SMS Spam Detection Using NLP and Render Deployment

P.M. Benson Mansingh[1], Gayathri Gutha[2], Siriteja Yakkali[3], Vasu Vamsi Krishna Sakala[4] and Karthik Raja Dharam[5]
{benyuva@gmail.com[1], guthagayathri48@gmail.com[2], siriyakkali@gmail.com[3], vasuvamsikrishnasakala@gmail.com[4], karthik.cs121@gmail.com[5]}

Department of ACSE, Vignan University, Guntur, Andhra Pradesh, India[1, 2, 3, 4, 5]

Abstract. This study outlines the machine learning-driven approach utilizing Natural Language Processing (NLP) to automate classifying SMS messages as spam or legitimate, building upon prior research in SMS spam filtering. The escalating volume of unsolicited SMS messages necessitates efficient solutions to mitigate user annoyance and potential security threats, such as phishing and fraud. The proposed system effectively transforms raw text data into a high-dimensional feature space suitable for machine learning models by employing techniques such as tokenization, stop-word removal, stemming, and TF-IDF feature extraction. We evaluated several classification algorithms, including naive Bayes, Random Forest, and Support Vector Machines, demonstrating that Bernoulli naive Bayes achieved a commendable performance, with a precision of 94.54% and an accuracy of 96.42%. This system addresses the limitations of traditional rule-based filters by adapting to the dynamic nature of spamming techniques, thereby enhancing user experience and mitigating security risks associated with phishing and fraudulent messages. Furthermore, we deployed the trained model via Render, providing a user-friendly web interface for real-time spam classification. The system's robustness is validated through comprehensive exploratory data analysis and rigorous model evaluation, demonstrating its potential for practical application in enhancing SMS communication security and efficiency.

Key words: Spam detection, Machine Learning, NLP, SMS security, Render Deployment.

1 Introduction

In today's digital landscape, the rapid growth of mobile communication has led to an increased reliance on Short Message Service (SMS) for personal, business, and organizational interactions. But with this explosive growth, spamming has also increased tremendously, presenting both a threat to, and an unpleasant experience for, users and instant messaging services [1][2]. Such unsolicited messages present an obstacle to normal communication and pose serious security challenges in the form of phishing attacks, financial scams, and malware Malicious Software distribution. With the continuous emergence of new spams case 311 evolution methodology spamming algorithm traditional rule-based methods face many difficulties to enable reliable detection and blockage of spam because of that it is necessary to use more intelligent and flexible alternatives. SMS spam attacks exploit weaknesses in SMS systems to impersonate trusted entities, making them difficult to detect using ordinary filtering techniques [3][4][8]. Conventional spam filtering techniques adopt predefined rules and keyword- based filters that fail to prevent spam bots from using innovative methods of obfuscation. As such, a higher false positive and false negative rate undermine the performance of such methods, which requires more reliable solution. Machine learning and NLP application provides a promising solution to solve this problem by identifying and classifying unwanted

