

# A Probabilistic Machine Learning Approach for Identifying Text Spams

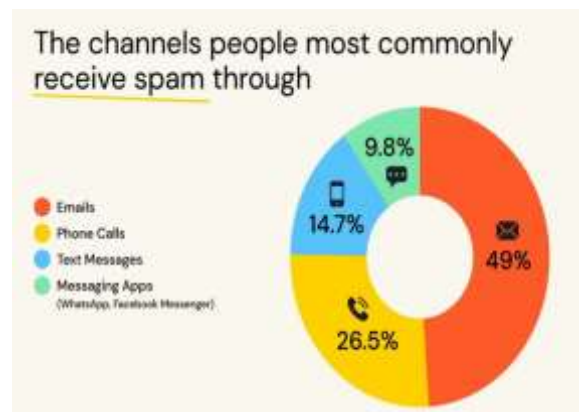
Ramlal Agrawal<sup>1</sup> Prof. Pankaj Raghuwanshi<sup>2</sup>

**Abstract**— With increased internet usage, one of the most prevalent problems faced is constant spamming. While web applications and mailing services are heavily spammed, the upsurge of handheld mobile devices has led to an outburst of heavy mobile spamming. The matter is more severe in mobile devices due to lesser sophisticated filtering mechanisms in built in mobile operating systems. Spam detection is challenging due to the need for semantic analysis of the mobile spam messages, which generally tend to have overlapping polarities. In this work, a mobile spam classification technique is developed based on Adaptive Neuro Fuzzy Inference System (ANFIS) comprising of Gini's index Fuzzy and Back-propagation in machine learning. The approach uses the Gini's splitting criteria for the data sets and backpropagation based neural network as the machine learning classifier. The evaluation of the proposed system is based on the accuracy of classification and number of iterations. The results obtained in the proposed work are compared with existing techniques and it is shown that the proposed technique outperforms them in terms of accuracy of classification.

**Keywords**— *Mobile Spam Classification, ANFIS, Gini's Index, Back Propagation, Training Iterations, Classification Accuracy*

## I. INTRODUCTION

Mobile spamming has become one of the most common techniques for promotions, customer churning and potential attacks targeting the frequently used handheld mobile devices which are more prone to such attacks [1]. The ease of collecting mobile contacts, connected data bases and relatively lesser sophisticated filtering mechanisms for the mobile spam filtering makes its extremely challenging to thwart spamming attacks. The most common sources of receiving spams are depicted in figure 1.



**Fig.1 Common Spamming Channels**  
(Source: <https://www.emailtooltester.com/en/blog/spam-statistics/>)

Some of the spamming attacks may be benign while others may be malignant trying to redirect mobile users to malicious websites where user security may be compromised [2]. Since the amount of data is staggering large and complex, off late machine learning based approaches are becoming common to filter out spams. One of the challenges which machine learning based approaches face for mobile spamming platforms is the limited computational and processing capabilities of handheld mobile devices. This makes it necessary to design and test algorithms which are compatible with various versions of mobile operating systems and also supported by limited memory and processing hardware as there exists a lot of diversity in the mobile hardware of different devices. This paper is organized as [3]:

Section I introduces the basic concepts pertaining to mobile spam classification and its necessities. Section II briefly summarizes the work done in the domain. Section III discusses the proposed approach. Section IV illustrates the obtained results. The findings of the paper are concluded in the conclusion.

## II. MACHINE LEARNING MODELS FOR SPAM CLASSIFICATION

This section highlights the popular machine learning models for identifying text spams:

Text spam includes unsolicited promotional messages, phishing attempts, malicious links, fraudulent financial offers, and impersonation attacks. Because these

messages are short, diverse, and often crafted to mimic legitimate text, detecting them is a challenging task. Machine learning (ML) has emerged as a powerful solution by analyzing linguistic patterns, statistical features, and contextual cues to automatically classify messages as spam or legitimate (ham). As text messaging becomes central to daily transactions and authentication systems, robust ML-based spam detection has become essential for digital security [4].

