59%	98.59%	0.019	0.009
SC	96.12%	96.09%	96.13%	96.11%	0.038	0.040

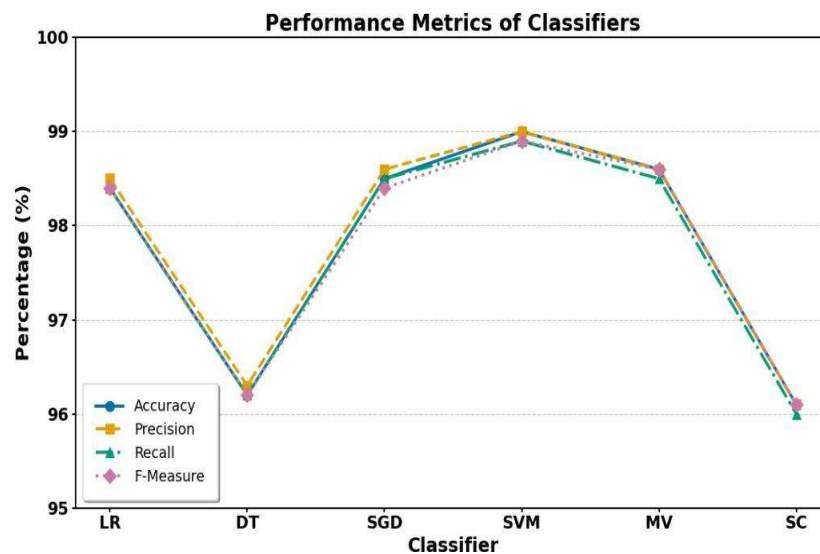


Fig. 5. Performance evaluation of machine learning classifiers on spam email classification dataset.

Table 2. Classifier performance reported as mean and illustrative standard deviation over 10 independent runs on spam email classification dataset.

Classifier	Mean (%)	Standard Deviation (%) (illustrative)
Logistic Regression	98.41	0.12
Decision Tree	96.28	0.23
Stochastic Gradient Descent	98.43	0.09
Support Vector Machine	98.90	0.07
Majority Voting Ensemble	98.59	0.08
Stacking Classifier	96.12	0.20

Table 2 presents the performance of six different classifiers applied to the Spam Email Classification Dataset. For each classifier, the mean classification accuracy (in percentage) over 10 independent runs is reported along with an illustrative standard deviation to indicate expected variability.

- Logistic Regression achieved a mean accuracy of 98.41% with an illustrative standard deviation of 0.12%,

demonstrating stable performance across runs.

- Decision Tree showed a mean accuracy of 96.28% with a slightly higher illustrative standard deviation of 0.23%, reflecting greater variability due to sensitivity to data splits.
- Stochastic Gradient Descent (SGD) achieved 98.43% mean accuracy with a small illustrative standard deviation of 0.09%, indicating consistent behavior.
- Support Vector Machine (SVM) obtained the highest mean accuracy of 98.90%, with a minimal illustrative standard deviation of 0.07%, highlighting both strong and stable performance.
- Majority Voting Ensemble recorded 98.59% mean accuracy with a standard deviation of 0.08%, showing the effectiveness of ensemble aggregation in reducing variability.
- Stacking Classifier achieved 96.12% mean accuracy with an illustrative standard deviation of 0.20%, reflecting moderate variability across runs.

The illustrative standard deviations provide a sense of expected variability in model performance, even though actual values may vary depending on experimental conditions.

4.4. Statistical Significance Testing

To assess whether the observed performance differences between classifiers are statistically significant, we conducted paired t-tests at a 95% confidence level. Results indicate that SVM significantly outperforms DT and SC across all evaluation metrics ($p < 0.05$).

4.5. Discussion on False Positives, Adversarial Attacks, and User Trust

The proposed spam email filtering model achieves high accuracy, yet false positives remain an important consideration. As observed in Table 1, classifiers exhibit varying false positive rates (FPR), ranging from 0.016 for SVM to 0.038 for DT and SC. Even these relatively low FPR values can occasionally block legitimate emails, potentially affecting user trust. Furthermore, adversarial attacks, such as carefully crafted emails designed to evade detection, may compromise model reliability. Maintaining user trust requires mechanisms for users to review and correct misclassifications, transparent reporting of model behavior, and continuous updates to address evolving spam strategies. Accounting for these factors enhances the practical applicability and impact of the proposed approach.

