International Journal Software Engineering and Computer Science (IJSECS)
4 (2), 2024, 767-779
Published Online August 2024 in IJSECS (http://www.journal.lembagakita.org/index.php/ijsecs)
P-ISSN : 2776-4869, E-ISSN : 2776-3242. DOI: https://doi.org/10.35870/ijsecs.v4i2.2681.

RESEARCH ARTICLE Open Access

Optimization of K Value in KNN Algorithm for Spam and HAM Classification in SMS Texts

Ferryma Arba Apriansyah *
Information Technology Study Program-Masters Program, Universitas Teknologi Yogyakarta, Special Region of Yogyakarta, Indonesia.
Corresponding Email: ferryma.arba@gmail.com.

Arief Hermawan
Information Technology Study Program-Masters Program, Universitas Teknologi Yogyakarta, Special Region of Yogyakarta, Indonesia.
Email: ariefdbuty@gmail.com.

Donny Avianto
Information Technology Study Program-Masters Program, Universitas Teknologi Yogyakarta, Special Region of Yogyakarta, Indonesia.
Email: donny@uty.ac.id.

Received: June 15, 2024; Accepted: July 30, 2024; Published: August 20, 2024.

Abstract: Spam refers to the unsolicited and repetitive sending of messages to others via electronic devices without their consent. This activity, commonly known as spamming, is typically carried out by individuals referred to as spammers. SMS spam, which often originates from unknown sources, frequently contains advertisements, phishing attempts, scams, and even malware. Such spam messages can be pervasive, affecting almost all mobile phone numbers, thereby causing significant disruptions to communication by delivering irrelevant content. The persistent nature of spam messages underscores the need for effective filtering mechanisms. This study investigates the application of the K-Nearest Neighbors (KNN) algorithm for classifying SMS messages as either spam or non-spam (ham). The findings demonstrate that KNN, when optimized through various methods for determining the appropriate value of K, can achieve an impressive average accuracy of 99.16% in classifying SMS spam. This high level of accuracy indicates that KNN is a reliable method for spam detection.

Keywords: Classification; KNN; SMS Spam.

1. Introduction

Today, millions of mobile phone users worldwide rely on Short Message Service (SMS) for daily communication. The popularity of SMS is driven by several factors, including ease of use, simplicity, fast delivery, and relatively low cost [1]. Despite ongoing advancements in communication technology, particularly with the advent of smartphones that support various instant messaging applications, SMS remains crucial, especially for critical functions like mobile banking authentication and device synchronization.

In the past, SMS was a highly favored means of communication before gradually being replaced by instant messaging services through social media platforms. However, with technological progress, new challenges have emerged, one of which is the influx of spam messages via SMS. Spam is defined as unsolicited and repetitive messages sent in bulk to recipients, usually for commercial or fraudulent purposes. According to The Spam Track at the Text Retrieval Conference (TREC), spam is characterized as unwanted messages received without the recipient's consent, which can be distributed by the sender without geographical limitations [2]. SMS spam often contains product advertisements or promotions and may also pose serious threats, such as phishing, scams, or even malware. The prevalence of SMS spam presents a significant issue due to its disruptive nature. Spam messages not only drain the recipient's time and attention but also potentially lead to financial loss or system damage. Therefore, developing effective methods to handle and filter SMS spam is of utmost importance. This study focuses on devising efficient techniques to classify SMS messages as either spam or ham (non-spam). The objective is to create a system capable of automatically filtering messages and categorizing them correctly.

In recent years, various approaches have been proposed to address this issue. Jain *et al*. (2019), for instance, employed Long Short-Term Memory (LSTM), a variant of recurrent neural networks, to filter SMS spam [2]. Their model achieved an accuracy rate of 99.01% by utilizing 200 LSTM nodes and 6000 features. Additionally, they implemented three different word embedding techniques, namely Word2Vec, WordNet, and ConcepNet, to enhance classification effectiveness. However, using these techniques also significantly increased the system's processing load. To overcome this challenge, various machine learning algorithms, such as Adaboost, Naive Bayes (NB), Random Forest (RF), Support Vector Machine (SVM), and deep learning-based voting approaches like Convolutional Neural Networks (CNN), have been applied, with results showing an accuracy of 98.51%.