Classical machine-learning models such as Naïve Bayes, Support Vector Machines (SVM), and Logistic Regression have served as foundational approaches for spam identification. Naïve Bayes is particularly effective because of its ability to model word occurrences using probabilistic assumptions. It performs well on short messages where certain keywords strongly correlate with spam behavior [5]. SVM, on the other hand, excels at separating high-dimensional text features using hyperplanes, making it suitable for TF-IDF and n-gram-based representations. Logistic Regression provides an interpretable baseline model that allows analysts to understand why certain messages are classified as spam. These classical models are computationally efficient and perform reliably when combined with appropriate preprocessing, such as tokenization, stop-word removal, and stemming [6].

As text spam techniques became more sophisticated, ensemble learning models gained popularity due to their superior predictive performance. Algorithms such as Random Forest, Gradient Boosting Machines (GBM), XGBoost, and LightGBM integrate decisions from multiple weak learners to generate a more accurate final prediction. They automatically capture feature interactions, nonlinear patterns, and message characteristics like unusual punctuation, message length, special characters, and URL presence. Ensemble models are especially effective when large labeled datasets are available and when spam messages contain subtle variations. Their robustness to noise and ability to handle imbalanced datasets make them ideal for detecting rare but harmful spam variants [7].

The introduction of deep learning significantly improved spam detection by enabling systems to learn semantic meaning and contextual dependencies. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks are capable of modeling sequential patterns in text, such as suspicious phrasing, repetitive patterns, or time-dependent spam bursts. These architectures understand the flow of language rather than relying solely on keyword frequency. Convolutional

Neural Networks (CNNs) have also been applied to text classification by extracting high-level features from word embeddings, capturing local n-gram patterns related to spam messages. Both CNN and LSTM networks outperform classical models when dealing with noisy text, slang, or multilingual spam [8].

More recently, attention mechanisms and Transformer-based architectures, such as BERT, RoBERTa, and DistilBERT, have revolutionized spam detection through their ability to learn deep contextual relationships within messages. These models understand the meaning of words within context rather than in isolation, allowing them to detect sophisticated spam messages that avoid traditional keywords. Transformers handle sarcasm, complex sentence structures, code-mixed languages, and disguised phishing attempts far better than earlier approaches. Fine-tuning pre-trained language models on spam datasets has shown state-of-the-art performance, significantly reducing false positives in real-world systems such as messaging apps and email filters [9].

Real-world spam detection systems must handle billions of messages daily, making scalability, real-time processing, and adaptability critical. ML pipelines typically incorporate feature extraction layers, model inference engines, continuous model retraining, and updating feedback loops [10]. Online learning algorithms such as stochastic gradient descent (SGD) or FTRL (Follow-The-Regularized-Leader) allow models to update their parameters incrementally as new spam samples are detected. This ensures that spam filters remain effective against evolving adversarial techniques, including obfuscated text, random character insertion, and image-based spam [11].

### III. PROPOSED SYSTEM MODEL

The proposed system model is presented next:

#### A. *Data processing and normalization:*

Since neural nets directly process numeric data sets, the processing of data is done prior to training a neural network [12]-[13]. The texts are first split into training and testing data samples in the ratio of 70:30 for training and testing. Further, a data vector containing known and commonly repeated spam and ham words is prepared. The SMS spam collection v.1 dataset is used as a dataset for the proposed work. Text normalization is followed by removal of special characters and punctuation marks [15]. Subsequently the data set structuring and preparation is performed based on the feature selection. The features selected are:

- 1) Spam words
- 2) Ham Words
- 3) URLs in the message
- 4) Lengthy numerical strings which can be contact numbers
- 5) Character length
- 6) Special symbols
- 7) Presence of currency values
- 8) Self-answering texts [16]

The feature vectors along with the list of commonly accepted spam and ham lists of words comprises of the training vector. A similar process is done for both the training and testing datasets.

### B. Adaptive Neuro Fuzzy Inference Systems (ANFIS)

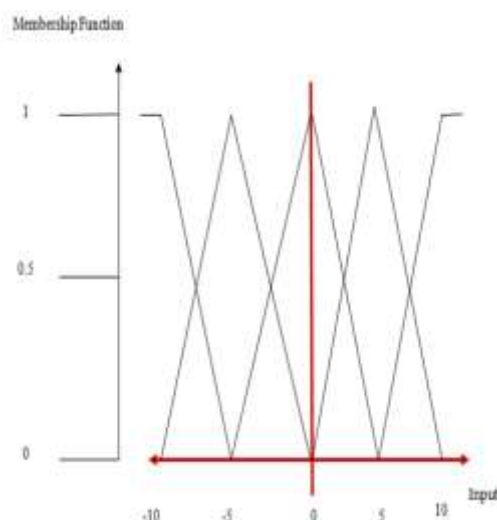
A very important tool that proves to be effective in several classification problems is fuzzy logic. It is often termed as expert view systems. It is useful for systems where there is no clear boundary among multiple variable groups. The relationship among the inputs and outputs are often expressed as membership functions expressed as [17]:

A membership function for a fuzzy set A on the universe of discourse (Input) X is defined as:

$$\mu_A: X \rightarrow [0, 1] \quad (1)$$

Here,

each element of X is mapped to a value between 0 and 1. It quantifies the degree of membership of the element in X to the fuzzy set A [18].



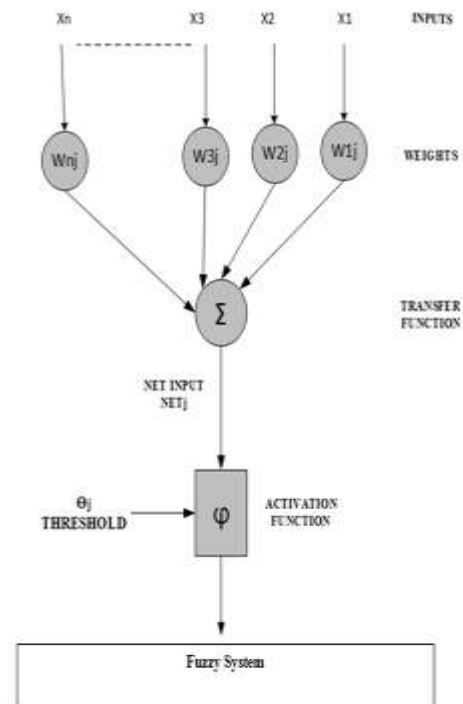
**Fig.2 Graphical Representation of Membership Functions**

Here,

x axis represents the universe of discourse (Input). y axis represents the degrees of membership in the [0, 1] interval.

The final category is neuro fuzzy expert systems which governs the defining range of the membership functions.

The ANFIS can be thought of as a combination of neural networks and fuzzy logic. In this mechanism, the neural network module decides the membership functions of the fuzzy module. The ANFIS structure is depicted in figure 3.



**Fig.3 Block Diagram of Neuro-Fuzzy Expert Systems**

The splitting can be done through the Gini's index which is especially useful for overlapping data sets since it can split data sets with overlapping classes based on conditional probability. The Gini's index for splitting is defined as [19]:

$$GI = 1 - \sum_{i=1}^n p_i^2 \quad (2)$$

Here,

GI represents the Gini's Index

P is the probability of a class

The prepared data vector for training is used for training wherein the weights are initialized randomly. A stepwise implementation is done as:

1. Prepare two arrays, one is input and hidden unit and the second is output unit.

Here, a two dimensional array  $W_{ij}$  is used as the weight updating vector and output is a one dimensional array  $Y_i$ .

3. Original weights are random values put inside the arrays after that the output [20].

$$x_j = \sum_{i=0} y_i W_{ij} \quad (3)$$

Where,

$y_i$  is the activity level of the  $j^{\text{th}}$  unit in the previous layer and

$W_{ij}$  is the weight of the connection between the  $i^{\text{th}}$  and the  $j^{\text{th}}$  unit.

4. Next, activation is invoked by the sigmoid function applied to the total weighted input.

$$y_i = \left[ \frac{e^x - e^{-x}}{e^x + e^{-x}} \right] \quad (4)$$

Summing all the output units have been determined, the network calculates the error (E).

$$E = \frac{1}{2} \sum_i (y_i - d_i)^2 \quad (5)$$

Where,  $y_i$  is the event level of the  $j^{\text{th}}$  unit in the top layer and  $d_i$  is the preferred output of the  $j_i$  unit.