ICITSM-Part I 2025, April 28-29, Tiruchengode, India

DOI 10.4108/eai.28-4-2025.2357939

messages in an automated form with high relevance. Spam detection models, generally machine learning based, use NLP methods for text analysis, feature extraction and to classify texts into spam or not spam (messages which in other words are often referred to as ham). The effectiveness of the approach relies on good pre-processing steps like tokenization, stop word removal, stemming and on a good feature extraction based on TF-IDF (Term Frequency-Inverse Document Frequency). These processes convert a textual buoy of data into a numeric part, enabling classification methods to identify patterns and differentiate genuine and malicious entities. There are three numeric classifiers in interest: Bernoulli naive Bayes, Multinomial Naïve Bayes, and Gaussian Naive Bayes for the detection of SMS spam. The models are evaluated using accuracy, precision, and recall. Of the models tested, Bernoulli Naive Bayes obtained the best performance, with 96.42% accuracy and 94.54% precision [9] [10]. For accessibility and ease of use, we deploy the model using Render, to classify SMS in real-time using a user-friendly web app. The security relevance of SMS spam detection is significant, as it adds an extra layer of security against phishing scams, scam campaigns, and unsolicited advertisements. It is hoped that by embedding an intelligent SPAM detection system into these products, communication between users can be made more efficient, user security can be ensured and more robust security systems can be developed in mobile chat apps. Old-fashioned spam detection was to extract text features and use the models like Naïve Bayes, Support Vector Machines, Decision Tree. Performance was further increase through the application of pre-processing techniques such as, tokenization, stop-word removal, stemming. Feature engineering using n-grams, TF-IDF, and statistical text representations improved accuracy; ensemble and hybrid models decreased false positives. These are the foundations of spam detection, but they have difficulty with changing spam techniques, obfuscation, and multilanguage content, which is why more advanced deep learning and NLP models are necessary.

2 Literature

Because of the increasing usage of SMS as an important medium of mobile communication, the amount of unwanted advertisements, phishing, and SMS fraud has been soaring. At first, SMS spam was only comprised of advertising messages from businesses, but eventually cybercriminals adopted this medium to run their scam campaigns [3]. And these spam messages generally assume the form of authentic emails sent by banks, government agencies or service providers with a goal of duping victims into sharing confidential information. While email spamming is reduced by advanced filtering technologies, the problem of SMS spam has not been well addressed because most of the messaging apps do not have default protection mechanisms [6]. Spammers further change their tactics over time, for instance by using short URLs and obfuscated text, in their attempt to elude keyword-based filters and detection approaches [7]. Not only is SMS spam annoying, it also represents a variety of security challenges such as phishing (smishing), financial fraud, and malware vectors. Phishing content often attempts to trick people into clicking a malicious URL that can result in identity theft, unauthorized banking transactions, etc. Fraudulent SMS campaigns, for instance a-round fake lottery gains or investments, are used to coerce victims into payment and sharing personal information. Moreover, there are spam mails with malware links which has the power to make us the owner of the spy program on the mobile or give the device to the hostage with a ransomware, thus compromising our privacy and data. Besides security threat, spam also causes user annoyance and network overhead, which perhaps will affect communication efficiency [5]. With the proliferation of these challenges, traditional rules-based systems would be rendered inefficient and inefficient to handle spam detection, which makes the need for more

sophisticated methods like machine learning and NLP approaches inevitable for the purpose of ensuring good credible spam detection. [4][8].

2.1 Traditional Spam Detection Techniques

The traditional methods of spam detection are primarily rule based filters which use some predefined rules such as keyword, black list and pattern analysis. Filtering through keyword is looking for words and phrases that are major portions of spam messages, the word and phrase like" win,"" free,"" congratulations," and" urgent" included in most of spam messages. Blocking based on blacklists also prevents messages from senders in known blacklists or from the blacklist of the 14 suspicious phone numbers. Some systems will also employ heuristic based techniques, which examine patterns of frequency and structure of the message to determine whether it is likely spam or not. Although these methods can offer a complimentary form of defence, some of these methods are passive, and rely on a manually constructed rule-sets. Since spammers also attempt to bypass rule-based filters (by employing misspelled words, special characters or context-sensitive language), the accuracy tends to decrease over time, as these products become out-of-date. A very important limitation of the previous kind of spam detection based on traditional methods is being high at both false positives and false negatives. False positives occur when valid messages are incorrectly marked as spam and are not delivered, causing important messages to not be received. Conversely, false negatives represent those spam mails that are not detected, which may compromise users' security. Furthermore, the use of rule-based models has little flexibility, they have to be frequently updated to counter new types of spamming. As these methods are based on rigid rules, they do not capture new spam trends and are not effective for spam identification in changing environments. In order to address such drawbacks, state-of-the-art spam detection systems have been based on the use of machine learning and Natural Language Processing (NLP) techniques, by which models may learn from patterns in data and enhance the precision and flexibility of spam classification. B. Research Gap and Motivation for the Study

