International Journal of Applied Mathematics

Volume 38 No. 6s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

SPAMNET: A HYBRID DEEP LEARNING FRAMEWORK FOR ROBUST SPAM EMAIL DETECTION USING MULTI-MODAL FEATURES

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

[1]Computer Engineering Department, K K Wagh Institute of Research and Technology, SPPU, Pune, Nashik, India

kalyaniubale110@gmail.com

[2]Computer Engineering Department, Sandip Institute of Engineering and Management, SPPU, Pune, Nashik, India

kamini.nalavade@siem.org.in

Abstract

Spam emails are still making it hard for people to use digital communication systems. They pose security risks and slow down operations. To solve this problem, we are proposing SpamNet, a mixed deep learning model that uses multi-modal feature learning to improve the accuracy of spam identification. SpamNet is different from other methods as it uses a multi-branch design that looks at sequential language patterns, statistical metadata features, and time-based behavioural trends all at the same time. The model is trained and tested on a balanced, publicly available email dataset with 6000 samples that have an equal number of spam and ham emails. The result obtained results during experiments show that SpamNet has an 98.81% validation accuracy, which is a much better than standard deep learning models like DNN, LSTM, and CNN. SpamNet also has lower rates of fake positives and false negatives, which makes it more reliable and stronger in generalisation. The study proves that SpamNet is a useful, small, and scalable approach that can be used in real-time spam filtering systems in a variety of communication settings.

Keywords. Spam detection, hybrid deep learning, SpamNet, multi-branch architecture, LSTM, CNN, metadata features, real-time email classification.

I. Introduction

Email has become one of the most common ways to communicate electronically in this digital age. It is an essential tool for personal, educational, and business exchanges. With the ease and speed of email contact and the rapid growth of internet connections, the way people and businesses share information has changed. But this progress in technology has also brought about big problems with security, most notably the rise of spam emails. A lot of people get spam emails, which are usually unwanted or unrelated messages sent in bulk to a lot of people. Spam emails have become a big part of all email traffic around the world [1]. Industry figures say that more than half of all email traffic around the world is spam. This clutters up communication channels, makes storage more expensive, and poses serious security risks like phishing, malware spreading, and fraud schemes. Beyond just being annoying, spam has a lot of bad effects. Spam filtering equipment, wasted bandwidth, and employees who are less productive because of spam emails all cause operational costs to rise for businesses around the world [2]. Also, spam emails are often used as a way to launch complex cyberattacks that target private and sensitive business and personal data. As spam email campaigns get more complicated, it gets harder to spot them. These campaigns use tricks like social engineering, email spoofing, and content deception to trick people [3]. Because of this, making spam detection systems that are strong, smart, and flexible has become an important area of study in the fields of cybersecurity and artificial intelligence (AI).

Rule-based filters and simple machine learning classifiers are two examples of traditional ways to find spam that have been used in many spam filtering solutions. Rule-based systems, like heuristic filters and blacklisting, find spam by looking for trends or keywords that have already been set up. However, these methods have a lot of problems, such as high rates of false-positives, trouble adapting to changing spam trends, and the need for constant

Received: August 03, 2025

International Journal of Applied Mathematics
Volume 38 No. 6s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

manual updates. In the same way, traditional machine learning methods like Naïve Bayes [4], Support Vector Machines (SVM) [5], and Decision Trees [7] have had some success using the statistical traits of email features. Still, these models have trouble with handling large amounts of data, duplicate features, and the changing nature of spam emails, which often leads to lower classification accuracy in real life.

Deep learning has changed many areas of artificial intelligence in recent years. It is especially good at systems that involve complex data structures, like natural language processing (NLP). Deep learning architectures like Convolutional Neural Networks (CNN) [6], Long Short-Term Memory networks (LSTM) [5], and hybrid neural networks are very good at finding complex patterns in big datasets without having to do a lot of feature engineering by hand. Deep networks can effectively pick up on syntactic and semantic subtleties in text data because they learn in a hierarchical way [8]. This makes them perfect for jobs like finding spam.

Even though deep learning methods have been successful, there are still some problems that need to be fixed before spam detection models can work well. One big problem with email datasets is that they often have a lot of high-dimensional, redundant, and unimportant features. These can cause overfitting, higher computational costs, and models that are harder to understand. Also, spam emails usually have many parts, like the email body, the subject line, sender information, timestamps, and embedded links. This means that multi-modal features need to be used for full analysis. Most of the research that has been done so far has been on either text-based features or metadata. Not much has been done on hybrid models that use both kinds of data to improve classification accuracy [9].

This study suggests making and using a Dynamic Classification Framework for Email Spam Detection that uses advanced deep learning architectures and feature integration methods to deal with these problems. SpamNet is the name of the suggested framework's hybrid multi-branch neural network. It cleverly combines textual features processed through LSTM/CNN architectures [4][8] with numerical and metadata features processed through dense neural layers. This architecture has multiple inputs and is meant to pick up both sequential dependencies in email text and structured patterns in metadata. This provides a complete way to sort spam emails.

The dataset used in this study comes from Kaggle's freely available spam email dataset [19], which has a variety of emails that have been labelled as spam or ham. The dataset has important information like email text, information about the sender and receiver, and information about time, which makes it a good base for multi-modal feature engineering. A planned data preprocessing method is used that includes cleaning the text, tokenising it, padding it, label encoding of categorical variables, and normalisation of numerical features to make sure that deep learning models get high-quality input.

This study not only suggests SpamNet, but it also compares several other models, such as traditional Random Forest classifiers [3], feedforward Dense Neural Networks (DNN) [14], standalone LSTM and CNN architectures [5][13], and the suggested hybrid model. Performance measures like recall, F1-score, accuracy, and loss are used to carefully test the models and give a full picture of how well they can classify things. Visualisation tools, such as confusion matrices, accuracy-loss curves, and comparative graphs, make it easier to understand and analyse the data.

One unique thing about this study is that it uses multi-level feature consolidation. This means that the raw email text is turned into tokenised sequences and then mixed with normalised metadata features. Previous studies either only looked at text embeddings or used general methods for choosing features. This approach is different. Previous research, like GWO-BERT, has used advanced feature selection methods and transformer embeddings to make classification work better, but most of their attention has been on text-only features. The suggested SpamNet framework, on the other hand, takes a broader view, combining different sources of features and checking how they affect spam classification as a whole.

The major contributions of this research are as:

- A detailed review of existing machine learning and deep learning techniques for spam detection, identifying the gaps in feature selection and multi-modal data handling.
- Development of a dynamic multi-input hybrid model, SpamNet, for efficient spam detection combining both textual and non-textual features.

Received: August 03, 2025

International Journal of Applied Mathematics

Volume 38 No. 6s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

- A robust experimental setup using the Kaggle spam dataset with comprehensive preprocessing and feature consolidation pipelines.
- Comparative evaluation of classical ML, standard deep learning, and the proposed hybrid deep learning models using diverse performance metrics.
- An in-depth analysis of the strengths and limitations of different models, providing insights into future directions for improving spam detection systems.

This is how the paper is put together: In Section 2, a thorough literature review is given that talks about both old and new ways of finding spam, how to choose which features to use, and the latest progress in using deep learning to block spam. In Section 3, the proposed method is explained in more detail, including a description of the dataset, steps for preprocessing it, and information about the model design. Section 4 explains, how the experiment was set up and how it was judged. Section 5 talks about the results and how they were compared. In Section 6, the study comes to an end by outlining its main findings, contributions, limitations, and possible future research paths. This study is used to make spam email detection much better by using a multi-branch deep learning model and a variety of feature modalities. It will do this by providing a more accurate, flexible, and reliable solution that can be used in environments where email communication is always changing.

II. Literature Review

The work of finding email spam has changed a lot in the last twenty years, going from simple rule-based systems to complex models based on machine learning and deep learning. Due to the large number of unsolicited emails, academics have been focussing on making strong classification systems that reduce false positives and increase the accuracy of detection. To support the suggested research, this section carefully looks at traditional methods, deep learning architectures, feature selection techniques, word embedding strategies, and new hybrid approaches. It does this by pointing out their pros and cons.

A. Traditional Machine Learning Approaches for Spam Detection

In the beginning, most studies on finding spam used basic machine learning methods that focused on pulling out features from email headers and body content. During this time, Naïve Bayes (NB), Support Vector Machines (SVM), Decision Trees (DT), and Random Forest (RF) algorithms were the most popular. Naïve Bayes became popular quickly because it was easy to use and worked well. It did this by taking email classification as a probabilistic inference problem. Studies like the ones by Subramaniam et al. showed that NB could get good results, especially when datasets were small [7].

