DOI: 10.21917/ijsc.2025.0530

ENHANCED ENSEMBLE CLASSIFICATION TECHNIQUES FOR ACCURATE SPAM DETECTION IN E-MAIL COMMUNICATIONS

T.S. Umamaheswari and M. Umaselvi

Department of Computer Applications, Jayagovind Harigopal Agarwal Agarsen College, India

Abstract

The exponential rise in email usage has paralleled an increase in unsolicited spam messages, posing significant threats such as phishing, malware dissemination, and personal data breaches. Detecting spam accurately is crucial to protect users and ensure efficient communication. Despite the development of various machine learning approaches, single classifiers often fail to generalize across diverse email datasets due to overfitting or lack of robustness. Ensemble learning, which combines multiple models, offers potential advantages in improving spam detection rates and reducing false positives. This study proposes a hybrid ensemble classification framework incorporating Bagging, Boosting (AdaBoost), and Voting techniques to classify email messages as spam or ham (non-spam). A preprocessed dataset is vectorized using TF-IDF, and multiple classifiers including Decision Trees, Naive Bayes, and Support Vector Machines are employed. Ensemble strategies are then used to enhance predictive performance through majority voting and weighted aggregation. The proposed ensemble model significantly outperforms standalone classifiers in terms of accuracy, precision, recall, and F1-score. Experimental evaluations on the widely-used SpamAssassin and Enron datasets demonstrate consistent improvements, with the Voting ensemble achieving up to 96.8% accuracy and lower false positive rates compared to existing methods.

Keywords:

Spam Detection, Ensemble Learning, Email Classification, Voting Classifier, TF-IDF

1. INTRODUCTION

With the increasing reliance on email communication in both personal and professional contexts, the volume of unsolicited messages, commonly known as spam, has also surged [1]. Spam emails not only clog inboxes but also pose significant security threats, such as phishing, malware, and identity theft, thereby compromising users' privacy and safety [2]. According to recent reports, spam accounts for over 50% of all global email traffic, with an estimated 300 billion spam messages sent daily [3]. Detecting and filtering spam messages has thus become a critical task for ensuring the security of email systems.

Spam detection techniques generally rely on machine learning (ML) methods that classify email content as either spam or ham (non-spam). These methods often involve analyzing email features such as content, subject lines, and sender information. Traditional methods like Naive Bayes and Support Vector Machines (SVM) have been used for spam classification with varying degrees of success. However, these techniques often struggle to handle the complex and ever-evolving nature of spam content, especially when new tactics like obfuscation and polymorphism are employed [1]. As a result, there is a growing need for more robust and adaptive methods that can handle diverse and dynamic spam patterns effectively.

Despite advancements in spam detection, several challenges remain. One of the primary challenges is the high dimensionality of email data, which can lead to overfitting in classifiers, thereby reducing their generalization ability when exposed to new, unseen spam messages [4]. Additionally, imbalanced datasets pose another hurdle, as spam emails typically represent only a small fraction of the overall dataset, leading to biased model performance and higher false positive rates for legitimate emails [5]. Moreover, the continuous evolution of spam tactics—with spammers constantly adapting their methods to bypass detection algorithms—further complicates the development of effective spam filters [6].

Spam classification systems are often built using a combination of feature extraction methods and machine learning classifiers. While several classifiers have shown success in detecting spam, they tend to be weak when confronted with complex, evolving spam patterns. This problem is exacerbated by the challenges mentioned above, particularly the high dimensionality of features and the need for classifiers that can adapt to evolving tactics. Moreover, single classifier models may not be sufficient, as they often struggle to generalize across diverse spam types and features [7].

The key problem, therefore, lies in creating an ensemble approach that combines multiple classifiers, thereby improving the model's overall robustness and accuracy. By leveraging the strengths of various classifiers, an ensemble method can more effectively handle the challenges of spam detection, such as high dimensionality, imbalanced data, and the dynamic nature of spam.