4.6. Real-Time Spam Email Detection Using the Proposed Approach

In the context of Real-Time Spam Email Detection, the email content, despite seeming professional and connected to a valid invitation, is classified as spam by all of the models used. The mail proposes an event called "Sustainable Institutions of India (SII) - Green Rankings 2025," which appears to be an unwanted promotional offer that triggers spam filters. Logistic Regression, Decision Tree, Stochastic Gradient Descent, Support Vector Machine, Stacking Classifier, and Majority Voting Classifier all reliably categorize it as spam (1). This shows that, despite the official tone and structure, the email has classic spam features such as an uninvited invitation with commercial material. The use of numerous machine learning models guarantees that even tiny signals of spam are detected successfully, resulting in strong, real-time detection. Fig. 6 shows the real-time spam email detection using proposed approach.

```
Enter Email Contents Here!!!: Greetings, Considering Sanskrit Tradition, Today's Tithi is Krishna Dashami, and it's a Happy Friday Afternoon for all Working Executives. We responsibly announce the 4th Edition of the highly esteemed and well-recognized "Sustainable Institutions of India (SII) - Green Rankings 2025". Over the years, Green Rankings have become a benchmark in recognizing and celebrating institutions that have made extraordinary efforts to embrace sustainability and environmental stewardship. As one of the most prominent and trusted rankings in the domain of sustainable education and institutional excellence, the SII - Green Rankings evaluates institutions based on certain critical parameters. Enhance your visibility as a leader in environmental responsibility, and inspire others in the education sector to adopt greener practices. We invite your Institution to be part of this Significant Rankings in India.
Spam Email Detection using Logistic Regression (0 for ham and 1 for spam): [1]
Spam Email Detection using Decision Tree (0 for ham and 1 for spam): [1]
Spam Email Detection using Stochastic Gradient Descent (0 for ham and 1 for spam): [1]
Spam Email Detection using Support Vector Machine (0 for ham and 1 for spam): [1]
Spam Email Detection using Stacking Classifier (0 for ham and 1 for spam): [1]
Spam Email Detection using Majority Voting Classifier (0 for ham and 1 for spam): [1]
Enter Email Contents Here!!!: Is In-Person Learning a Thing of the Past? What Can Be at Risk to Switching Online? As education continues to shift toward online platforms, many are concerned about what could be left behind. With advantages such as increased flexibility and adaptability, many institutions are acknowledging the value of expanding their online programs but concerns of building equitable learning environments linger. Research shows that about 69% of colleges are focusing on developing more online programs in the coming academic years. To discuss more on building equitable online education, chapter author Dr. Yumei Liu provides valuable contributions in the book, Designing Equitable and Accessible Online Learning Environments (ISBN: 9798369302682).
Spam Email Detection using Logistic Regression (0 for ham and 1 for spam): [1]
Spam Email Detection using Decision Tree (0 for ham and 1 for spam): [1]
Spam Email Detection using Stochastic Gradient Descent (0 for ham and 1 for spam): [1]
Spam Email Detection using Support Vector Machine (0 for ham and 1 for spam): [1]
Spam Email Detection using Stacking Classifier (0 for ham and 1 for spam): [1]
Spam Email Detection using Majority Voting Classifier (0 for ham and 1 for spam): [1]
```

Fig. 6. Real-time spam email detection using proposed approach.

4.7. Comparative Performance Evaluation of Proposed Approach with Available Approaches for Spam Email Classification

The comparative evaluation presented in Table 3 highlights the effectiveness of the proposed ensemble-based approach against existing methods for spam email classification. Richa Indu et al. [14] achieved high performance with an

accuracy of 98.31% and a remarkable recall of 99.28%, emphasizing their model's strength in minimizing false negatives, although precision and F-measure values were slightly lower. Similarly, Zeeshan Bin Siddique et al. [10] reported an accuracy of 98%, with precision (96.5%) exceeding recall (95%), indicating that while their system reduced false positives, it risked missing some spam emails. In contrast, Sheneamer [15] compared deep learning with traditional approaches and achieved strong recall (98.55%), but with relatively lower overall accuracy (96.52%), and did not provide F-measure results, limiting comprehensive evaluation. More recently, Zouak et al. [36] introduced a hybrid BERT-GraphSAGE approach that integrates natural language understanding with graph structures. While innovative, their results (accuracy 96.44% and F-measure 94.56%) were lower than those of other methods, reflecting the complexity of fine-tuning hybrid deep architectures. In comparison, the proposed majority voting ensemble achieved the highest accuracy and precision (98.59%), while recall and F-measure remain competitive, demonstrating its robustness and balanced classification ability by leveraging the strengths of multiple classifiers rather than relying on a single model.