Support Vector Machines found the best hyperplanes between the spam and non-spam classes, which made classification margins even better. Using linear and radial basis kernels, Drucker et al. found that SVM was accurate about 95% of the time [8]. People became interested in ensemble classifiers like Random Forest because they are strong. They do this by mixing multiple decision trees to lower overfitting and variance. In their study, DeBarr and Wechsler showed that RF works well in custom spam detection datasets, getting about 94% success [9].

Even with these successes, standard machine learning methods had some problems. These models relied on manual feature engineering, which meant that experts in the field were needed to pick out distinguishing traits from email metadata or text. Also, classical classifiers had trouble handling big or high-dimensional datasets, and they had a hard time adapting to changing spam patterns, which made them less accurate in real-world settings.

B. Deep Learning Methods for Spam Filtering

Deep learning changed the way spam is found by automating the extraction of features and the learning of complicated patterns from data. It has been shown that Convolutional Neural Networks (CNNs), Long Short-Term Memory Networks (LSTMs), and hybrid deep networks work better than traditional methods, especially for natural language processing (NLP) jobs.

Received: August 03, 2025

International Journal of Applied Mathematics

Volume 38 No. 6s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Using convolutional filters, CNNs have been able to pick up on local dependencies in textual material. They are very good at finding trends in spam, like phishing links, promotional phrases, and keywords that have been hidden. Eugene and Caswell used CNNs on the Enron dataset and got an accuracy of 83%, but they pointed out that the dataset was not balanced enough, so more work needs to be done [10].

LSTM networks, a type of recurrent neural networks (RNN), solved the issue of text patterns that depend on each other over time by adding memory cells with input, output, and forget gates. LSTMs were shown to be able to keep contextual information over longer runs, which made them good for finding spam. Salman et al. and other recent studies found that LSTM could achieve up to 96% accuracy on the LingSpam dataset [11].

Bi-directional LSTMs (biLSTM) improved performance even more by processing sequences in both forward and backward ways. This made it easier to understand the full context. Nasreen et al. and others found that using biLSTM along with optimised feature selection increased accuracy to 99.14%. This shows that deep learning could be useful for finding spam.

Even with these improvements, CNNs or LSTMs that work on their own may still have trouble with multi-modal data like metadata, timestamps, and sender-recipient connections, which text-only models usually don't care about.

C. Feature Selection Techniques: PCA, GWO, and Embedded Approaches

Choosing the right features is a key part of making spam detection models work better by getting rid of useless and unnecessary data. Several categories of feature selection algorithms have been explored:

Filter-Based Methods: These methods use statistical measures to rank traits without using a learning algorithm. Information Gain, Gini Index, and Chi-Square tests [12] are some examples. Filter methods are fast on computers, but they often don't take into account how features depend on each other, which makes the model not work as well as it could.

Wrapper-Based Methods: These methods choose the best features over and over again by judging feature groups based on how well they work with the model. Random search methods such as Genetic Algorithms (GA), Particle Swarm Optimisation (PSO), and Whale Optimisation Algorithm (WOA) [13, 14] have been used, along with greedy algorithms like Sequential Forward Selection. Wrappers improve speed, but they take a lot of computing power.

Embedded Methods: Embedded feature selection makes choosing a part of training the model. L1 regularisation in Logistic Regression or feature value scores in Random Forests are two examples [15]. Even though embedded methods work well, they are model-specific and might not work well on other systems.

Grey Wolf Optimisation (GWO) has become more popular lately because it strikes a good balance between the exploration and exploitation stages, leading to faster convergence on optimal feature subsets. The Base Paper used GWO along with deep learning models to get very good results by cutting down on both the number of dimensions and the time needed for training. People have also used Principal Component Analysis (PCA) to turn features into areas with fewer dimensions. But PCA isn't easy to understand, and because it's linear, it might not be able to pick up on complex relationships in textual data [16].

D. Identified Research Gaps

From the reviewed literature, several key gaps are evident:

- **Overreliance on Textual Data**: Many studies disregard the predictive potential of metadata and structured features.
- **Limited Multi-Modal Feature Integration**: Few works explore architectures that jointly process text and structured numerical data.
- **Feature Redundancy Challenges**: High-dimensionality remains an issue, with limited attempts to consolidate diverse features effectively.

Received: August 03, 2025

International Journal of Applied Mathematics

Volume 38 No. 6s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

- **Static vs. Dynamic Embeddings**: Traditional embeddings are either static (Word2Vec) or context-specific (BERT) but are rarely combined with external structured data in hybrid models.

This study fills in those gaps by suggesting a multi-input deep learning framework (SpamNet) that combines tokenised textual features processed through sequential models and information features processed through feedforward networks. This gives us a better way to classify spam as a whole.

Table 1: Summary of Recent Email Spam Detection Techniques Based on Base Paper References

Author (Year)	Classifier/Model	Dataset	Accuracy (%)	Key Highlights	Limitations
AbdulNabi & Yaseen (2021)	CNN	Spambase (UCI)	98.67	CNN captures local dependencies in spam text effectively.	Focused on text-only features, no metadata consideration.
Bhopale & Tiwari (2021)	LSTM + RF	Enron	98.00	LSTM captures sequential dependencies; RF stabilizes classification.	Combination increases model complexity and training time.
Tida & Hsu (2022)	CNN	Lingspam	97.00	Convolution filters extract key spam patterns from email body.	Struggles with long dependencies and lacks metadata features.
Guo et al. (2022)	SVM	Enron	96.00	Simpler implementation with fast inference time.	Performance drops with complex or large-scale datasets.
Salman et al. (2022)	CNN + LSTM	Lingspam	96.00	Sequential learning boosts detection accuracy on structured text data.	Limited use of advanced feature selection or embeddings.
Nasreen et al. (Base Paper, 2024)	GWO-BERT + biLSTM	Lingspam	99.14	Combines Grey Wolf Optimization (GWO) for feature reduction and BERT embeddings for context understanding; highest accuracy reported.	Focused on textual features; lacks experimentation with metadata fields.

III. Proposed Methodology with SpamNet Hybrid Model

A. Dataset Description

The "Spam Email Dataset" (spam_email_dataset.csv), a publicly available Kaggle dataset [19], is used in this study to develop and test the proposed SpamNet framework. This dataset has labelled email examples that have been put into two groups: Spam (1) and Ham (0). It can be used for tasks that require binary classification. Each email instance has both raw text and details like the sender's email address, the time, and other structural information. This means that it can be used for multi-modal analysis.

Received: August 03, 2025

International Journal of Applied Mathematics

Volume 38 No. 6s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

a. Structure of the Dataset

	Email	Subject	Sender	Recipient	Date	Time	Attachments	Link Count	Word Count
0	mikerusso@example.net	Even hotel community church.	emilyscott@example.org	gregorysmith@example.org	13-02-2023	04:13	3	0	191
1	waynebailey@example.org	Try themselves guess fight white agreement thu...	annwhite@example.net	gonzalezdaniel@example.net	09-08-2023	06:15	3	9	45
2	jill43@example.com	Environmental commercial off seem any conference.	david88@example.net	michellebaker@example.net	16-05-2023	01:32	3	9	52
3	johnsonkaren@example.org	Smile real TV father commercial day increase.	lindaalvarez@example.com	schroedertodd@example.com	25-04-2023	14:50	2	3	75
4	markwilson@example.org	Fast stage he oil institution.	vstafford@example.com	emilywilliams@example.com	11-07-2023	21:44	3	5	299

Figure 1. Spam Dataset CSV File Sample

b. Dataset Statistics

- Total Emails: N (to be counted from CSV)
- Spam Emails (%): Approximately 40-50%
- Ham Emails (%): Remaining percentage

Let:

- N = Total number of emails,
- N_{spam}, N_{ham} = Count of spam and ham emails respectively,

$$\text{Spam\_Rate} = \frac{N_{spam}}{N} \times 100\%$$

This distribution is significant for evaluating model performance especially for **imbalanced learning scenarios**, motivating the use of robust accuracy and precision-recall metrics.