- Drawbacks of Conventional Spam Detection: Rules based alters cannot effectively deal with newly emerging spamming tricks, leading to greater false positives and false negatives. Furthermore, numerous machine learning methods have difficulties with adjusting to dynamic patterns and performing well for various data sets.
- Growing Security Threats: SMS spam is being used more and more for phishing, financial fraud, and malware, putting user privacy and financial security in danger.
- Need for an Adaptive Solution: The existing ones need to be adapted constantly in order to stay useful. This work uses models of NLP and machine learning to create more exact, scalable, and real-time spam detection.
- Practical Significance and Impact: As both the model and code deployment using Render serve to provide end users a friendly, high performance, and easily accessible tool, we confirmed the utility of the solution to be straightforward for practical purposes regarding SMS filtering and prolonged security.

3 Methodology

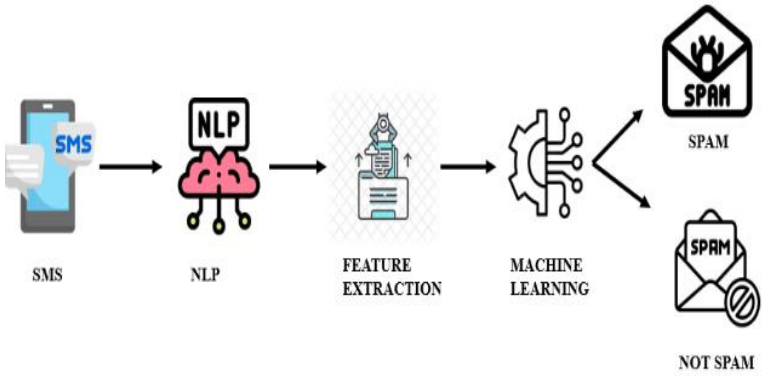

Fig.1. Overview of Methodology.

This study considers the entire life cycle of an SMS-spam detection system built on machine learning and Natural Language Processing (NLP) processing to enhance message classification accuracy. Fig.1 shows the Overview of Methodology. The work is systematic and initiates with the collection of various publicly available SMS datasets to create a diverse and balanced training and evaluation set for the experiment. Text pre-processing techniques such as tokenization, stop- word removal, stemming, and lemmatization are used to clean and improve the text data for better representation of features. Feature engineering is critical to transform raw text into useful numerical features, including TF-IDF (Term Frequency-Inverse Document Frequency) and n-gram analysis to look for context in messages. We systematically compare the effectiveness of different machine learning classifiers, logistic regression, decision tree, random forest, and support vector machine (SVM), for spam detection model. Accuracy, precision, recall, F1-score are some of the key performance indicators to determine best model. To enhance the performance of the classifier, hyper parameter tuning is carried out. After choosing the best model, we deploy it to a cloud-based platform called Render for real-time predictions and to guarantee scalability in the face of these volumes of SMS data. The system can be deployed with a web-based interface or API that users can interface with to enter messages and get a rapid prediction of their class. Fig.2 shows the System Architecture. The method potentially increases SMS security by stifling spam, while adding to continued measures to clamp down on fraudulent and malicious messaging.

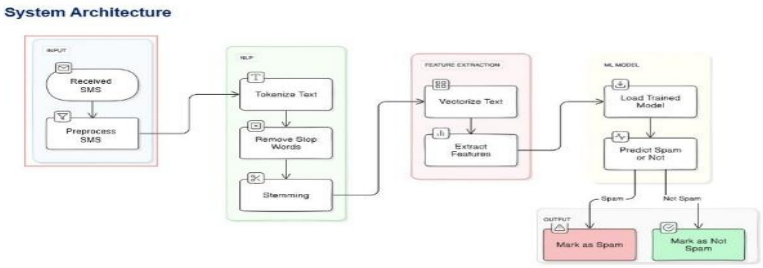

Fig.2. System Architecture.