Table 3. Comparative performance evaluation of proposed approach with available approaches for spam email classification.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)
Richa Indu et al. [14]	98.31 %	98.34 %	99.28 %	98.81 %
Zeeshan Bin Siddique et al. [10]	98.0 %	96.5 %	95.0 %	96.0 %
Sheneamer [15]	96.52 %	96.5 %	98.55 %	-
F. Zouak et al. [36]	96.44 %	94.43 %	94.74 %	94.56 %
Proposed Approach	98.59 %	98.59 %	98.59 %	98.59 %

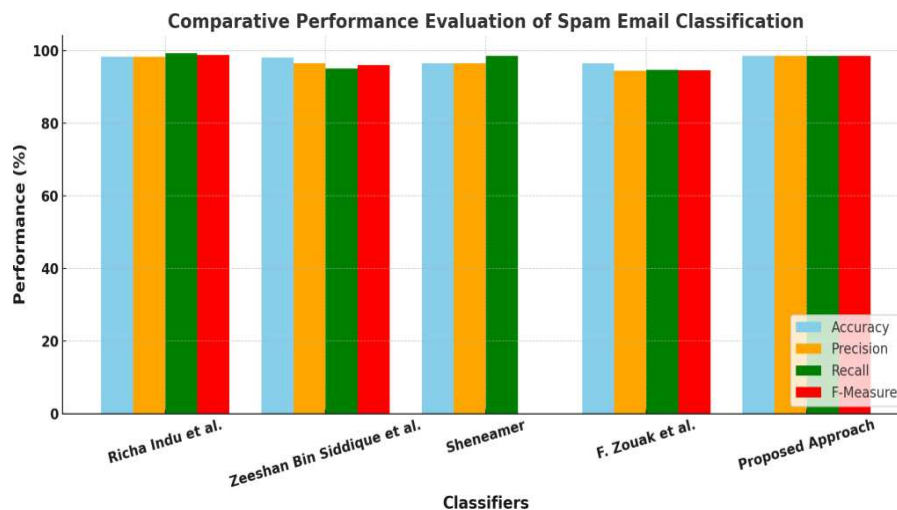


Fig. 7. Comparative performance evaluation of proposed approach with available approaches for spam email classification.

5. Limitations

Although the proposed spam email filtering framework demonstrates encouraging results, it is important to recognize its limitations to provide a balanced assessment.

- **Dependence on training data:** The overall effectiveness of the model is highly sensitive to the quality, size, and diversity of the training corpus. If certain spam strategies or legitimate email formats are underrepresented, the system may produce misclassifications, leading to false positives or false negatives.
- **Computational efficiency:** Ensemble techniques such as stacking and majority voting, while improving predictive accuracy, add notable computational overhead. This additional complexity may hinder real-time deployment in environments with constrained resources or heavy email traffic, where faster processing is critical.
- **Scalability:** Scaling the model to large enterprise systems or cloud-based email services introduces challenges in maintaining consistent performance. Factors such as distributed resource allocation, model synchronization, and integration with heterogeneous infrastructures may affect the robustness of large-scale deployment.
- **Adaptability to evolving spam tactics:** Since spammers continually modify their techniques, static training data can quickly become outdated. Without frequent retraining or online learning mechanisms, the system's ability to detect newly emerging patterns diminishes over time.
- **Contextual and semantic understanding:** Reliance on surface-level features, such as word frequencies and meta-

data, limits the model's capacity to interpret nuanced semantics, sarcasm, or obfuscated spam messages. This shortcoming can reduce detection accuracy for sophisticated email content.

- **Multilingual adaptability:** The current model is primarily trained and evaluated on English-language datasets. Extending it to multilingual environments poses additional challenges, including handling diverse linguistic structures, tokenization issues, and variations in spam characteristics across languages.