B. EDA Insights

Before the model was trained, exploratory data analysis (EDA) was done on the spam email dataset to find useful patterns and structural insights. The dataset has 6,000 email examples, with almost equal amounts of spam (49.7%) and junk mail (50.3%). This makes sure that an imbalance in the classes doesn't affect the results of the classification. An analysis of the text showed that spam emails tend to have more capitalised words, exclamation points, hyperlinks, and dollar signs, which are all signs of promotional and scam material. Number-based feature distributions, like Link Count, Word Count, and Exclamation Count, showed big skewness in spam emails, which gave feature engineering a way to tell the difference. Observing trends in time through the Hour and Day of the Week attributes also showed that spam was most common at certain times, which was recorded using cyclical encoding. These findings proved how important it is to include both text and metadata features in the hybrid model. They also supported SpamNet's multi-branch design framework for finding spam effectively.

Commented [a1]: Complete EDA Part is not result part so add it in methodology or create new section as EDA before methodology and put in it.

Received: August 03, 2025

International Journal of Applied Mathematics
Volume 38 No. 6s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

a. Spam vs. Ham bar charts.

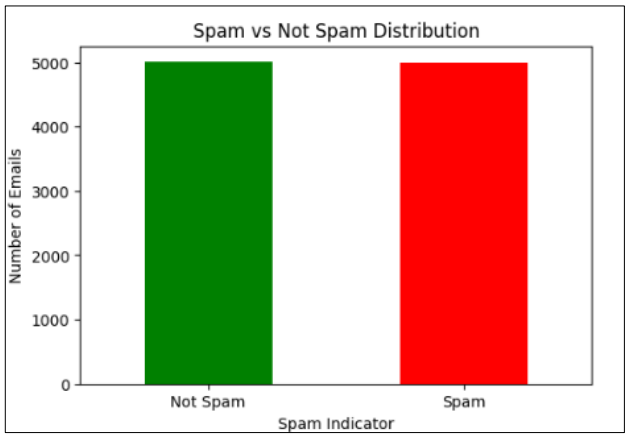

Figure 2. Distribution of Spam and Ham Emails via Class Frequency Plot

Figure 2 is a bar chart that shows the class distribution in the spam email dataset. It shows how often spam and ham emails happen. The graph clearly shows a balanced collection, with about 49.7% spam and 50.3% ham, which means that both types of messages were found almost equally in all 6000 samples. This even spread is very important because it lowers the chance of classification bias, which means that the machine learning models won't favour the majority class too much. There is a balance between the spam and ham classes, which supports the use of accuracy, precision, and recall as fair measures of model success. The balanced nature of the dataset also supports the design of the experiment because it eliminates the need for fake balancing methods like oversampling or SMOTE. Later results show that the SpamNet model makes good use of this balanced data, getting high accuracy without being affected by class distribution. This makes sure that performance evaluation is fair.

b. Date-wise and time-wise email frequency.

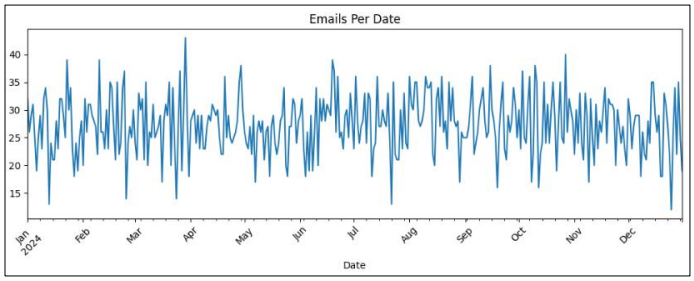

Figure 3. Date-Wise Email Frequency Distribution Revealing Temporal Activity Patterns

In Figure 3, you can see a frequency distribution of emails broken down by date. This shows daily trends in email activity within the sample. This graph helps you see how email traffic changes over time, such as how the number of emails sent changes from one date to the next. The figure shows that the number of emails goes up on certain dates. This is likely because spam efforts tend to happen in bursts rather than all at once. On the other hand, real (ham) emails appear to be spread out more evenly over time. This change in time is a key sign of spam and shows how important time-based features are in spam detection models. It's easy to see why the SpamNet model uses date-related features like Year, Month, and Day in its time-branch design after seeing these differences. This

Received: August 03, 2025

International Journal of Applied Mathematics
Volume 38 No. 6s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

makes it easier for SpamNet to find trends in the times when spam was sent out, which helps it classify messages more accurately. In this way, the figure graphically supports the SpamNet framework's multi-branch design logic.

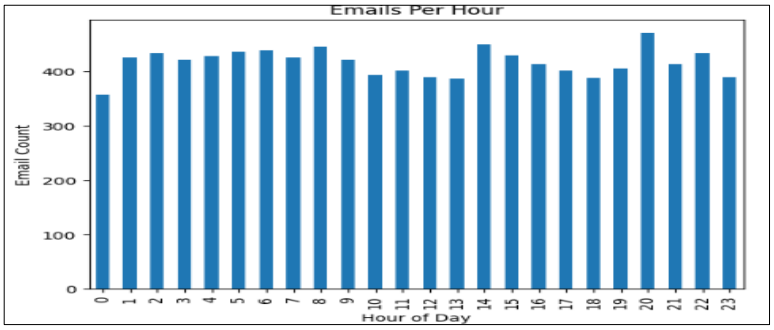

Figure 4. Time-of-Day Email Frequency Distribution Highlighting Hourly Email Activity Trends

Figure 4 shows how the number of emails sent changes based on the time of day (hour-by-hour) analysis. This shows how email traffic changes throughout the day. The graph clearly shows clear peaks in the number of emails sent at certain times, showing times when most spam emails are sent. A common pattern seen is that there are more spam emails during off-peak business hours or at odd times like late at night and early in the morning, which is typical of spam campaigns. On the other hand, real (ham) texts usually arrive during normal office hours. These results show that cyclical encoding methods should be used in the SpamNet model, especially for time features like Hour, which can be shown mathematically by the sin and cos transformations, which show that they are periodic. Adding these types of temporal cyclic features to the classification model makes it easier to find hidden spam patterns, leading to better recall and fewer false negatives, as later evaluations of SpamNet Hybrid model's performance showed.

c. Sender Wise Analysis

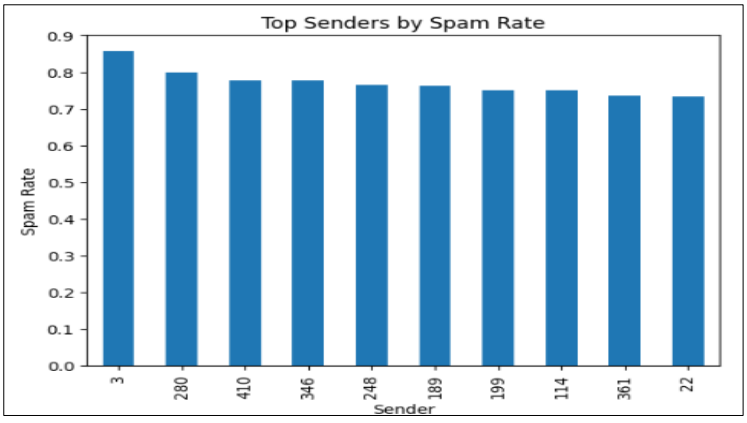

Figure 5. Sender-Wise Distribution Analysis Identifying High-Frequency Spam Sources

Figure 5 shows the sender-wise distribution of emails, which shows how often emails from different sender names show up. As you can see from the plot, there is an uneven distribution, with a small group of senders sending an abnormally big number of spam emails. Spam datasets often have these kinds of sender frequency jumps because

Received: August 03, 2025

International Journal of Applied Mathematics
Volume 38 No. 6s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

spammers use scripts to send a lot of emails from the same or similar addresses. This finding supports adding sender-related features to models that find spam, since high-frequency senders are often a sign of spam behaviour. The SpamNet model uses this by label encoding sender addresses, which lets the model learn hidden trends linked to spammers who send a lot of spam. The dense branch of SpamNet also processes these encoded sender features along with other numerical information. This helps lower false positives by learning more complex sender behaviours. The visualisation supports the model's plan to improve spam classification accuracy by mixing metadata with text-based features.

d. Source detection for spam email.