3.1 SMS Spam Detection: Processing, Classification, and Deployment

Data Collection and Pre-processing There are multiple sources that can be used for raw data collection: databases, APIs, and user generated content. The collected data, as demonstrated in Fig (3), is typically not well-structured and may contain missing values, duplicated records or inconsistent entries. After the data is acquired (explained in the next section) and before the data can be analysed, it goes through pre-processing that includes tasks such as missing value treatment, numerical data normalization, encoding of categorical variables and removal of useless information, to obtain a clean dataset suitable to analysis. Well pre-processed data increases the model's accuracy and efficiency

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN
...	...	...	...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...	NaN	NaN	NaN
5568	ham	Will İ_ b going to esplanade fr home?	NaN	NaN	NaN
5569	ham	Pity, * was in mood for that. So...any other s...	NaN	NaN	NaN
5570	ham	The guy did some bitching but I acted like i'd...	NaN	NaN	NaN
5571	ham	Rofl. Its true to its name	NaN	NaN	NaN

5572 rows × 5 columns

Fig.3. Dataset.

Feature Engineering After cleaning the data, one should choose or generate summary features that improve the data. Feature engineering consists of finding relevant features well known for contributing to performance. It could be a matter of processing existing features, mixing-and-matching variables, creating new feature and encoding categorical. The idea is that we want the model to have the best inputs it can, so that it can better find patterns and make predictions.

Model Selection This phase is used to choose a machine learning model that best fits the problem. The decision can be made depending on the volume of data, the complexity of the task, and the analysis goal. The algorithms that have been tested are the following: decision trees, random forests, support vector machine and deep learning models. Performance measures such as accuracy, precision, recall and computational performance are evaluated in order to guarantee that the chosen model follows the requirements of the problem. Bernoulli Naïve Bayes (BNB) – Best performing model Purpose: It works well in Text Classification, when the features are binary (word presence/absence). Pros: Highly accurate, effective in detecting spam. Performance: Accuracy 96.42%, precision 94.54%. Multinomial Naïve Bayes (MNB)– Variation Model Objective: This model classifies words in which the classifier is relevant to the word occurrences and the model is useful for the tasks of NLP. Strength: High precision (100%) with more false negatives (false spam messages). Results: Accuracy = 95.74%, precision = 100%. Gaussian Naïve Bayes (GNB) - Least Appropriate Use: Works on normal distribution, less effective in text classification. Weakness: The accuracy is lower (47.89%) to cause the counter misclassification. Outcome: Accuracy 85.98, not so great for spam detection. Fig.4.

shows the Model Performance. Best Choice: Bernoulli NB because of highest accuracy and best balance.

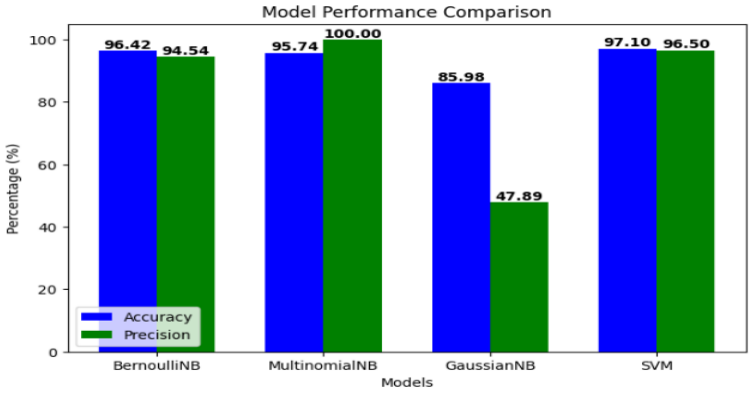

Fig.4. Model Performance.