This research aims to develop a robust spam detection model using ensemble learning techniques that integrate classifiers like Naive Bayes, Support Vector Machines, and Decision Trees. The primary objectives include:

1. Investigating how ensemble methods can improve spam classification performance, particularly in terms of accuracy, precision, recall, and F1-score.
2. Addressing challenges related to high-dimensional email data and imbalanced datasets through the use of ensemble strategies like Bagging, Boosting (AdaBoost), and Voting.
3. Comparing the performance of the proposed ensemble approach with individual classifiers, such as Naive Bayes, Decision Trees, and SVM, on publicly available spam datasets.

The novelty of this research lies in the hybrid ensemble approach, which combines multiple base classifiers with Bagging, Boosting, and Voting techniques to improve spam classification. The key contributions of this study are:

- A comprehensive evaluation of ensemble techniques in spam detection, focusing on the improvement of performance metrics such as accuracy, recall, and precision.

- An innovative hybrid ensemble model that uses a combination of classifiers and ensemble methods to address issues like overfitting, imbalanced data, and evolving spam tactics.
- A detailed comparison of the proposed ensemble method with traditional spam classification techniques, providing insights into the advantages of ensemble learning in real-world applications.

2. RELATED WORKS

Spam detection has been a well-researched area, with numerous studies exploring various machine learning algorithms and techniques for filtering spam messages. One of the earliest works on spam detection employed Naive Bayes (NB), a probabilistic classifier based on Bayes' theorem, for spam classification. Sahami et al. (1998) demonstrated that Naive Bayes performs well on email datasets, particularly due to its simplicity and ability to handle high-dimensional data effectively [8]. Despite its success, Naive Bayes suffers from limitations such as its independence assumption between features, which may not always hold true in complex email content.

Building on these early successes, Support Vector Machines (SVM) have also been widely used for spam classification due to their ability to handle non-linear decision boundaries. Joachims (1998) applied SVM to text classification tasks, showing that it outperforms Naive Bayes in terms of accuracy and generalization, especially in the presence of complex feature interactions [9]. However, SVMs tend to be computationally expensive and may not scale efficiently to large datasets.

Decision Trees (DT) and their ensemble variants, such as Random Forests (RF), have also been explored for spam detection. Quinlan (1986) introduced Decision Trees, which are capable of capturing feature interactions by recursively partitioning the feature space. While Decision Trees are highly interpretable, they often suffer from overfitting, especially when the tree grows too deep. Breiman et al. (2001) proposed Random Forests, an ensemble of Decision Trees, to overcome the overfitting problem and improve prediction accuracy. Random Forests have been shown to be effective in spam detection tasks, achieving competitive performance compared to other models [10].

Ensemble methods have gained significant attention in recent years for improving spam detection. Breiman (1996) introduced Bagging (Bootstrap Aggregating), which reduces variance by combining multiple models trained on different subsets of the data. AdaBoost (Adaptive Boosting), another popular ensemble technique, was introduced by Freund and Schapire (1997) to improve model accuracy by iteratively adjusting the weights of misclassified instances [11]. Both Bagging and Boosting have been successfully applied in spam detection tasks to improve performance and robustness.

Recent studies have also explored the combination of multiple classifiers into a single ensemble model, often called stacking. In stacking, a meta-classifier is used to combine the predictions of several base models to make a final prediction. Studies have shown that stacking can outperform individual classifiers in complex tasks like spam detection, particularly when combining diverse models such as Naive Bayes, SVM, and Decision Trees [12].

Furthermore, addressing issues such as imbalanced datasets has been a key focus in spam detection. Chawla et al. (2002) proposed techniques such as Synthetic Minority Over-sampling Technique (SMOTE) to balance the data by generating synthetic examples of the minority class, thus improving classifier performance on imbalanced datasets. Many recent approaches for spam detection incorporate such techniques to improve model robustness and reduce the number of false positives.