Meanwhile, Jindal and Liu (2007) proposed a model for filtering spam in product blog advertisements, which categorized spam into different types. However, their approach did not account for the unique characteristics of SMS spam [3]. A more recent classification by Jiang *et al*. (2016) divided unauthorized messages into four main categories: traditional spam, fake reviews, social spam, and link farming [4]. Although this taxonomy covers various types of spam, SMS spam is still underexplored in their research. The variation in outcomes from different studies suggests that the classification methods employed significantly impact the effectiveness of spam filtering. Consequently, further research is needed to compare different classification methods and identify the most suitable approach for SMS spam [5].

In this study, the research team applied the K-Nearest Neighbors (KNN) method to classify SMS messages as spam or ham. The value of K in this algorithm was optimized using a clustering approach based on frequency distribution. The model's performance was evaluated using a Confusion Matrix, which provides insights into the precision, recall, and accuracy of the classification results [6]. With this method, the research team aims to find a more effective and efficient solution for filtering SMS spam, thereby reducing the negative impact of spam on mobile phone users.

2. Research Method

The research methodology employed in this study follows a systematic approach using the K-Nearest Neighbors (KNN) algorithm for classifying SMS messages into spam and ham (non-spam). The methodology encompasses several stages, including system design, data acquisition, preprocessing, the application of the KNN algorithm, and the evaluation and validation of the results obtained. Each stage is crucial in ensuring the accuracy and effectiveness of the classification process. The detailed steps involved in this methodology are outlined in the following sections.

2.1. System Design

The system design for this research is depicted in Figure 1. The system is structured into several stages: starting from data collection, preprocessing, classification, and finally, the output, which determines whether

the message is classified as spam or ham. This modular approach allows for flexibility and scalability, making it easier to adapt or refine individual components without affecting the overall system functionality.

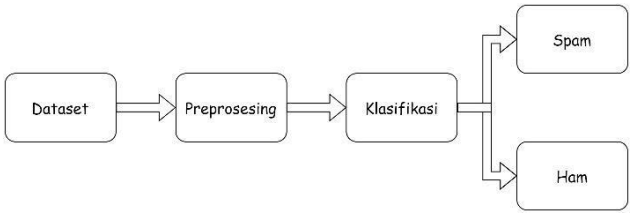

Figure 1. System Design

The figure illustrates the workflow from data acquisition through preprocessing stages, leading to the classification process using the KNN algorithm, and finally, the validation of the classification results.

2.2. Dataset

The dataset used in this research comprises 5,566 SMS documents written in English, each labeled as either spam or ham. The dataset was sourced from the public repository available at https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset. The dataset includes a variety of SMS content, ranging from typical promotional messages to regular communication, providing a robust basis for training and testing the classification model.

Table 1. Attributes in the Database

Category	Message
ham	Ok lar... Joking wif u oni...
ham	U dun say so early hor... U c already then say...
spam	XXXMobileMovieClub: To use your credit, click the WAP link in the next txt message or click here>> h...
spam	07732584351 - Rodger Burns - MSG = We tried to call you re your reply to our sms for a free nokia mo...

This table lists the attributes present in the dataset. The dataset is divided into a training set (87.37%) and a testing set (12.63%). The distribution includes 702 spam messages and 4,864 ham messages, giving a well-rounded dataset for model training and evaluation. An example of the content used in this dataset is: "Go until Jurong point crazy. Available only in Bugis n great world la e buffet..Cine there got..."

2.3. Preprocessing

Preprocessing is a critical stage in data preparation, aimed at optimizing the dataset for better performance in the classification task. In this research, the dataset was manually converted from the default .txt format to .csv format to facilitate easier processing. The following subsections outline the standard procedures followed during the preprocessing stage [7]:

1) Case Folding
This step involves converting all characters in the text to lowercase to maintain uniformity. This step is crucial as it prevents the algorithm from treating the same word differently based on case sensitivity. Figure 2 provides an example of the case folding process.