### C. Implementing Back Prop:

Calculation of error for the back propagation algorithm is as follows:

Error Derivative ( $EA_j$ ) is the modification among the real and desired target:

$$EA_j = \frac{\partial E}{\partial y_j} = y_j - d_j \quad (6)$$

Here,

E represents the error

y represents the Target vector

d represents the predicted output

Error Variations is total input received by an output changed given by:

$$EI_j = \frac{\partial E}{\partial x_j} = \frac{\partial E}{\partial y_j} X \frac{dy_j}{dx_j} = EA_j y_j (1 - y_i) \quad (7)$$

Here,

E is the error vector

X is the input vector for training the neural network

In Error Fluctuations calculation connection into output unit is computed as [21]:

$$EW_{ij} = \frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial x_j} = \frac{\partial x_j}{\partial w_{ij}} = EI_j y_i \quad (8)$$

Here,

W represents the weights

I represents the Identity matrix

I and j represent the two dimensional weight vector indices

Overall Influence of the error:

$$EA_i = \frac{\partial E}{\partial y_i} = \sum_j \frac{\partial E}{\partial x_j} X \frac{\partial x_j}{\partial y_i} = \sum_j EI_j W_{ij} \quad (9)$$

The partial derivative of the Error with respect to the weight represents the error swing for the system while training. The gradient is computed as:

$$g = \frac{\partial e}{\partial w} \quad (10)$$

Here,

g represents the gradient

e represents the error of each iteration

w represents the weights.

The gradient is considered as the objective function to be reduced in each iteration. A probabilistic classification using the Bayes theorem of conditional probability is given by:

$$P\left(\frac{H}{X}\right) = \frac{P\left(\frac{X}{H}\right)P(H)}{P(X)} \quad (11)$$

Here,

Posterior Probability [ $P(H/X)$ ] is the probability of occurrence of event H when X has already occurred  
Prior Probability [ $P(H)$ ] is the individual probability of event H

X is termed as the tuple and H is is termed as the hypothesis.

Here, [ $P(H/X)$ ] denotes the probability of occurrence of event X when H has already occurred.

The final classification accuracy is computed as:

$$Ac = \frac{TP+TN}{TP+TN+FP+FN} \quad (12)$$

Here,

TP represents true positive

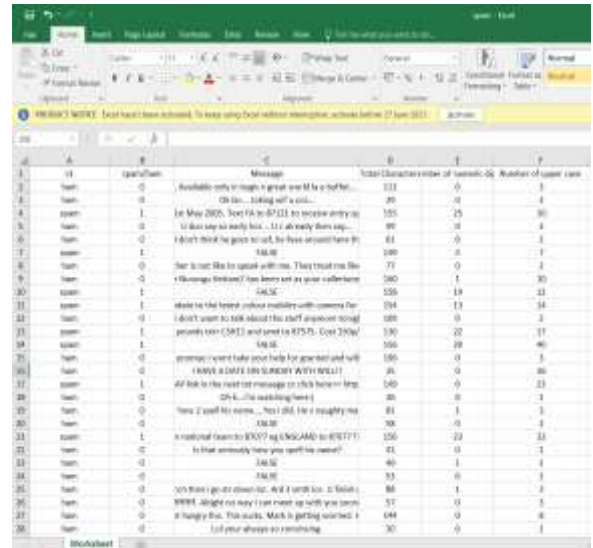
TN represents true negative

FP represents false positive

FN represents false negative

## IV. RESULTS

The system is implemented on Matlab. The results obtained on implementing the proposed system is discussed in this section.