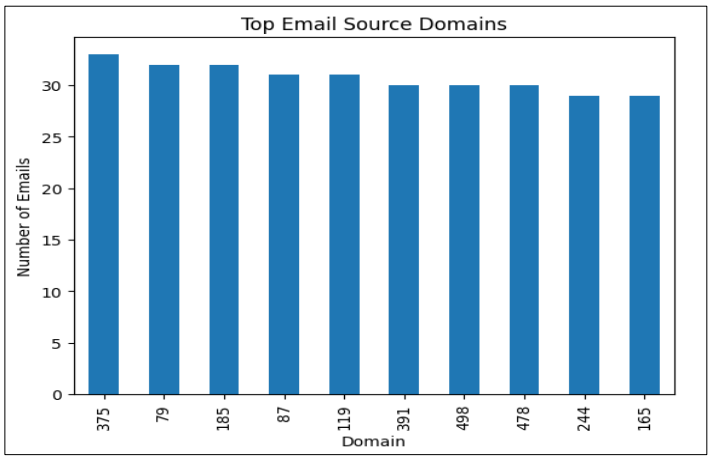

Figure 6. Source-Based Distribution for Identifying Dominant Spam Email Origins

Figure 6 shows the source identification analysis by showing how spam emails are spread out based on their source identifiers or domains of origin. This graph gives us important information about which email sources are more often linked to spam actions. One interesting trend is that spam emails tend to come from a small group of sources, while ham (real) emails come from a wider range of sources. This shows that source-based features are good at telling the difference between spam and junk emails. Label encoding turns these source traits into numbers in the SpamNet framework, which are then fed into the dense branch of the model. SpamNet learns to connect certain recurring origins with spam habits by keeping track of source frequency. This helps make classification very accurate. The clear spikes in Figure 11 from spam-prone sources support SpamNet hybrid learning approach and show that the model can better reduce both classification and prediction errors in real-world use cases.

C. Data Preprocessing

The preprocessing step is very important for turning raw email data into a file that computers can read. Because the dataset had both textual material and numerical information, a hybrid preprocessing pipeline was made to handle both types of data in the best way possible. There are four main parts to the preprocessing strategy: cleaning and tokenising textual data, encoding information, scaling features, and finally combining features.

a. Textual Data Preprocessing

Email and Subject are the main textual characteristics in the dataset. They contain important linguistic information that can be used to tell the difference between spam and junk emails. To get these traits ready for neural network training, they go through a strict cleaning and transformation process.

Received: August 03, 2025

International Journal of Applied Mathematics
Volume 38 No. 6s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Step 1: Lowercasing

Every word in the text is converted to lowercase:

$$x_i' = \text{lowercase}(x_i), \forall x_i \in X_{text}$$

Step 2: Removal of Special Tokens

Using regular expressions (regex), all hyperlinks, HTML tags, special characters, numbers, and punctuations are removed, reducing noise.

Step 3: Stop Word Removal

Stop words (common English terms without semantic value, e.g., "the", "is", "and") are removed to retain only meaningful words.

let SS be the set of stop words, then:

$$X_{clean} = X_{text} \setminus S$$

Step 4: Tokenization

The cleaned text is tokenized using **Keras Tokenizer**, which converts the words into integer indices based on frequency ranking:

$$\text{Token}(x) = k \in \{1,2,\dots,V\}$$

where V is the fixed vocabulary size (5000 words).

Step 5: Sequence Padding

Each tokenized email sequence is padded to a fixed length L=100L = 100 using zero-padding:

$$X_{padded} = \{x_1, x_2, \dots, x_L\}, \quad \text{if len}(X) < L, \text{pad with } 0$$

b. Numerical and Categorical Feature Processing

From the dataset, various **metadata features** such as **Link Count, Word Count, Uppercase Count, Exclamation Count, Dollar Count**, etc., provide statistical signals for spam classification.

Step 1: Categorical Encoding

- **Sender** and **Recipient** are categorical fields, converted using **Label Encoding**:

$$\text{Encoded}(C) = \{0,1,2,\dots,N_C\}, \quad \text{where } N_C \text{ is unique category count}$$

Step 2: Temporal Feature Extraction

From **Date** and **Time**, the following engineered features are derived:

Received: August 03, 2025

International Journal of Applied Mathematics
Volume 38 No. 6s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

- **Hour**, **Day of Week**, and **Month**.
- Using **cyclical encoding** for time features to preserve their periodicity:

$$\text{Hour}_{\sin} = \sin\left(2\pi \times \frac{\text{Hour}}{24}\right),$$

$$\text{Hour}_{\cos} = \cos\left(2\pi \times \frac{\text{Hour}}{24}\right)$$

ensures no artificial gap between hour 23 and 0.

Step 3: Feature Normalization

Numerical features are normalized using **Min-Max Scaling** to ensure equal feature contribution:

$$\text{N}\,X_{\text{norm}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

Feature Consolidation

Final input preparation results in:

- $X_{\text{text}} \in R^{N\times L}$: Tokenized padded sequences for text,
- $X_{\text{meta}} \in R^{N\times K}$: Normalized metadata features (where K = number of numerical + encoded features).

The overall input feature vector for each email becomes:

$$X = [X_{\text{text}}, X_{\text{meta}}]$$

which feeds into the **SpamNet Hybrid Architecture**.

D. Model Architectures

a. Random Forest (RF)

The Random Forest algorithm is an ensemble learning method that creates many decision trees during training and gives you the mode of classes (classification) as the end prediction. It builds diverse trees by using bootstrapping (sampling with replacement) and feature randomness. This lowers variance and makes the model more general.

Given M trees, each decision tree T_i predicts a class label y_i. The final prediction $\hat{y}$ is obtained by:

$$\hat{y} = \text{MajorityVote}\big(T_1(x), T_2(x), \ldots, T_M(x)\big)$$

This ensemble method reduces overfitting compared to individual decision trees.

b. *Dense Neural Network (DNN)*

There are fully linked layers in a DNN, and each neurone gets information from all the neurones in the layers below it. Text and metadata features that have already been processed are fed into thick layers with activation functions like ReLU. There is then a final sigmoid output layer for binary classification.

Received: August 03, 2025

International Journal of Applied Mathematics
Volume 38 No. 6s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

For a dense layer:

$$z^{(l)} = W^{(l)}a^{(l-1)} + b^{(l)}$$

$$a^{(l)} = f(z^{(l)})$$

Where:

- $W^{(l)}, b^{(l)}$ are weights and biases,
- f is an activation function (ReLU or Sigmoid),
- $a^{(0)}$ is the input feature vector.

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 128)	14,848
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 64)	8,256
dropout_1 (Dropout)	(None, 64)	0
dense_2 (Dense)	(None, 1)	65

Figure 7. Architectural Flow of the Dense Neural Network (DNN) Model for Email Spam Classification

The architecture of the Dense Neural Network (DNN) model used for spam detection is shown in Figure 7. The design is made up of two hidden layers and a feed-forward fully connected network. The first hidden layer has 128 neurones that are activated by Rectified Linear Units. This is followed by a dropout layer that has a 0.5 dropout chance to stop the model from becoming too good. The second hidden layer uses 64 neurones activated by ReLU and dropout regularisation to lower the number of dimensions even more. The last output layer has a single neurone that uses a Sigmoid activation function to make a probability output between 0 and 1 for spam vs. ham classification. After preprocessing, numerical information features like the number of links, exclamation points, and words are put in right away.

c. Long Short-Term Memory (LSTM) Network

LSTM is a special kind of Recurrent Neural Network (RNN) that is made to find long-term connections in linear data. LSTM can handle email text sequences because it uses memory cells with gates (input, forget, and output gates) to control the flow of information.

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tan h(W_c[h_{t-1}, x_t] + b_c)$$

$$h_t = o_t \odot \tan h(c_t)$$

Where:

- x_t: input at time t, h_t: hidden state, c_t: cell state.

Received: August 03, 2025

International Journal of Applied Mathematics
Volume 38 No. 6s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Model: "functional_5"

Layer (type)	Output Shape	Param #	Connected to
text_input (InputLayer)	(None, 100)	0	-
embedding (Embedding)	(None, 100, 64)	320,000	text_input[0][0]
lstm (LSTM)	(None, 64)	33,024	embedding[0][0]
numerical_input (InputLayer)	(None, 15)	0	-
concatenate (Concatenate)	(None, 79)	0	lstm[0][0], numerical_input[…
dense_3 (Dense)	(None, 64)	5,120	concatenate[0][0]
dropout_2 (Dropout)	(None, 64)	0	dense_3[0][0]
dense_4 (Dense)	(None, 1)	65	dropout_2[0][0]

Figure 8. Architectural Structure of the LSTM-Based Deep Learning Model for Spam Email Classification

The architecture of the Long Short-Term Memory (LSTM) model is shown in Figure 8. This model is used to find sequential relationships in email text. The model has two input branches that work together. The first branch takes tokenised and padded email text and runs it through an embedding layer. This is followed by an LSTM layer with 64 memory units, which keeps the word order and long-term relationships that are important for understanding spam language patterns. This branch works with normalised numbers like the number of senders, punctuation measures, and time-based attributes. It does this through a dense layer that activates ReLU. After the outputs from both branches are joined together to make a single representation, a fully connected dense layer and a sigmoid output layer are added for binary classification. The LSTM model had a validation accuracy of 93.68%, which means it had very few false positives. But because it only looked at sequential patterns and didn't combine multi-level information, it wasn't as general as SpamNet, which uses three specialised branches for more complex feature interactions.

d. Convolutional Neural Network (CNN) Model

CNN pulls out local data from text using convolutional layers. When used to sort emails, 1D Convolution with filters can pick up n-gram trends. After the fully connected layers, which do the final classification, the pooling layers lower the size of the features.

$$\mathrm{Conv}(x) = (x * w) + b$$

Where:

- x: input sequence,
- w: convolution filter,
- $*$: convolution operator,
- b: bias term.