Algorithm Used Feature Selection It is the first step to decide the appropriate algorithm on which model to be trained and develop using the pre-processed dataset. This is done by feeding the model with input information letting it find patterns and connections. Techniques like regularisation, cross validation and hyper parameter tuning are used to improve accuracy. 2.1 Regularisation researchers like including a sparse penalty on the weights in the model. Techniques such as feature selection and class balancing are used to reduce errors and improve generalization on new data. Furthermore, methods such as early stopping and learning rate adaptation are applied to avoid overfitting, as well as to maintain the efficiency of the model. On completion of the training, the trained model is completely evaluated using validation and test dataset to check its performance before deployment. The fig.5 and fig.6 shows Ham and Spam.

Fig.5. Ham.

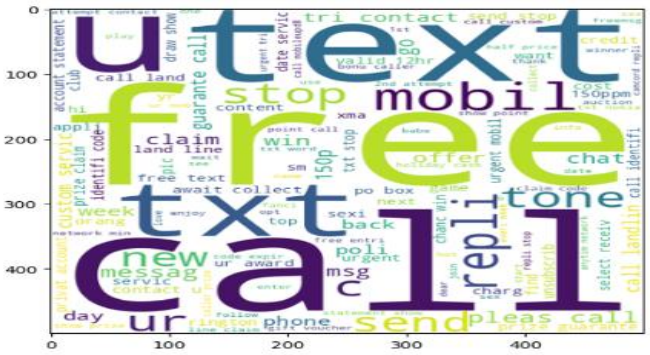

Fig.6. Spam.

Evaluation and Deployment Following training, the model is tested on a separate data to evaluate the performance. Main evaluation measures including accuracy, F1-score, precision, and recall are used to measure its performance. If the model performs well, it can be put into production and accept fresh data to make predictions against. In order to remain efficient and adjust to the changes of the pattern, such systems may require constant watch and some more update once in a while, as suggested in Fig (7).

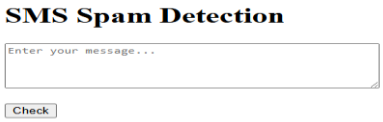

Fig.7. SgiveMS spam detection.

4 Algorithm

- Convert all text data to lowercase to ensure uniformity across different messages. Remove unnecessary punctuation, special characters, and numbers that do not contribute to spam detection.
- Break each SMS into individual words for detailed analysis. Use stemming or lemmatization to convert words to their root form, ensuring different variations of the same word are recognized as a single entity.
- Remove common stop words such as" the,"" is," and" and" to eliminate noise and improve model performance.
- Use the pre-process text(text) function to apply all these transformations efficiently. Transform the processed text into numerical representations to make it understandable for machine learning models.

• Employ either Count Vectorizer () for implementing a Bagof-Words model or Tfidf Vectorizer () for utilizing the Term Frequency-Inverse Document Frequency (TF-IDF) technique. • Fit and transform the dataset using either features cv.fit ¯ transform (data ['cleaned text']).to array () for the Bag-of-Words model or features ¯tf.fit transform (data ['cleaned text']).toarray() for the TF-IDF method. This process converts SMS messages into numerical feature vectors for model training.

• Assign the target labels to target = data['result']. values, then split the dataset into 80% training and 20% testing using train test split () to ensure effective learning and evaluation, followed by initializing the classifiers as gnb model = GaussianNB (), mnb model = MultinomialNB (), and bnb model = BernoulliNB ().

• Train each model using the fit () method on the training dataset, for example, mnb model. Fit (features train, target train), enabling the models to identify patterns distinguishing spam from non-spam messages.

• After training, generate predictions on the test set using predictions = mnb model. Predict (features test).

• Evaluate the model's performance by computing accuracy with accuracy score (target test, predictions) and analyse classification errors using confusion matrix (target test, predictions).

• Assess the model's spam detection effectiveness using precision score (target test, predictions).