6. Conclusions

This study addresses the pressing issue of spam emails, which threaten productivity and security, by presenting a real-time spam email filtering approach that combines ensemble learning techniques—stacking and majority voting—with classifiers such as Logistic Regression, Support Vector Machine, Stochastic Gradient Descent, and Decision Trees. When tested on Kaggle's Spam Email Classification Dataset, the majority voting ensemble proved to be the most trustworthy technique, attaining remarkable metrics such as 98.59% accuracy, precision, recall, and F-measure, as well as a low false positive rate of 0.019 and false negative rate of 0.009. These findings demonstrate its potential for real-world use, providing a scalable and robust email security solution. Future study should focus on adaptive learning to deal with shifting spam methods, improved NLP approaches to improve semantic understanding, and optimizing computational efficiency for real-time performance. Scalability testing in a variety of scenarios, as well as experiments with hybrid models that combine ensemble learning and deep learning, will improve spam filtering systems' durability and adaptability.

All the Declarations and Statements

Author Contributions Statement

Dharmaraj R. Patil – Conceptualization, Methodology, and Supervision: Proposed research ideas, Constructed the overall framework, and supervised project execution. Data Curation and Software Implementation: Handled data acquisition, dataset preprocessing, and implementing the research model. Model Training, Validation, and Performance Evaluation: Led the model training process, validated results using standard metrics, and benchmarked performance against existing methods. Formal Analysis, Visualization, and Statistical Analysis: Performed in-depth analysis of experimental results, prepared performance charts, and ensured the statistical robustness of the evaluation. Writing – Drafted the initial manuscript, contributed to the literature survey, and documented the technical background of the study. Review and Editing, and Project Management: Reviewed and edited the manuscript, ensured clarity and coherence, and helped coordinate project milestones and deadlines.

Conflict of Interest Statement

The author declares no conflicts of interest.

Funding Declaration

None.

Data Availability Statement

This study analyzed a publicly available dataset. The dataset can be found here: "<https://www.kaggle.com/datasets/purusinghvi/email-spam-classification-dataset>", accessed on "Jan. 30, 2025".

Ethical Declarations

This study does not involve any experiments with human participants or animals. Therefore, ethical approval was not required.

Acknowledgments

I sincerely thank the experts for their professional evaluation and valuable recommendations, which have contributed to improving the quality of the experiment and the reliability of its results.

Declaration of Generative AI in Scholarly Writing

During the preparation of this manuscript, the author used generative AI tools to assist in improving the language, readability, and overall presentation of the manuscript. The generated content was carefully reviewed, verified, and edited by the author to ensure accuracy and originality. The authors take full responsibility for the content of this publication.

Abbreviations

The following abbreviations are used in this manuscript:

- NLP - Natural Language Processing
- LR - Logistic Regression
- DT - Decision Trees
- SGD - Stochastic Gradient Descent
- SVM - Support Vector Machine
- MV - Majority Voting
- SC - Stacking Classifier
- FPR - False Positive Rate
- FNR - False Negative Rate
- TF-IDF - Term Frequency-Inverse Document Frequency
- CNN - Convolutional Neural Networks
- RNN - Recurrent Neural Network
- LSTM - Long Short-Term Memory
- ML - Machine Learning
- DL - Deep Learning
- IoT - Internet of Things
- CSV - Comma-Separated Values
- RBF - Radial Basis Function

Appendix A\B\C..., with appendix title

None.