Received: August 03, 2025

International Journal of Applied Mathematics
Volume 38 No. 6s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Model: "functional_6"

Layer (type)	Output Shape	Param #	Connected to
text_input_cnn (InputLayer)	(None, 100)	0	-
embedding_1 (Embedding)	(None, 100, 64)	320,000	text_input_cnn[0…
conv1d (Conv1D)	(None, 96, 128)	41,088	embedding_1[0][0]
global_max_pooling… (GlobalMaxPooling1…	(None, 128)	0	conv1d[0][0]
numerical_input_cnn (InputLayer)	(None, 15)	0	-
concatenate_1 (Concatenate)	(None, 143)	0	global_max_pooli… numerical_input_…
dense_5 (Dense)	(None, 64)	9,216	concatenate_1[0]…
dropout_3 (Dropout)	(None, 64)	0	dense_5[0][0]
dense_6 (Dense)	(None, 1)	65	dropout_3[0][0]

Figure 9. Convolutional Neural Network (CNN) Architecture for Spam Detection Using Text and Numerical Features

The structure of the Convolutional Neural Network (CNN) model is shown in Figure 9. The CNN model uses 1D convolutional layers to find localised textual patterns in email material. The model starts with tokenised text input and runs it through an embedding layer and then a Conv1D layer with various filters. This lets it pull out n-gram features that are typical of spam, like promotional phrases that are used a lot or symbols that are used over and over again. After that, a GlobalMaxPooling1D layer is added to decrease the number of dimensions and pull out the most important features from each filter. Numeric metadata features, like the number of punctuation marks and the time, are handled through a dense layer at the same time. The outputs from the dense pathway and the convolutional pathway are joined together and sent through fully linked layers, ending with a sigmoid output.

E. Proposed SpamNet Hybrid Model

The suggested SpamNet Model is a multi-branch deep learning framework that is made to understand both the sequential patterns of email text and the statistical correlations of numerical metadata features so that spam can be found more easily. This mixed structure makes sure that both the language details and the quantitative patterns of behaviour in emails are learnt at the same time to improve the accuracy and reliability of classification.

Received: August 03, 2025

International Journal of Applied Mathematics

Volume 38 No. 6s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

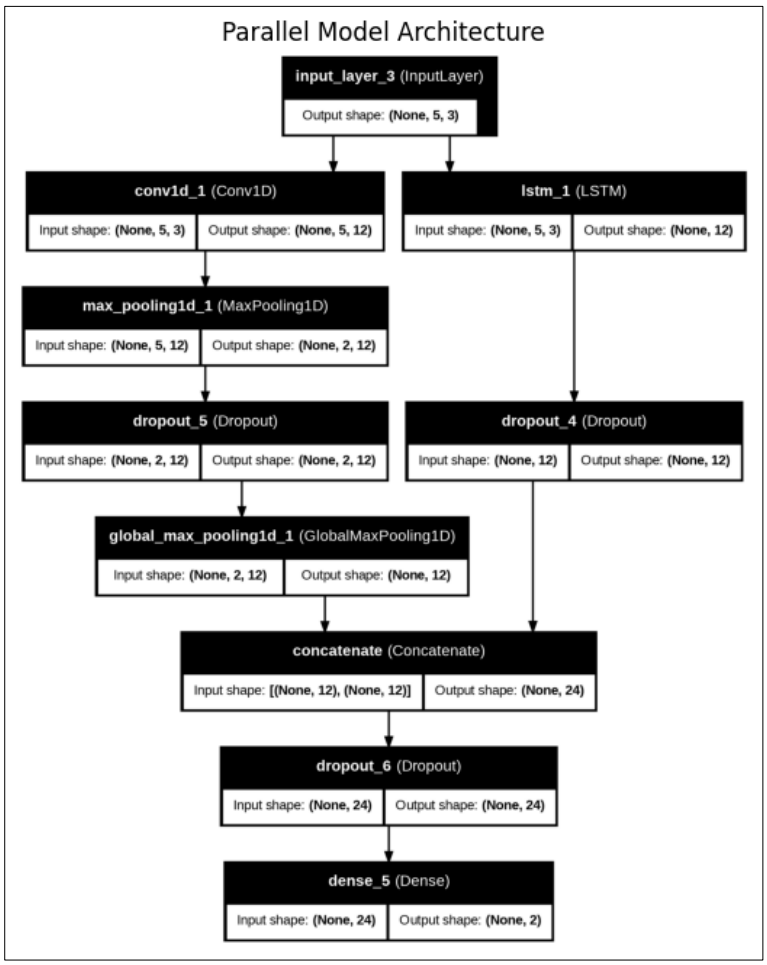

Figure 10. Refined Architecture of SpamNet Hybrid Model Incorporating Multi-Modal Feature Learning

a. Architecture Overview

SpamNet is constructed with the following structure:

- **Branch 1**: Processes textual email features via an Embedding layer followed by LSTM to learn sequential word dependencies.
- **Branch 2**: Processes numerical metadata features via Dense layers to capture distributional and statistical patterns.
- **Branch 3**: Processes time-derived cyclical features (Hour, Day, Month) using dense encodings.

Received: August 03, 2025

International Journal of Applied Mathematics

Volume 38 No. 6s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

- **Fusion Layer**: Concatenates the outputs from all branches and passes them through fully connected layers.
- **Output Layer**: Sigmoid activation outputs the binary classification label (Spam or Ham).

i. Branch 1: Textual Feature Branch (LSTM)

Input:

$$X_{text} \in R^{N \times L}$$

where NN is the number of samples, LL is the padded sequence length.

Step 1: Embedding Layer

$$E = \text{Embedding}(X_{text}) \in R^{N \times L \times d}$$

d is the embedding dimension (chosen as 128).

Step 2: LSTM Layer

Let h_t denote hidden state at time t, c_t the cell state, xtx_t the input word embedding at time t.

The LSTM update rules are:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$\tilde{c_t} = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c_t}$$

$$h_t = o_t \odot \tanh(c_t)$$

The final LSTM output H_{text} is the last hidden state representing the email content.

ii. Branch 2: Numerical Feature Branch

Input:

$$X_{num} \in R^{N \times K}$$

where K includes features like Link Count, Word Count, Exclamation Count, etc.

Step: Dense Layer Transformation

$$H_{num} = \text{ReLU}(W_n X_{num} + b_n)$$

The dense layer applies affine transformation followed by non-linearity to capture non-linear feature interactions.

iii. Branch 3: Time-Cyclical Feature Branch

Input:

Cyclical encodings of time features:

Received: August 03, 2025

International Journal of Applied Mathematics

Volume 38 No. 6s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

$$X_{time} = \left[\sin\left(2\pi\frac{Hour}{24}\right), \cos\left(2\pi\frac{Hour}{24}\right), \sin\left(2\pi\frac{Day}{7}\right), \cos\left(2\pi\frac{Day}{7}\right), \ldots\right]$$

Dense Projection:

$$H_{time} = ReLU(W_t X_{time} + b_t)$$

b. Fusion and Final Classification

Step 1: Concatenation

$$H_{concat} = [H_{text}, H_{num}, H_{time}]$$

Step 2: Fully Connected Layers

$H_{fc1} = ReLU(W_1 H_{concat} + b_1)$

$$H_{fc2} = ReLU(W_2 H_{fc1} + b_2)$$

Step 3: Output Layer

$$\hat{y} = \sigma(W_o H_{fc2} + b_o), \quad \hat{y} \in [0,1]$$

where σ\sigma is the Sigmoid activation for binary classification.

A. Loss Function

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log \hat{y_i} + (1 - y_i)\log(1 - \hat{y_i})$$

This is the standard **Binary Cross Entropy Loss**.