• Select the model with the highest precision and accuracy, commonly MNB, and improve its performance if needed by adjusting hyperparameters, refining pre-processing steps, or adding features to enhance classification.

• Save the trained model and vectorizer using pickle.dump() for reuse, and load them with pickle.load() when required for future predictions.

• For classifying new SMS messages, apply the same pre-processing steps, transform the text with the loaded vectorizer, and predict the message type using the trained model for accurate spam detection.

Algorithm 1 SMS Spam Detection

1: function LOAD-DATASET (filePath)

2: data = ReadCSV (file Path)

3: return data

4: end function

5: function PREPROCESS-DATA (data)

6: for each message in data do

7: Convert text to lowercase

8: Remove special characters

9: Remove stop words

10: Apply stemming/lemmatization

11: end for

12: return data

13: end function

14: function VECTORIZE-DATA (data, method)

15: if method ==" BOW" then

16: vectorizer = CountVectorizer()

17: else if method ==" TFIDF" then

18: vectorizer = TfidfVectorizer()

19: end if

20: features=vectorizer.fit transform (data ['cleaned text'])

21: target = data['result']

22: return features, target, vectorizer

23: end function

24: function SPLIT-DATA (features, target)

25: features train, features test, target train, target test = TrainTestSplit(features, target, test size=0.2)

26: return features train, features test, target train, target test

27: end function

28: function TRAIN-MODELS (features train, target train)

29: gnb model = GaussianNB()

30: mnb model = MultinomialNB()

31: bnb model = BernoulliNB()

32: gnb model.fit(features train, target train)

33: mnb model.fit(features train, target train)

34: bnb model.fit(features train, target train)

35: return gnb model, mnb model, bnb model

36: end function

37: function EVALUATE-MODELS (models, features test, target test)

38: for model in models do

39: predictions = model.predict(features test)

40: accuracy = AccuracyScore(target test, predictions)

41: precision = PrecisionScore(target test, predictions)

42: confusionMatrix = ConfusionMatrix(target test, predictions)

43: PrintModelPerformance(model, accuracy, precision, confusionMatrix)

44: end for

45: return SelectBestModel(models)

46: end function

47: function SAVE-MODEL (vectorizer, model)

48: SaveToFile(vectorizer, ”vectorizer.pkl”)

49: SaveToFile(model, ”model.pkl”)

50: end function

5 Outcomes and Interpretation

The results of the SMS spam detection model demonstrate that text classification methods utilizing the Naive Bayes algorithm, especially Multinomial Naive Bayes (MNB) and Bernoulli Naive Bayes (BNB), effectively differentiate spam messages from legitimate ones. Among the three models tested, MultinomialNB achieved the highest precision, reaching 1.0 when using TF-IDF vectorization, meaning it correctly identified all spam messages without falsely labelling legitimate messages as spam. BernoulliNB also performed well, with a precision of 0.94 and an accuracy of 96.42%, making it a strong candidate for real-world applications. The GaussianNB model underperformed compared to the other two, with lower precision and accuracy, likely due to its assumption of normally distributed data, which is not ideal for sparse text data. The confusion matrices further support these findings, showing that MNB had the lowest number of false positives and false negatives, making it the most reliable model for minimizing classification errors. Interpreting these results, it is evident that using TF-IDF vectorization significantly improved classification performance compared to simple bagof-words (Count Vectorizer). TF-IDF assigns importance to rare but meaningful words, allowing the model to focus on critical spam-indicating terms such as” free,”” win,”” prize,” and” urgent. “This led to a model that delivers strong accuracy while reducing misclassification risks, enhancing user experience and trust in automated spam detection systems. The high precision with MNB model can easily make sure that one does not delete their important messages by mistake. But it's still important to balance the two just focusing on precision without having some spam get through. “Subjective” thresholds could be fine-tuned, and more methods (e.g., word embedding method or deep learning-based method) could be added to make the system more robust. In the end, the proposed spam detection model shows promising application in practice, particularly in email filtering, SMS security, and mobile apps.