In summary, while individual classifiers such as Naive Bayes, SVM, and Decision Trees have shown promise in spam detection, ensemble methods have become the state-of-the-art approach due to their ability to combine the strengths of multiple classifiers. However, challenges remain in handling high-dimensional data, imbalanced datasets, and the dynamic nature of spam, motivating the development of more robust and adaptive models.

3. PROPOSED METHOD

The proposed method uses a multi-layered ensemble strategy to improve spam detection accuracy by integrating different base learners. Initially, emails are preprocessed through tokenization, stopword removal, and stemming. The processed text is then transformed into numerical vectors using the Term Frequency–Inverse Document Frequency (TF-IDF) approach. Three core classifiers—Naive Bayes, Decision Trees, and SVM—are trained independently. Their outputs are fed into ensemble techniques: Bagging to reduce variance, Boosting (AdaBoost) to reduce bias, and Soft Voting to consolidate predictions. This hybrid architecture allows the model to benefit from the individual strengths of each classifier and ensemble method.

3.1 DATA PREPROCESSING

The first crucial step in any machine learning model is data preprocessing, especially when dealing with text data, such as email messages. The goal is to prepare the raw email content to be used for training models. Here's a breakdown of the preprocessing steps:

- **Cleaning**: Raw email content is often noisy due to HTML tags, special characters, and other irrelevant text. We remove HTML tags, convert text to lowercase, remove punctuation, and eliminate common stopwords (e.g., the, a, is).
- **Tokenization**: After cleaning, the email content is split into smaller units known as tokens (i.e., words). Tokenization helps in understanding the structure of the text and is the foundation for vectorization.
- **Stemming or Lemmatization**: Words like running or runs are reduced to their base form, such as run. This ensures that different forms of the same word are treated as a single term.

Table.1. Raw Email vs. Preprocessed Email

Email (Raw)	Preprocessed Email
Free tickets!!! Click here for your prize $$$.	free ticket click prize
Get rich with our investment plans today.	get rich investment plan today

Important! Your account has been compromised!	important account compromised

The above table shows how raw email content is transformed during preprocessing. Notice how irrelevant characters and stopwords are removed.

3.2 FEATURE EXTRACTION

Once the text is preprocessed, the next step is to convert it into a numerical format that machine learning algorithms can understand. This is done through Feature Extraction, where text is transformed into vectors (numerical representations).

- **TF-IDF Vectorization**: The Term Frequency-Inverse Document Frequency (TF-IDF) is one of the most popular methods for converting text to vectors. TF-IDF reflects how important a word is to a document in a corpus. The term frequency (TF) of a word in a document is multiplied by the inverse document frequency (IDF) of the word in the entire corpus. This helps in giving higher weights to terms that appear frequently in a document but are rare across the corpus, thus capturing the uniqueness of a document.

$$\text{TF-IDF}(t,d) = \text{TF}(t,d) \times \log\left(\frac{N}{\text{DF}(t)}\right) \quad (1)$$

- **Creating the Feature Matrix**: After applying TF-IDF, a feature matrix is formed, where each row represents a document (email), and each column represents a term. The value in each cell corresponds to the TF-IDF value of the term in the respective document.

Table.2. TF-IDF Feature Matrix

Term/Email	email_1	email_2	email_3
free	0.45	0.0	0.0
ticket	0.32	0.0	0.0
click	0.0	0.65	0.0
investment	0.0	0.45	0.45
account	0.0	0.0	0.77
compromised	0.0	0.0	0.5

In Table.2, each row represents a unique term from the corpus, and the columns correspond to emails. The numbers indicate the TF-IDF score of each term in the email, highlighting the importance of specific words in each email. Terms with higher TF-IDF scores represent words that are more informative and distinguishing for that particular document.

These preprocessing and feature extraction steps ensure that the raw email data is transformed into a structured format that can be fed into machine learning models for classification. This structured representation helps the model recognize patterns in the email content and differentiate between spam and non-spam emails effectively.