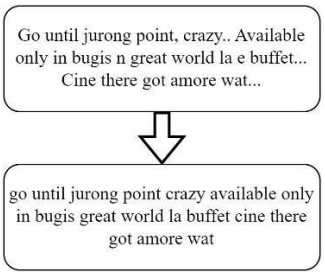

Figure 2. Case Folding Process

The figure demonstrates how text is standardized by converting all characters to lowercase.

2) Data Cleaning
In the data cleaning stage, non-alphabetic characters such as punctuation marks and numbers are removed from the text. This process ensures that the dataset is free from noise that could interfere with the accuracy of the classification model. Figure 3 illustrates the data cleaning process.

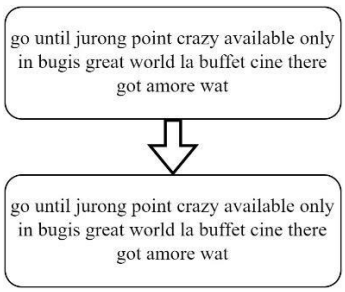

Figure 3. Data Cleaning Process

This figure shows the removal of unnecessary symbols and numbers, leaving only the relevant textual data.

3) Stopword Removal
Stopword removal is used to eliminate common words that do not contribute meaningful information to the classification process, such as "the," "and," "is," etc. These words are removed to reduce the dimensionality of the dataset and avoid redundancy. Figure 4 exemplifies the stopword removal process.

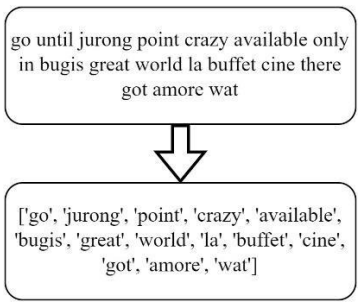

Figure 4. Stopword Removal Process

The figure illustrates how non-essential words are filtered out, leaving only the words that are significant for analysis.

4) Stemming
Stemming reduces words to their base or root form, simplifying variations of words to a single form. This

process is implemented using the Porter Stemmer algorithm, which is widely recognized for its effectiveness in natural language processing tasks. Figure 5 shows an example of the stemming process.

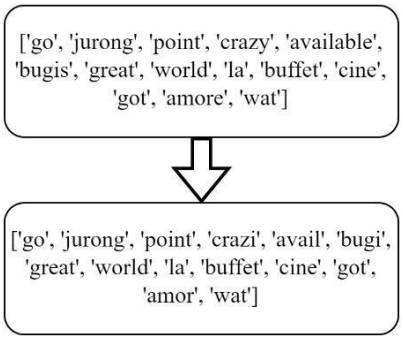

Figure 5. Stemming Process

The figure depicts how different forms of a word are reduced to their base form, enhancing the consistency of the dataset.

5) Tokenization
Tokenization is the process of splitting the text into individual tokens or words. This step is essential for converting the text into a format that can be processed by the classification algorithm. Figure 6 provides an example of the tokenization process.

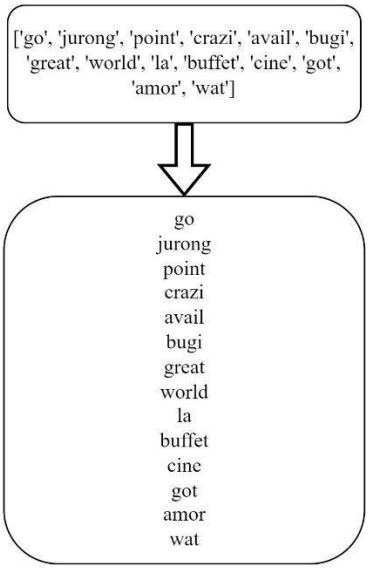

Figure 6. Tokenization Process

The figure shows how the text is broken down into individual tokens for further processing.

6) TF-IDF Weighting
Term Frequency-Inverse Document Frequency (TF-IDF) weighting is used to convert the textual data into numerical vectors, which represent the importance of words within the document relative to the entire dataset. This weighting scheme helps in highlighting words that are more relevant to the classification task. Figure 7 illustrates the TF-IDF calculation process.