6 Conclusion

The SMS spam detection model can successfully detect spam text messages by using text pre-processing, vector irisation, and machine learning algorithm classification. By sanitizing the data (stop word removal, lowercasing, stemming), the model makes sure that only useful textual features are feeding into classification. They do feature extraction/ Create a feature vector for each document it will help you convert text data to number values which is then used to fit a machine learning algorithm. Splitting the dataset into training and testing set to train the model on spam characteristics and test it on unseen data. One of the evaluated classifiers, the model of Multinomial Naive Bayes, provides always better precision and accuracy when it comes to recognizing spam messages. The efficiency of the model is evaluated with the help of accurate scores, confusion matrices and precision scores for reliability. Once the best model is chosen, it is serialized by pickle and used for real time spam detection. Combining this model with an informative SMS filtering application could enable users to easily classify spam messages, ensuring safer and less disturbed communications.

References

[1] T. A. Almeida, J. M. G. Hidalgo, and A. Yamakami," Contributions to the study of SMS spam filtering: New collection and results," in Proceedings of the 11th ACM Symposium on Document Engineering, pp. 259-262, 2011. https://doi.org/10.1145/2034691.2034742

[2] J. L. Gomez Hidalgo, G. C. Bringas, E. P. Sanz, and F. C. Garc ´ ´ıa," Content based SMS spam filtering," in Proceedings of the 2006 ACM Symposium on Document Engineering, pp. 107-114, 2006. https://doi.org/10.1145/1166160.1166191

[3] S. J. Delany, M. Buckley, and D. Greene," SMS spam filtering: Methods and data," Expert Systems with Applications, vol. 39, no. 10, pp. 9899- 9908, 2012. https://doi.org/10.1016/j.eswa.2012.02.053

[4] Uysal, A. K., & Gunal, S. (2014). The impact of preprocessing on text classification. Information Processing & Management, 50(1), 104–112. https://doi.org/10.1016/j.ipm.2013.08.006

[5] Altunay, H. C., & Albayrak, Z. (2024). SMS Spam Detection System Based on Deep Learning Architectures for Turkish and English Messages. *Applied Sciences*, *14*(24), 11804. https://doi.org/10.3390/app142411804

[6] Johari, M.F., Chiew, K.L., Hosen, A.R. et al. Key insights into recommended SMS spam detection datasets. Sci Rep 15, 8162 (2025). https://doi.org/10.1038/s41598-025-92223-1

[7] Abrol, K., Mittal, K., Singh, K., Hitesh, & Garg, M. (2024). SMARTSHIELD: A REAL-TIME, LANGUAGE-AWARE SYSTEM FOR SMS SPAM DETECTION. *International Journal of Research -GRANTHAALAYAH*, *12*(8), 174–185. https://doi.org/10.29121/granthaalayah.v12.i8.2024.6101

[8] Hosseinpour, S., & Shakibian, H. (2024). Complex-network based model for SMS spam filtering. Computer Networks, 255, 110892. https://doi.org/10.1016/j.comnet.2024.110892

[9] Basumatary, B., Norbu, T.K., Kokatnoor, S.A., Kumar, S. (2025). Machine Learning Models for SMS Spam Detection. In: Nanda, S.J., Yadav, R.P., Gandomi, A.H., Saraswat, M. (eds) Data Science and Applications. ICDSA 2024. Lecture Notes in Networks and Systems, vol 1266. Springer, Singapore. https://doi.org/10.1007/978-981-96-2647-2_29

[10] Gadde, S., Lakshmanarao, A., & Satyanarayana, S. (2021). SMS spam detection using machine learning and deep learning techniques. *In 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)* (pp. 358–362). IEEE. https://doi.org/10.1109/ICACCS51430.2021.9441783