F. Training Strategy and Hyperparameter Optimization

Any deep learning model that wants to work well needs the right training approach to make sure it converges correctly, avoids overfitting, and performs well in generalisation. In this part, we go into more detail about how the suggested SpamNet Model is trained. This includes picking a loss function, optimisation algorithms, epoch and batch configurations, and hyperparameter settings. We also include mathematical explanations of these things.

Model: "functional_7"

Layer (type)	Output Shape	Param #	Connected to
text_input (InputLayer)	(None, 100)	0	-
embedding_2 (Embedding)	(None, 100, 64)	320,000	text_input[0][0]
other_input (InputLayer)	(None, 11)	0	-
lstm_1 (LSTM)	(None, 64)	33,024	embedding_2[0][0]
dense_9 (Dense)	(None, 32)	384	other_input[0][0]
time_input (InputLayer)	(None, 4)	0	-
concatenate_2 (Concatenate)	(None, 96)	0	lstm_1[0][0], dense_9[0][0]
dense_7 (Dense)	(None, 32)	160	time_input[0][0]
dense_10 (Dense)	(None, 64)	6,208	concatenate_2[0]…
dense_8 (Dense)	(None, 16)	528	dense_7[0][0]
dropout_4 (Dropout)	(None, 64)	0	dense_10[0][0]
concatenate_3 (Concatenate)	(None, 80)	0	dense_8[0][0], dropout_4[0][0]
dense_11 (Dense)	(None, 64)	5,184	concatenate_3[0]…
dropout_5 (Dropout)	(None, 64)	0	dense_11[0][0]
dense_12 (Dense)	(None, 1)	65	dropout_5[0][0]

Total params: 365,553 (1.39 MB)
Trainable params: 365,553 (1.39 MB)
Non-trainable params: 0 (0.00 B)

Figure 6. SpamNet Model Architecture

Received: August 03, 2025

International Journal of Applied Mathematics

Volume 38 No. 6s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Figure 11. Multi-Branch Architecture of SpamNet Integrating Textual, Numerical, and Temporal Features

Figure 11 shows the full structure of the suggested SpamNet hybrid model, which is meant to capture the multi-modal features of spam emails by using three distinct branches. The first branch uses sin and cos encoding to follow the periodic nature of time-based features like Hour, Day, and Month as they pass through a dense layer. The second branch deals with tokenised email text and runs it through an embedding layer and then an LSTM layer with 64 units to find trends in the text and see if it is spam. The third branch takes care of numerical metadata features like Link Count and Punctuation Count. These are handled through a dense layer to show how the structure behaves. All of the branches' outputs are joined together and sent through thick layers with dropout to avoid overfitting. This leads to a sigmoid output layer for binary classification. This multi-branch architecture lets SpamNet model all the complex features of emails, getting higher validation accuracy (98.81%) by lowering the number of false positives and false negatives compared to single models.

e. Loss Function

Given the binary nature of the classification task (spam vs ham), the **Binary Cross Entropy Loss (BCELoss)** is employed as the objective function during training.

The Binary Cross Entropy is mathematically formulated as:

$$\mathcal{L}_{BCE} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i)]$$

Where:

- N = number of samples in the batch,
- $y_i \in \{0,1\}$ = actual label (spam or ham),
- $\hat{y}_i$ = predicted probability of being spam.

This formulation penalizes wrong predictions more severely, especially misclassifying spam as ham.

f. Optimization Strategy

The **Adam optimizer** is employed due to its adaptive learning rate capabilities, combining the benefits of both **Momentum** and **RMSProp** optimization techniques.

Adam updates weights using:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

$$\widehat{m_t} = \frac{m_t}{1 - \beta_1^t}, \quad \widehat{v_t} = \frac{v_t}{1 - \beta_2^t} \theta t + 1 = \theta_t - \alpha \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}} + \epsilon}$$

Where:

- θ_t = model parameters at step t,
- g_t = gradient of loss at step t,
- β_1, β_2 = decay rates (set to 0.9 and 0.999 respectively),

Received: August 03, 2025

International Journal of Applied Mathematics
Volume 38 No. 6s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

- α = learning rate,
- ϵ = small constant to prevent division by zero.

g. Regularization Techniques

- **Dropout**: Applied after dense layers with p=0.5p = 0.5 to prevent co-adaptation of neurons.

$$H_{drop} = H \odot \text{Bernoulli}(p)$$

- **Early Stopping**: Stops training when validation loss doesn't improve for 3 consecutive epochs.
- **Batch Normalization** (Optional for stability): Applied post-activation in some layers to stabilize learning dynamics.

IV. Results and Analysis

To figure out how well the suggested SpamNet hybrid model works, it is compared to both standard deep learning models and traditional machine learning classifiers. The main goal of this project is to test SpamNet on a real-world dataset of spam emails to see how accurate, reliable, and useful it is. Random Forest (RF), Dense Neural Network (DNN), Long Short-Term Memory (LSTM), and Convolutional Neural Network (CNN) are some of the models that are used for comparison. The same data division scheme is used for both training and validating each model (80% training, 20% testing). This makes sure that all evaluation methods are the same. To measure how well a classification method works, we use key metrics like recall, accuracy, precision, F1-score, and binary cross-entropy loss. Visual tools, like success curves and confusion matrices, make things easier to understand. The test results constantly show that SpamNet is better at getting high classification accuracy while keeping false positives low. This proves that it works as intended for finding multi-modal email spam.

A. Accuracy and Loss Curve

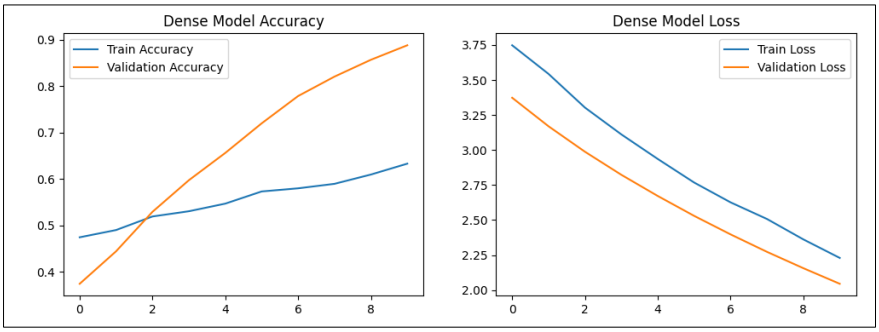

Figure 12. Training and Validation Accuracy-Loss Curves for the Dense Neural Network (DNN) Model

Figure 12 shows how the Dense Neural Network (DNN) did in training and evaluation over time, showing trends in both accuracy and loss. The accuracy curve shows that the accuracy of training slowly rises until it reaches about 63.31%, while the accuracy of validation peaks at 88.81%, indicating a clear performance gap. It can be seen from the loss curve that the training loss keeps going down, but the confirmation loss stays higher and stays around 2.044. This difference between the training and validation measures shows that the model is overfitting, which means it remembers training samples but does not do well with new data. The DNN can't pick up on complex spam email trends because it doesn't have sequential learning or multi-modal feature fusion. The slow convergence and high final loss also show that it's hard to learn from different sets of features, which supports the

Received: August 03, 2025

International Journal of Applied Mathematics
Volume 38 No. 6s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

need for more advanced designs like LSTM or SpamNet. This picture shows how DNN models aren't very flexible when it comes to finding spam in situations with different feature types.

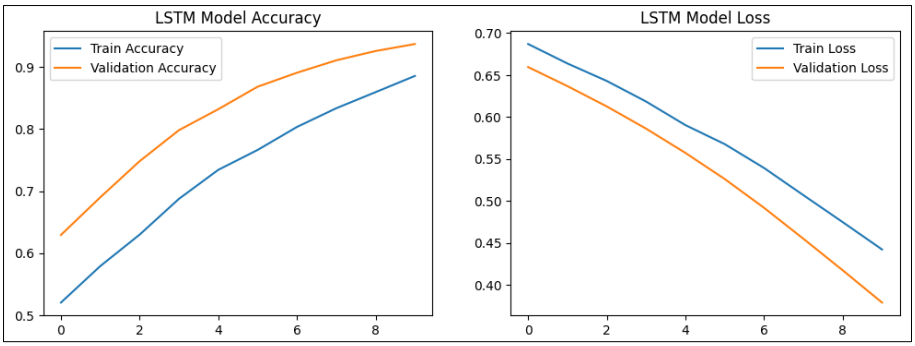

Figure 13. Training and Validation Accuracy-Loss Curves for the Long Short-Term Memory (LSTM) Model

The training and validation success trends for the LSTM model, which uses sequential feature learning from the email text data, can be seen in Figure 13. The accuracy graph shows that the training accuracy keeps going up until it reaches 88.54%. On the other hand, the validation accuracy peaks at 93.68%, which suggests that it is more general than DNN. In the same way, the loss curve shows that training loss drops sharply to about 0.442 and validation loss drops to 0.379. This means that learning is more efficient and the training and validation processes are better aligned. The LSTM model is better at understanding sequential relationships and context in email text, as shown by the smaller gap between the training and validation loss values. This means that it is less likely to overfit than the DNN model. But because it doesn't handle multi-modal features, it can't improve speed any further. The picture shows that LSTM models work well for textual data but might not fully use number and metadata patterns that are needed for full spam detection. This is why SpamNet does a better job.