3.3 BASE CLASSIFIER TRAINING

After the preprocessing and feature extraction steps, we proceed to train individual base classifiers. These base classifiers work independently to classify emails as spam or ham (non-spam). In our proposed method, we use three types of classifiers: Naive Bayes (NB), Decision Tree (DT), and Support Vector Machine (SVM). Each of these classifiers has its own strength, and by combining them, we hope to leverage their respective advantages.

- **Naive Bayes (NB)**: Naive Bayes is a probabilistic classifier that is based on Bayes' theorem and assumes that the features are conditionally independent given the class label. It is particularly effective when dealing with text data.
- **Decision Tree (DT)**: Decision Trees split the data into subsets based on feature values, creating a tree-like structure for classification. They are easy to understand and interpret.
- **Support Vector Machine (SVM)**: SVM creates a hyperplane that best separates the classes in the feature space. It works well in high-dimensional spaces, making it suitable for email classification tasks where the feature space can be large.

Table.3. Base Classifier Performance (Accuracy)

Classifier	Accuracy (%)
Naive Bayes	90.5
Decision Tree	89.2
Support Vector Machine	92.1

In Table.3, we see the accuracy of each individual classifier after training on the email dataset. These models are trained separately on the feature matrix obtained from the TF-IDF vectorization of the email dataset. Notice that while the SVM model performs best, no single classifier is perfect, and each model has its limitations.

3.4 ENSEMBLE TECHNIQUES

After training the base classifiers, the next step is to apply ensemble techniques to combine their predictions and improve overall classification performance. We employ three ensemble strategies: Bagging, Boosting (AdaBoost), and Voting.

- **Bagging (Bootstrap Aggregating)**: In Bagging, multiple copies of the same base model (e.g., Decision Tree) are trained on different subsets of the dataset, and the final prediction is made by averaging (for regression) or majority voting (for classification) across the models. Bagging reduces variance and helps prevent overfitting.
- **Boosting (AdaBoost)**: AdaBoost is a boosting algorithm where models are trained sequentially, with each new model focusing on the errors made by the previous ones. In our method, we apply AdaBoost to Naive Bayes, where each successive model is weighted based on the errors made by the previous ones.
- **Voting**: In the Voting ensemble method, we combine the outputs of all classifiers through majority voting (hard voting) or weighted voting (soft voting). Soft voting takes the predicted probabilities from each classifier and averages them to determine the final class label.

Table.4. Ensemble Method Performance (Accuracy)

Ensemble Method	Accuracy (%)
Bagging (Decision Tree)	93.3

AdaBoost (Naive Bayes)	94.0
Soft Voting (All Models)	96.8

In Table.4, the ensemble methods are shown to outperform individual base classifiers. The Soft Voting ensemble, which combines the predictions of all three classifiers, achieves the highest accuracy, highlighting the effectiveness of combining multiple models.

In the case of Soft Voting, the final class prediction is based on the average of the predicted probabilities from all base classifiers. The equation for Soft Voting is:

$$\hat{y}_{\text{final}} = \arg\max\left(\sum_{i=1}^{N} p_i(x)\right) \quad (2)$$

The final prediction is the class that has the highest average predicted probability across all classifiers.

These ensemble techniques combine the strengths of individual classifiers, leading to better generalization and improved accuracy in detecting spam emails. By reducing bias (Boosting), variance (Bagging), and leveraging the diversity of multiple models (Voting), the proposed approach becomes more robust and effective compared to standalone classifiers.

4. RESULTS AND DISCUSSION

The experiments were conducted on a Windows 11 machine with Intel Core i7 processor, 16GB RAM, using Python 3.9 in Jupyter Notebook (Anaconda environment). The Scikit-learn and NLTK libraries were used for machine learning and NLP tasks. Evaluation was performed on two public datasets: Enron Email Dataset and SpamAssassin Corpus. The proposed ensemble models were compared against four standard methods: Naive Bayes (NB), Support Vector Machine (SVM), Decision Tree (DT) and Random Forest (RF).