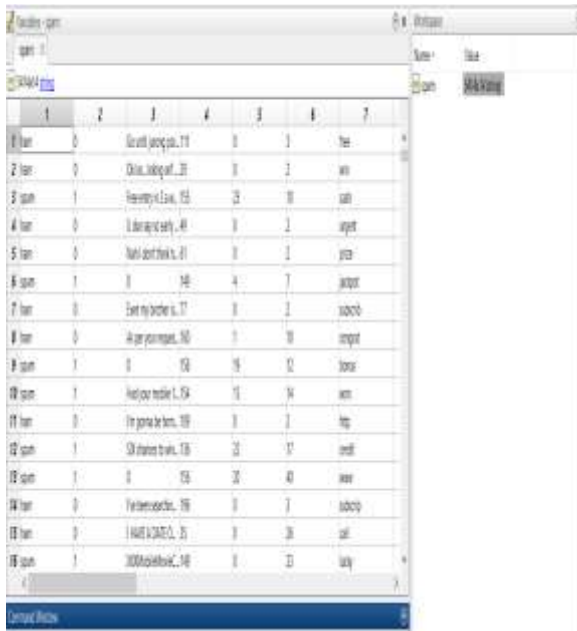


	Message	Character count	Number of words
1	Available only in English. Great work!	333	10
2	Oh yes... taking well a cold...	20	10
3	Let May 2025. Next FA to 0711 to receive entry to	105	25
4	to don say so early too... it's already there say...	49	10
5	I don't think he goes to school, for those arguments (P)	63	10
6	ALLO	549	10
7	She is not like to speak with me. They treat me like	77	10
8	a (thousands) friends? You know not as your colleague	280	10
9	ALLO	559	10
10	ALLO	559	10
11	I don't want to talk about this stuff anymore tonight	485	10
12	ALLO	559	10
13	ALLO	559	10
14	ALLO	559	10
15	ALLO	559	10
16	ALLO	559	10
17	ALLO	559	10
18	ALLO	559	10
19	ALLO	559	10
20	ALLO	559	10
21	ALLO	559	10
22	ALLO	559	10
23	ALLO	559	10
24	ALLO	559	10
25	ALLO	559	10
26	ALLO	559	10
27	ALLO	559	10
28	ALLO	559	10
29	ALLO	559	10
30	ALLO	559	10

Fig.4 Raw data samples

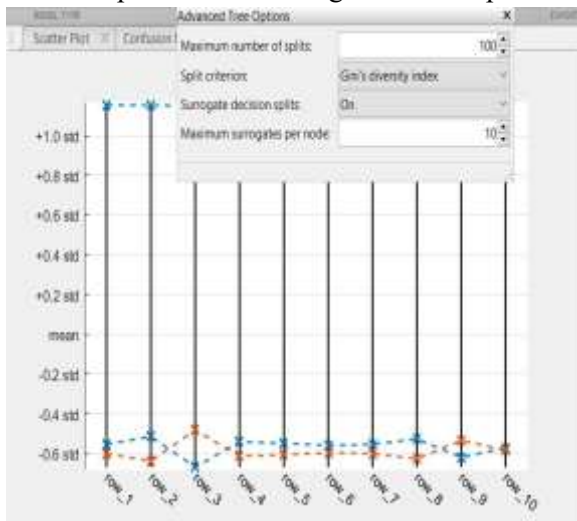
The raw data samples are collected after which it is imported to the Matlab workspace.





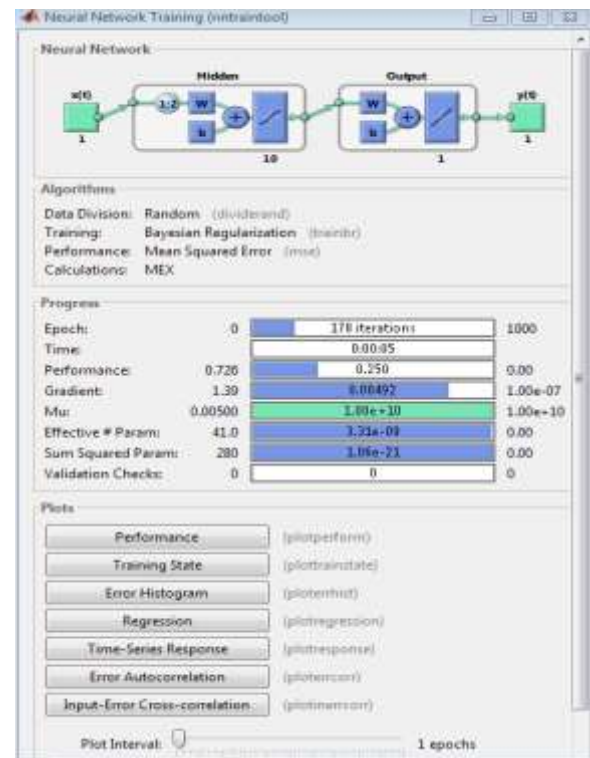
**Fig.5 Conversion of Data into string**

Subsequently, the data is converted into strings for ease of analysis of textural data. The data is split into training and testing data samples in the ratio of 70:30. While other data division ratios could have been used, but in this work, the standard 70:30 ratio is adhered to. The next process is invoking the Gini's split criterion.