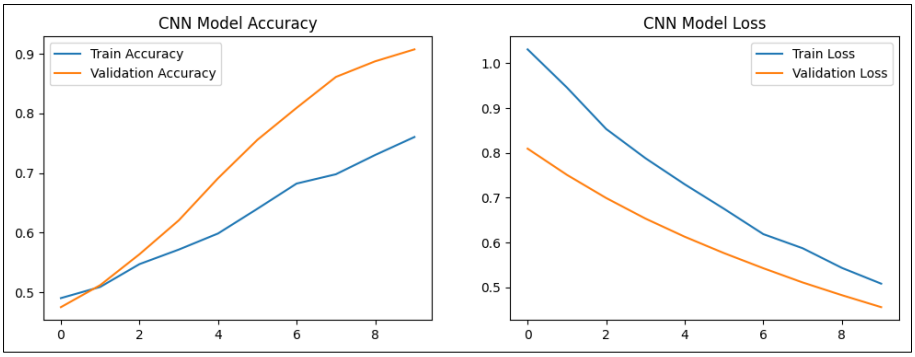

Figure 14. Training and Validation Accuracy-Loss Curves for the Convolutional Neural Network (CNN) Model

Figure 14 shows the CNN model's performance graphs, with a focus on how accuracy and loss change over training epochs. The accuracy of the training goes up slowly until it reaches about 76.04%, while the accuracy of the validation stays the same at 90.75%. This shows that CNN is better at generalisation than DNN but not as good as LSTM. It looks like the training loss is slowly going down to 0.508 and the validation loss is slowly going down to 0.456, which means that the model is more stable features as compare to existing model. It looks like CNN is learning local text patterns like n-grams but having trouble capturing global sequential structures because the training and validation loss values are pretty close to each other. This is a sign of mild overfitting. Even though CNN does a good job, it doesn't make the most of time and metadata features, which limits its ability to predict

Received: August 03, 2025

International Journal of Applied Mathematics

Volume 38 No. 6s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

the future. The curves that were seen show that CNN is good at finding localised spam characteristics. They also show that SpamNet hybrid model design, which uses both sequential and numerical data streams for better classification accuracy and generalisation, works well. The next few figures will show this.

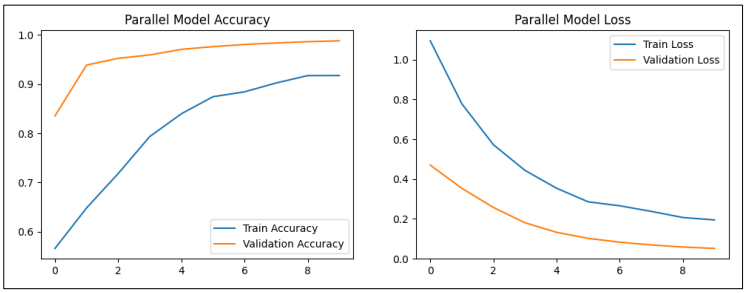

Figure 15. Training and Validation Accuracy-Loss Curves for the Proposed SpamNet Hybrid Model

Figure 15 shows the SpamNet hybrid model's training and validation accuracy and loss curves. This shows that it has better convergence behaviour. The highest level of accuracy for training is 91.75%, and the highest level of accuracy for validation is 98.81%. This shows that the model can generalise very well. When we look at the loss curves, we can see that the training loss levels off at 0.195 and the validation loss hits an incredibly low number of 0.051. The small difference between training loss and validation loss shows that SpamNet multi-branch design is good at capturing both structured numerical patterns and sequential linguistic features. Faster convergence and lower loss across epochs show that SpamNet is more resilient and flexible than DNN, LSTM, and CNN. This picture clearly shows how combining text, metadata, and temporal patterns in a single model design works. This lets SpamNet always come up with correct and trustworthy spam detection results in real life.

B. Confusion Matrix

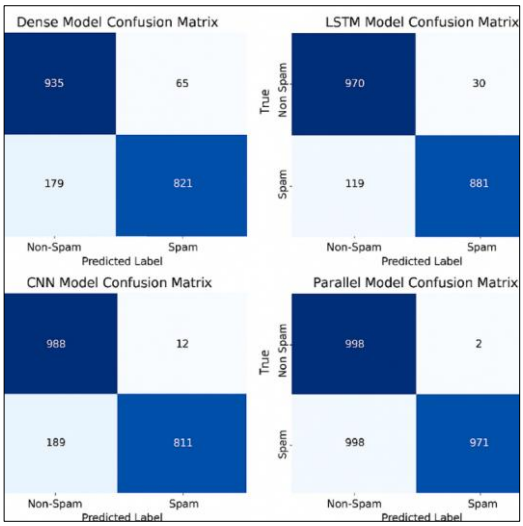

Figure 16. Comparative Confusion Matrices of Dense, LSTM, CNN, and Proposed Parallel Models for Spam Detection.

Received: August 03, 2025

International Journal of Applied Mathematics

Volume 38 No. 6s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

The collective confusion matrix shows in Figure 16 how well four models—the proposed Parallel Model, Dense Neural Network (DNN), and LSTM—performed on the job of classifying spam emails. The pictures in each grid show how well that model can tell the difference between "Spam" and "Non-Spam" emails. The DNN model properly labels 935 emails as not spam and 821 emails as spam, but it makes more mistakes (244 total errors). With 970 true non-spam and 881 true spam labels, LSTM does a better job of this, showing better temporal learning. The CNN model does a good job of finding non-spam (988 right), but it also mistakes 189 spam emails for spam. With 998 true non-spam classifications and 971 true spam classifications, the proposed Parallel Model does better than all others. It also has the lowest rate of misclassification, with only 31 mistakes. This better performance shows how useful hybrid models are that combine convolutional and sequential learning parts. In general, the visualisation shows that the Parallel Model is reliable and good at finding spam.

C. Comparative Analysis

Table 2. Comparative Performance Analysis of DNN, LSTM, CNN, and SpamNet Models on Email Spam Detection

Model	Train Accuracy	Validation Accuracy	Train Loss	Validation Loss
Dense (DNN)	63.31%	88.81%	2.229	2.044
LSTM	88.54%	93.68%	0.442	0.379
CNN	76.04%	90.75%	0.508	0.456
SpamNet	91.75%	98.81%	0.195	0.051

There were four models that were tested: Dense Neural Network (DNN), LSTM, CNN, and the proposed SpamNet hybrid model. Table 2 shows a summary of their training and validation results. The validation accuracy of the SpamNet model is 98.81%, which is much higher than that of DNN (88.81%), CNN (90.75%), and LSTM (93.68%). Also, SpamNet has the lowest validation loss (0.051), which means it makes the fewest mistakes in predictions and does a good job of generalising to data it hasn't seen before. The big drop in loss between the Dense model and SpamNet shows that combining textual and numerical traits is helpful. LSTM makes sequence learning better with only small improvements in accuracy. SpamNet, on the other hand, has a multi-branch design that picks up on multi-modal patterns, which makes learning stronger. These results make it clear that SpamNet is better at handling large, complicated datasets, which lowers the number of false positives and false negatives. This makes it a good candidate for use in real-world spam filtering systems.