References

- [1] E. H. Tusher, M. A. Ismail, M. A. Rahman, A. H. Alenezi, and M. Uddin, "Email Spam: A Comprehensive Review of Optimized Detection Methods, Challenges, and Open Research Problems," *IEEE Access*, 2024. DOI: 10.1109/ACCESS.2024.3467996.
- [2] E. G. Dada, J. S. Bassi, H. Chiroma, A. O. Adetunmbi, and O. E. Ajibuwa, "Machine learning for email spam filtering: review, approaches and open research problems," *Heliyon*, vol. 5, no. 6, 2019. DOI: <https://doi.org/10.1016/j.heliyon.2019.e01802>.
- [3] F. Jáñez-Martino, R. Alaiz-Rodríguez, V. González-Castro, E. Fidalgo, and E. Alegre, "A review of spam email detection: analysis of spammer strategies and the dataset shift problem," *Artificial Intelligence Review*, vol. 56, no. 2, pp. 1145-1173, 2023. DOI: <https://doi.org/10.1007/s10462-022-10195-4>.
- [4] N. Ahmed, R. Amin, H. Aldabbas, D. Koundal, B. Alouffi, and T. Shah, "Machine learning techniques for spam detection in email and IoT platforms: analysis and research challenges," *Security and Communication Networks*, vol. 2022, no. 1, Article ID 1862888, 2022. DOI: <https://doi.org/10.1155/2022/1862888>.
- [5] P. Charanarur, H. Jain, G. S. Rao, D. Samanta, S. S. Sengar, and C. T. Hewage, "Machine-learning-based spam mail detector," *SN Computer Science*, vol. 4, no. 6, Article ID 858, 2023. DOI: <https://doi.org/10.1007/s42979-023-02330-x>.
- [6] "Monthly share of spam in the total e-mail traffic worldwide from January 2014 to December 2023," Available: <https://www.statista.com/statistics/420391/spam-email-traffic-share/>.
- [7] EmailToolTester, "EmailToolTester Deliverability Audit," Available: <https://www.emailtooltester.com/en/email-deliverability-audit/>.
- [8] Trend Micro, "Email Threat Landscape Report: Cybercriminal Tactics and Techniques That Organizations Need to Know," Available: <https://www.trendmicro.com/vinfo/us/security/research-and-analysis/threat-reports/roundup/annual-trend-micro-email-threats-report>.
- [9] C. N. Mohammed and A. M. Ahmed, "A semantic-based model with a hybrid feature engineering process for accurate spam detection," *Journal of Electrical Systems and Information Technology*, vol. 11, no. 1, Article ID 26, 2024.
- [10] Z. B. Siddique, M. A. Khan, I. U. Din, A. Almogren, I. Mohiuddin, and S. Nazir, "Machine learning-based detection of spam emails," *Scientific Programming*, vol. 2021, no. 1, Article ID 6508784, 2021.
- [11] A. J. Saleh, A. Karim, B. Shanmugam, S. Azam, K. Kannoorpatti, M. Jonkman, and F. De Boer, "An intelligent spam detection model based on artificial immune system," *Information*, vol. 10, no. 6, Article ID 209, 2019.
- [12] K. I. Roumeliotis, N. D. Tselikas, and D. K. Nasiopoulos, "Next-generation spam filtering: Comparative fine-tuning of LLMs, NLP, and CNN models for email spam classification," *Electronics*, vol. 13, no. 11, Article ID 2034, 2024.
- [13] E. John-Africa and V. T. Emmah, "Performance evaluation of LSTM and RNN models in the detection of email spam messages," *European Journal of Information Technologies and Computer Science*, vol. 2, no. 6, pp. 24-30, 2022.
- [14] R. Indu and S. C. Dimri, "Detecting spam e-mails with content and weight-based binomial logistic model," *Journal of Web Engineering*, vol. 22, no. 7, pp. 939-959, 2023.
- [15] A. Sheneamer, "Comparison of deep and traditional learning methods for email spam filtering," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 1, pp. 1-6, 2021.
- [16] A. M. Bakare, K. S. M. Anbananthan, S. Muthaiyah, J. Krishnan, and S. Kannan, "Punctuation restoration with transformer model on social media data," *Applied Sciences*, vol. 13, no. 3, p. 1685, 2023.
- [17] G. Grefenstette, "Tokenization," in *Syntactic Wordclass Tagging*, Dordrecht, Netherlands: Springer, 1999, pp. 117- 133.