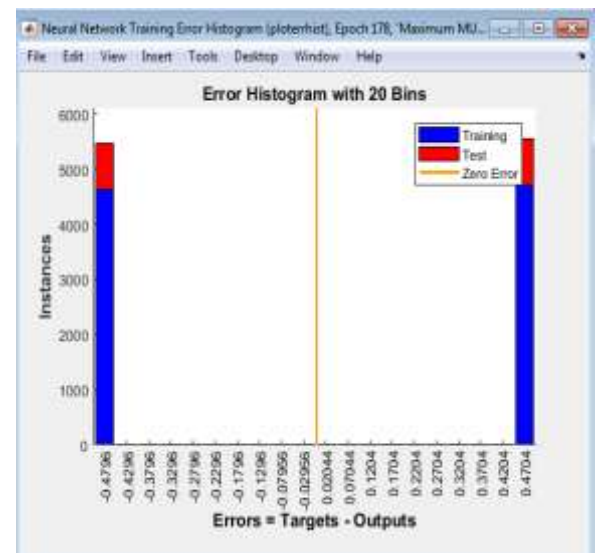
**Fig.6 Invoking the Gini's Split criteria**

The Gini's split criteria is the precursor to the training of the system.



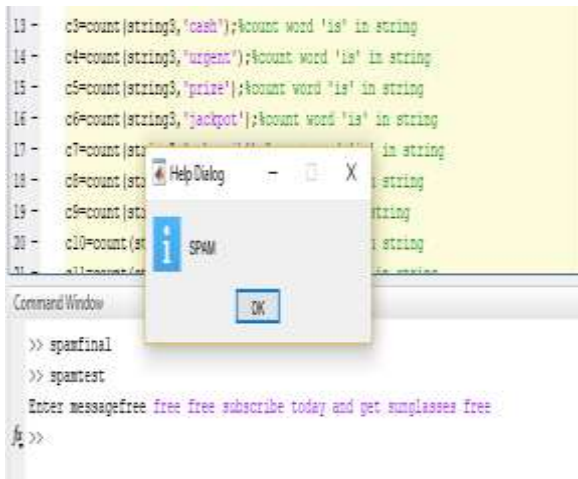
**Fig.7 Training Performance**

Figure 7 depicts the training parameters of the proposed system which consumes 5 seconds to run 378 iterations of the back propagation algorithm. A 20 neuron hidden layer is designed for the system.



**Fig.8 Training Error Histogram**

The training error histogram is depicted in figure 7 which is an indicator of the errors occurring during the training process.



**Fig.9 GUI for detection**

Figure 8 depicts the GUI for spam detection. A similar GUI represents that of the non-spam or ham case. A comparative analysis with existing approaches is tabulated in table I.

**Table I: Comparative Accuracy Analysis of Proposed and Existing Algorithms**

S.No.	Technique	Accuracy (%)
1.	SNAP	83.9
2.	AIR SENTI	80.5
3.	Naïve Baye's	64
4.	Random Forests	63
5.	ANN with BackProp	95.81
6.	<b>Proposed Approach: Gradient Descent with BackProp and Gini-Index</b>	<b>99.75</b>

It can be observed from the tabulated results, that the proposed work outperforms the existing algorithms such as SNAPM AIR SENTI, Naïve Bayes', Random Forests and ANN with Back Prop.

**CONCLUSION:** It can be concluded from the aforesaid arguments that mobile spam classification is extremely challenging and non-trivial due to the constraints of computational power and memory at our disposal. Moreover, easier access to handheld devices makes systems more prone to spamming attacks. Text spam classification is non trivial in the sense that it generally belongs to non-clear or fuzzy boundary datasets. The proposed approach presents a mobile spam classification mechanism Using Gini's Index and ANFIS. It has been shown that the proposed approach outperforms the existing techniques in terms of classification accuracy. Additionally, the technique consumes moderate

number of iterations and low execution time which are critical considerations for mobile devices. It can be observed that the proposed approach outperforms existing approaches in terms of accuracy.