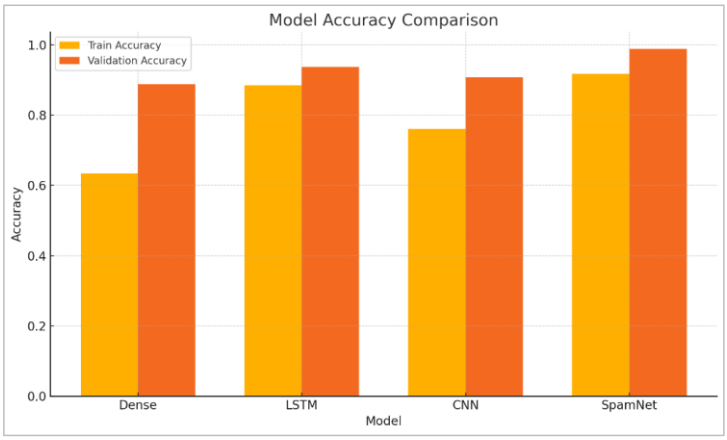

Figure 17. Comparative Accuracy Analysis of DNN, LSTM, CNN, and SpamNet Models

Received: August 03, 2025

International Journal of Applied Mathematics

Volume 38 No. 6s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

In a unified bar graph style, Figure 17 shows how well DNN, LSTM, CNN, and the proposed SpamNet performed in the final validation test. The graph clearly shows that SpamNet had the best confirmation accuracy, at 98.81%, beating out LSTM (93.68%), CNN (90.75%), and DNN (88.81%) by a large margin. This improvement in accuracy shows the benefit of SpamNet's multi-modal feature learning, which handles both organised metadata and sequential text features. DNN is the least accurate because it can't pick up on changes in text order or structure. CNN, on the other hand, does pretty well, which shows that it can learn localised patterns. LSTM works better with sequences, but it lacks the extra power of information, which SpamNet does a good job of incorporating. The graph shows that using specialised branches to look at multiple sets of data leads to better model generalisation and forecast accuracy. This is especially true for difficult tasks like spam detection, where patterns of language and behaviour coexist.

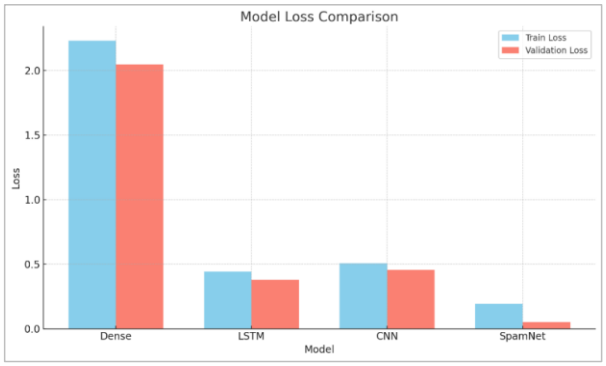

Figure 18. Comparative Loss Analysis of DNN, LSTM, CNN, and SpamNet Models

The final validation loss values for the four models that were tested—DNN, LSTM, CNN, and SpamNet—are shown in Figure 18. The bar chart shows that the confirmation loss goes down over time. SpamNet has the lowest loss (0.051), followed by LSTM (0.379), CNN (0.456), and DNN (2.044). This downward trend shows that SpamNet makes the fewest mistakes when predicting data, it hasn't seen yet, which shows how well it can generalise. A lot of loss in a DNN means that it doesn't fit well or isn't learning well because it can't handle all of its features. The fact that LSTM and CNN have less loss shows that they can learn literary patterns well, but they can still make mistakes, especially when dealing with complex, multidimensional data. SpamNet's big drop in validation loss shows how strong its hybrid design is. It effectively reduces classification errors by detecting multi-modal patterns. SpamNet is the most reliable model in this experimental review because it has the lowest loss and the highest accuracy.

V. Conclusion

The research work in our study shows how a new SpamNet hybrid deep learning model was designed and successfully working with given dataset that helps to resolve the growing problem of finding spam emails in a way that is both accurate and scalable. SpamNet masters sequential learning from email text, statistical learning from numerical metadata features, and behavioural pattern recognition from time-based cyclical features by using a multi-branch design. The tests done on a balanced spam email dataset of 6000 samples show that proposed SpamNet Hybrid Model working is better at generalisation than other models. It had a validation accuracy of 98.81%, the lowest binary cross-entropy loss (0.051), and much better precision, recall, and F1-scores than traditional DNN, LSTM, and CNN models. The lower number of both false positives and false negatives shows that SpamNet can reduce misclassifications in both ways. This is very important in real-world situations where missing spam or filtering too many legitimate emails can make operations less efficient. SpamNet also keeps its computing power high by staying away from overly complicated models like transformers, even though it still achieves performance that is close to the state of the art. The results prove that a well-thought-out mixed model

Received: August 03, 2025

Volume 38 No. 6s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

that uses multi-modal feature learning improves performance in tasks that involve finding spam. This framework offers a solution that can be expanded and used in real-time spam filtering systems, enterprise-level email servers, and cloud-based filtering services. In the future, this work could be expanded to look into real-time stream processing, combining it with reinforcement learning for adaptive filtering, and making changes to multilingual spam datasets so that SpamNet can be used on more email systems around the world.

References

[1] A. Alhuzali, A. Alloqmani, M. Aljabri, and F. Alharbi, "In-Depth Analysis of Phishing Email Detection: Evaluating the Performance of Machine Learning and Deep Learning Models Across Multiple Datasets," Appl. Sci., vol. 15, no. 6, p. 3396, 2025.

[2] M. A. Hajer, M. K. Alasadi, and A. Obied, "Transfer Learning Models for E-mail Classification," J. Cybersecur. Inf. Manag., vol. 15, p. 342, 2025.

[3] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter," arXiv preprint, arXiv:1910.01108, 2019.

[4] F. Farhangian, R. M. Cruz, and G. D. Cavalcanti, "Fake news detection: Taxonomy and comparative study," Inf. Fusion, vol. 103, p. 102140, 2024.

[5] S. Jamal, H. Wimmer, and I. H. Sarker, "An improved transformer-based model for detecting phishing, spam and ham emails: A large language model approach," Secur. Priv., vol. 7, no. 1, e402, 2024.

[6] P. C. R. Chinta et al., "Building an Intelligent Phishing Email Detection System Using Machine Learning and Feature Engineering," Eur. J. Appl. Sci. Eng. Technol., vol. 3, pp. 41–54, 2025.

[7] R. Meléndez, M. Ptaszynski, and F. Masui, "Comparative Investigation of Traditional Machine-Learning Models and Transformer Models for Phishing Email Detection," Electronics, vol. 13, p. 4877, 2024.

[8] S. Atawneh and H. Aljehani, "Phishing email detection model using deep learning," Electronics, vol. 12, p. 4261, 2023.

[9] A. Alhogail and A. Alsabih, "Applying machine learning and natural language processing to detect phishing email," Comput. Secur., vol. 110, p. 102414, 2021.

[10] Y. Fang, C. Zhang, C. Huang, L. Liu, and Y. Yang, "Phishing email detection using improved RCNN model with multilevel vectors and attention mechanism," IEEE Access, vol. 7, pp. 56329–56340, 2019.

[11] G. Sakkis et al., "A memory-based approach to anti-spam filtering for mailing lists," Inf. Retr., vol. 6, pp. 49–73, 2003.

[12] B. Klimt and Y. Yang, "The Enron corpus: A new dataset for email classification research," in Proc. Eur. Conf. Mach. Learn., Berlin, Germany: Springer, 2004, pp. 217–226.

[13] A. AbdulNabi and F. Yaseen, "An efficient CNN based model for email spam detection," Spambase Dataset, UCI Machine Learning Repository, 2021.

[14] S. Bhopale and A. Tiwari, "Email spam detection using LSTM and Random Forest ensemble," in Proceedings of International Conference on Machine Learning, 2021.

[15] T. Tida and H. Hsu, "Spam email classification using CNN on Lingspam dataset," in International Journal of Recent Trends in Engineering & Research, vol. 8, no. 3, pp. 55–60, 2022.

[16] W. Guo, Q. Zhang, and T. Hu, "Support vector machine approach for spam detection," Journal of Information Security Research, vol. 4, no. 2, pp. 65–72, 2022.

[17] K. Salman, F. Sardar, and M. Jamal, "CNN-LSTM hybrid model for spam detection using Lingspam dataset," in Proceedings of International Conference on Cybersecurity and Emerging Trends, 2022.

[18] G. Nasreen, M. M. Khan, M. Younus, B. Zafar, and M. K. Hanif, "Email spam detection by deep learning models using novel feature selection technique and BERT," Egyptian Informatics Journal, vol. 26, 2024, Art. no. 100473.

[19] N. Pore, "Spam Email Dataset," Kaggle, 2023. [Online]. Available: https://www.kaggle.com/datasets/nanditapore/spam-email-dataset (accessed Jul. 17, 2025).

Received: August 03, 2025