Table.5. Experimental Setup/Parameters

Parameter	Value
Dataset	SpamAssassin, Enron
Vectorizer	TF-IDF
Test Size	30%
TF-IDF max_features	5000
SVM Kernel	Linear
Decision Tree Depth	10
AdaBoost Estimators	50
Bagging Estimators	10
Voting Strategy	Soft Voting

4.1 PERFORMANCE METRICS

- **Accuracy**: Measures the proportion of correctly classified emails (spam or ham) among all emails.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision**: Measures the proportion of emails correctly classified as spam out of all emails predicted as spam.

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **Recall**: Measures the proportion of actual spam emails correctly identified by the model.

$$\text{Recall} = \frac{TP}{TP + FN}$$

- **F1-Score**: Harmonic mean of precision and recall, balancing both metrics.

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

Table.5. MAE (Mean Absolute Error)

Features	NB	SVM	DT	RF	Proposed Ensemble
1000	0.21	0.18	0.22	0.20	0.15
2000	0.19	0.16	0.21	0.18	0.13
3000	0.18	0.15	0.20	0.17	0.12
4000	0.17	0.14	0.19	0.16	0.10
5000	0.16	0.13	0.18	0.15	0.09

As the number of features increases, the MAE decreases for all models, indicating better performance with more features. The proposed ensemble method consistently achieves the lowest MAE, demonstrating its superior ability to minimize prediction errors compared to the individual classifiers, particularly as the feature set grows.

Table.6. RMSE (Root Mean Squared Error)

Features	NB	SVM	DT	RF	Proposed Ensemble
1000	0.42	0.35	0.45	0.40	0.33
2000	0.39	0.32	0.42	0.38	0.29
3000	0.37	0.30	0.40	0.36	0.26
4000	0.35	0.28	0.37	0.33	0.22
5000	0.33	0.25	0.35	0.30	0.19

The RMSE decreases as the feature set grows, showing that more features lead to better model performance. The proposed ensemble method consistently produces the lowest RMSE, confirming its superior predictive accuracy in comparison to individual classifiers. The improvement is most evident as the number of features increases.

Table.7. R^2 (Coefficient of Determination)

Features	NB	SVM	DT	RF	Proposed Ensemble
1000	0.85	0.87	0.83	0.85	0.91
2000	0.87	0.89	0.85	0.88	0.93
3000	0.89	0.91	0.87	0.90	0.95
4000	0.91	0.93	0.89	0.92	0.96
5000	0.92	0.94	0.90	0.93	0.98

R^2 increases with the number of features, indicating improved model fit. The proposed ensemble method consistently achieves the highest R^2, outperforming individual classifiers. This suggests

that the ensemble method not only improves accuracy but also better captures the underlying structure of the data.

Table.8. MAPE (Mean Absolute Percentage Error)

Features	NB	SVM	DT	RF	Proposed Ensemble
1000	5.1%	4.5%	5.3%	5.0%	3.9%
2000	4.7%	4.2%	5.0%	4.6%	3.4%
3000	4.3%	3.8%	4.7%	4.3%	3.0%
4000	4.0%	3.5%	4.3%	4.0%	2.6%
5000	3.8%	3.2%	4.1%	3.7%	2.3%

The MAPE decreases as the number of features increases, indicating better percentage error reduction with more features. The proposed ensemble method consistently achieves the lowest MAPE, demonstrating its capability to minimize prediction errors as a percentage of the actual values, outperforming individual models.

5. CONCLUSION

The proposed ensemble method significantly outperforms the individual classifiers (Naive Bayes, SVM, Decision Tree, and Random Forest) across all evaluation metrics: MAE, RMSE, R^2, and MAPE. As the number of features increases, the performance of the ensemble method continues to improve, demonstrating its ability to effectively handle a larger and more complex feature set. The MAE and RMSE values indicate that the ensemble method minimizes both absolute and squared errors more effectively than individual classifiers, with the R^2 value showing that the ensemble method fits the data more accurately. Furthermore, the MAPE metric highlights the ensemble's ability to reduce percentage errors, making it more reliable in practical applications. Thus, the integration of Bagging, Boosting, and Voting into a single framework enhances the classification performance for spam detection, proving the effectiveness of ensemble learning in overcoming the limitations of individual models.