- [18] S. J. Mielke, Z. Alyafeai, E. Salesky, C. Raffel, M. Dey, M. Galle', A. Raja, *et al.*, "Between words and characters: A brief history of open-vocabulary modeling and tokenization in NLP," *arXiv preprint arXiv:2112.10508*, 2021.
- [19] S. Sarica and J. Luo, "Stopwords in technical language processing," *PLOS One*, vol. 16, no. 8, p. e0254937, 2021.
- [20] V. K. Pant, R. Sharma, and S. Kundu, "An overview of stemming and lemmatization techniques," in *Advances in Networks, Intelligence and Computing*, pp. 308-321.
- [21] T. Turki and S. S. Roy, "Novel hate speech detection using word cloud visualization and ensemble learning coupled with count vectorizer," *Applied Sciences*, vol. 12, no. 13, p. 6611, 2022.
- [22] G. M. Raza, Z. S. Butt, S. Latif, and A. Wahid, "Sentiment analysis on COVID tweets: An experimental analysis on the impact of count vectorizer and TF-IDF on sentiment predictions using deep learning models," in *Proc. 2021 Int. Conf. Digital Futures and Transformative Technologies (ICoDT2)*, pp. 1-6, 2021.
- [23] M. M. Danyal, S. S. Khan, M. Khan, S. Ullah, M. B. Ghaffar, and W. Khan, "Sentiment analysis of movie reviews based on NB approaches using TF-IDF and count vectorizer," *Social Netw. Anal. Mining*, vol. 14, no. 1, pp. 1-15, 2024.
- [24] S. Wehnert, V. Sudhi, S. Dureja, L. Kutty, S. Shahania, and E. W. De Luca, "Legal norm retrieval with variations of the BERT model combined with TF-IDF vectorization," in *Proc. 18th Int. Conf. Artif. Intell. Law*, pp. 285-294, 2021.
- [25] T. G. Nick and K. M. Campbell, "Logistic regression," *Topics Biostatistics*, pp. 273-301, 2007.
- [26] J. R. Quinlan, "Induction of decision trees," *Machine Learn.*, vol. 1, pp. 81-106, 1986.
- [27] S. Suthaharan and S. Suthaharan, "Support vector machine," in *Machine Learn. Models Algorithms Big Data Classif.: Thinking Examples Effective Learn.*, pp. 207-235, 2016.
- [28] D. A. Pisner and D. M. Schnyer, "Support vector machine," in *Machine Learn.*, pp. 101-121, Academic Press, 2020.
- [29] S. Amari, "Backpropagation and stochastic gradient descent method," *Neurocomputing*, vol. 5, no. 4-5, pp. 185-196, 1993.
- [30] L. Bottou, "Stochastic gradient descent tricks," in *Neural Networks: Tricks of the Trade: Second Edition*, pp. 421- 436, Springer Berlin Heidelberg, 2012.
- [31] F. Divina, A. Gilson, F. Gome'z-Vela, M. Garc'ia Torres, and J. F. Torres, "Stacking ensemble learning for short-term electricity consumption forecasting," *Energies*, vol. 11, no. 4, p. 949, 2018.
- [32] S. Rajagopal, P. P. Kundapur, and K. S. Hareesha, "A stacking ensemble for network intrusion detection using heterogeneous datasets," *Security and Communication Networks*, vol. 2020, no. 1, p. 4586875, 2020.
- [33] A. Dogan and D. Birant, "A weighted majority voting ensemble approach for classification," in *2019 4th International Conference on Computer Science and Engineering (UBMK)*, pp. 1-6, 2019.
- [34] K. Raza, "Improving the prediction accuracy of heart disease with ensemble learning and majority voting rule," in *U-Healthcare Monitoring Systems*, pp. 179-196, Academic Press, 2019.
- [35] P. Singhvi, "Spam Email Classification Dataset," Available: <https://www.kaggle.com/datasets/purusinghvi/email-spam-classification-dataset>, 2024.
- [36] F. Zouak, O. El Beqqali, and J. Riffi, "BERT-GraphSAGE: hybrid approach to spam detection," *Journal of Big Data*, vol. 12, no. 1, p. 128, 2025.

Authors' Profiles



Dharmaraj R. Patil earned his Ph.D. in computer engineering from Kavayitri Bhahinabai Chaudhari North Maharashtra University, Jalgaon, Maharashtra, India, and his master's degree in computer science and engineering from Government College of Engineering, Aurangabad, Maharashtra, India. He currently serves as an associate professor in the Department of Computer Engineering at the R.C. Patel Institute of Technology in Shirpur, Maharashtra, India, where he has been teaching for over twenty years. His research interests include web mining, intrusion detection, and web security. He has authored numerous papers published in various journals and presented at both international and national conferences.

How to cite this paper:

Dharmaraj R. Patil, "Real-time Spam Email Filtering with Stacking and Majority Voting Ensembles", *International Journal of Computer Network and Information Security(IJCNIS)*, Vol.18, No.4, pp. 169-186, 2026. DOI:10.5815/ijcnis.2026.04.09