## References

1. Liu, Xiaoxu, Haoye Lu, and Amiya Nayak. "A Spam Transformer Model for SMS Spam Detection." IEEE Access, vol. 9, 2021, pp. 80253–80263
2. Oswald, Christopher, S. E. Simon, and Anupam Bhattacharya. "SpotSpam: Intention Analysis–Driven SMS Spam Detection Using BERT Embeddings." ACM Transactions on the Web, vol. 16, no. 3, 2022, pp. 1–27
3. Altunay, Hakan C., and Zafer Albayrak. "SMS Spam Detection System Based on Deep Learning Architectures for Turkish and English Messages." Applied Sciences, vol. 14, no. 24, 2024, article 11804
4. Ghourabi, A., A. M. Mahmood, and M. Q. Alzubi. "Enhancing Spam Message Classification and Detection with Pre-trained Transformers and Ensemble Learning." Future Internet (special issues 2023–2024)
5. Shaaban, Mai A., Yasser F. Hassan, and Shawkat K. Guirguis. "Deep Convolutional Forest: A Dynamic Deep Ensemble Approach for Spam Detection in Text." arXiv, 2021. <https://arxiv.org/abs/2110.15718>
6. N. Sharma, "A Methodological Study of SMS Spam Classification Using Machine Learning Algorithms," 2022 2nd International Conference on Intelligent Technologies (CONIT), Hubli, India, 2022, pp. 1-5
7. P. Manasa et al., "Tweet Spam Detection Using Machine Learning and Swarm Optimization Techniques," in IEEE Transactions on Computational Social Systems, vol. 11, no. 4, pp. 4870-4877, Aug. 2024
8. X. Liu, H. Lu and A. Nayak, "A Spam Transformer Model for SMS Spam Detection," in IEEE Access, vol. 9, pp. 80253-80263, 2021
9. M. Ketcham, T. Ganokratana, P. Pramkeaw and N. Chumuang, "Spam Text Detection using Machine Learning Model," 2023 International Technical Conference on Circuits/Systems, Computers, and Communications (ITC-CSCC), Jeju, Korea, Republic of, 2023, pp. 1-6
10. Pudasaini, S. "SMS Spam Detection Using Relevance Vector Machine." Procedia Computer Science (or similar proceedings), 2023
11. Del Rosario, B. D. P. Fernandez and D. A. Padilla, "Email Spam Classification using DistilBERT," 2023 IEEE 15th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM), Coron, Palawan, Philippines, 2023, pp. 1-6
11. M. Salman, M. Ikram and M. A. Kaafar, "Investigating Evasive Techniques in SMS Spam Filtering: A Comparative Analysis of Machine Learning Models," in IEEE Access, 2024, vol. 12, pp. 24306-24324.

12. R. Agarwal et al., "A Novel Approach for Spam Detection Using Natural Language Processing With AMALS Models," in IEEE Access, 2024 vol. 12, pp. 124298-124313.
13. H. A. Al-Kabbi, M. -R. Feizi-Derakhshi and S. Pashazadeh, "Multi-Type Feature Extraction and Early Fusion Framework for SMS Spam Detection," in IEEE Access, 2023 vol. 11, pp. 123756-123765
14. P. Joseph and S. Y. Yerima, "A comparative study of word embedding techniques for SMS spam detection," 2022 14th International Conference on Computational Intelligence and Communication Networks (CICN), 2022, pp. 149-155
15. AK Jain, D Goel, S Agarwal, Y Singh, G.Bajaj, "Predicting Spam Messages Using Back Propagation Neural Network", Journal of Wireless Personal Communications, Springer 2021, vol. 110, pp. 403-422.
16. KS Adewole, NB Anuar, A Kamsin, "SMSAD: a framework for spam message and spam account detection", Journal of Multimedia Tools and Applications, Springer 2021, vol. 78, pp. 78, 3925–3960.
17. Aliaksandr Barushka, Petr Hajek, "Spam filtering using integrated distribution-based balancing approach and regularized deep neural networks", Springer 2018
18. Surendra Sedhai, Aixin Sun, "Semi-Supervised Spam Detection in Twitter Stream", IEEE 2018
19. Chao Chen, Yu Wang, Jun Zhang, Yang Xiang, Wanlei Zhou, Geyong Min, "Statistical Features-Based Real-Time Detection of Drifted Twitter Spam", IEEE 2017S
20. N. Mirza, B. Patil, T. Mirza and R. Auti, "Evaluating efficiency of classifier for email spam detector using hybrid feature selection approaches," 2017 International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2017, pp. 735-740
21. B. C. Sulochana, B. S. Pragada, K. Lokesh and M. Venugopalan, "PySpark-Powered ML Models for Accurate Spam Detection in Messages," 2023 2nd International Conference on Futuristic Technologies (INCOFT), Belagavi, Karnataka, India, 2023, pp. 1-6.