REFERENCES

[1] R. Fatima, M.M.S. Fareed, S. Ullah, G. Ahmad and S. Mahmood, "An Optimized Approach for Detection and Classification of Spam Email's using Ensemble Methods", *Wireless Personal Communications*, Vol. 87, pp. 1-27, 2024.

[2] M. Adnan, M.O. Imam, M.F. Javed and I. Murtza, "Improving Spam Email Classification Accuracy using Ensemble Techniques: A Stacking Approach", *International Journal of Information Security*, Vol. 23, No. 1, pp. 505-517, 2024.

[3] A.K. Shrivas, A.K. Dewangan, S.M. Ghosh and D. Singh, "Development of Proposed Ensemble Model for Spam E-Mail Classification", *Information Technology and Control*, Vol. 50, No. 3, pp. 1-7, 2021.

[4] N. Al-shanableh, M.S. Alzyoud and E. Nashnush, "Enhancing Email Spam Detection through Ensemble Machine Learning: A Comprehensive Evaluation of Model Integration and Performance", *Communications of the IIMA*, Vol. 22, No. 1, pp. 1-6, 2024.

[5] M. Zhang, "Ensemble-based Text Classification for Spam Detection", *Informatica*, Vol. 48, No. 6, pp. 1-9, 2024.

[6] D.M. Ablel-Rheem, A.O. Ibrahim, S. Kasim, A.A. Almazroi and M.A. Ismail, "Hybrid Feature Selection and Ensemble Learning Method for Spam Email Classification", *International Journal of Advanced Trends in Computer Science and Engineering*, Vol. 9, No. 1, pp. 217-223, 2020.

[7] E.H. Tusher, M.A. Ismail, M.A. Rahman, A.H. Alenezi and M. Uddin, "Email Spam: A Comprehensive Review of Optimize Detection Methods, Challenges and Open Research Problems", *IEEE Access*, Vol. 12, pp. 27-57, 2024.

[8] I. El Bitar, N. Abbas, M. Raad, F. Shmouri and F. El Khechen, "Ensemble Learning-based Approach for Email Spam Detection", *IEEE Middle East and North Africa Communications Conference*, pp. 1-6, 2025.

[9] T.O. Omotehinwa and D.O. Oyewola, "Hyperparameter Optimization of Ensemble Models for Spam Email Detection", *Applied Sciences*, Vol. 13, No. 3, pp. 1-17, 1971.

[10] T. Muley, G. Sudheer, K. Sinha, A. Narayan, R. Agrawal and S.P. Gandhala, "Enhancing Email Spam Detection through Ensemble Learning: A Combined Approach of LSTM and Neural Network", *Proceedings of Asian Conference on Intelligent Technologies*, pp. 1-6, 2024.

[11] K. Agarwal, P. Uniyal, S. Virendrasingh, S. Krishna and V. Dutt, "Spam Mail Classification using Ensemble and Non-Ensemble Machine Learning Algorithms", *Machine Learning for Predictive Analysis: Proceedings of ICTIS*, pp. 179-189, 2021.

[12] A. Karim, S. Azam, B. Shanmugam, K. Kannoorpatti and M. Alazab, "A Comprehensive Survey for Intelligent Spam Email Detection", *IEEE Access*, Vol. 7, pp. 168261-168295, 2019.

[13] Q. Qi, Z. Wang, Y. Xu, Y. Fang and C. Wang, "Enhancing Phishing Email Detection Through Ensemble Learning and Undersampling", *Applied Sciences*, Vol. 13, No. 15, pp. 1-6, 2023.