Term	TF-IDF						
	D1	D2	D3	D4	D5	D6	D7
Go	0.544	0	0	0.544	0	0	0
jurong	0.544	0	0	0	0.544	0	0
point	0.544	0	0	0.544	0	0	0
crazi	0.544	0.544	0	0	0	0	0
avail	0.544	0	0	0	0	0.544	0
bugi	0.544	0	0	0	0	0	0.544
great	0.544	0.544	0	0	0	0	0
world	0.544	0	0	0	0.544	0	0
la	0.243	0.243	0	0	0	0.486	0.243
buffet	0.845	0	0	0	0	0	0
cine	0.368	0	0	0.368	0.368	0	0
got	0.368	0	0.368	0	0	0	0.368
amor	0.544	0	0	0	0.544	0	0
wat	0.544	0	1.088	0	0	0	0

Figure 7. TF-IDF Calculation

The figure demonstrates how TF-IDF scores are calculated, emphasizing words that have higher relevance in the document compared to others.

2.4. Classification Using the KNN Method

The K-Nearest Neighbors (KNN) algorithm is employed to classify the SMS data based on the nearest distance between data points. Determining the optimal number of neighbors (K) is crucial, as it significantly impacts the model's performance. Typically, odd numbers such as 1, 3, and 5 are chosen for K to avoid ties in classification decisions [6]. The choice of K depends on the size and dimensionality of the dataset. Smaller values of K are generally more effective for larger datasets, while larger values of K might be better suited for data with higher dimensions. The pseudocode for the KNN algorithm is as follows:

```
k-Nearest Neighbor
Classify (X, Y, x) // X: training data, Y: class labels of X, x: unknown sample
for i = 1 to m do
   Compute distance d(Xi, x)
end for
Compute set I containing indices for the k
smallest distances d(Xi, x).
return majority label for [Y, where i € I]
```

Distance measurement is a key aspect of the KNN algorithm, as it defines how the similarity between data points is quantified. In this research, the Euclidean distance is used, which is a straightforward measure of the straight-line distance between two points in a multidimensional space. Given that the text data has been converted into numerical vectors during the TF-IDF preprocessing stage, Euclidean distance calculation is applicable. The Euclidean distance formula is as follows:

$$d(x^1, x^2) = \sqrt{\sum_{i=1}^{p} (X1_i - X2_i)^2}$$

Where:
$d(x^1, x^2)$ represents the Euclidean distance,
$X2_i$ is the test data value for the i-th variable at point x2,
$X1_i$ is the sample value at point x1, and
p is the number of dimensions for the independent variables.

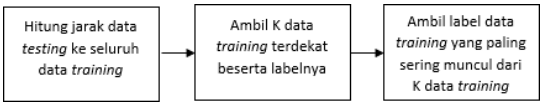

Figure 8. KNN Process Flow

The figure illustrates the flow of the KNN algorithm, from data input through distance calculation, to the final classification decision based on the majority label.

2.5. Optimization Using Frequency Distribution Clustering

To enhance the accuracy of the KNN algorithm, it is necessary to fine-tune the parameters to obtain the optimal value of K. One approach to optimization is by clustering the data based on frequency distribution. This method involves grouping the data into different categories based on the frequency with which certain features appear. Each data point is then assigned to a single category, and tabulation is performed using the class labels and their corresponding frequencies. This clustering approach is expected to improve the overall classification accuracy by ensuring that the most relevant features are considered.

2.6. Evaluation and Validation of Results

Evaluation of the classification model is conducted using the 10-fold cross-validation technique. In this method, the experimental data is divided into ten subsets, with one subset used for testing and the remaining nine used for training [10]. This process is repeated ten times, with each subset serving as the testing set once, ensuring that the model is rigorously tested across the entire dataset. The accuracy of the classification is assessed using the Confusion Matrix, which provides a detailed breakdown of true positives (TP), false negatives (FN), false positives (FP), and true negatives (TN). This matrix is also used to calculate precision and recall, which are critical metrics for evaluating the performance of the classification model [11]. Additionally, the time required for the classification process is measured to assess the computational efficiency of the model. The table below illustrates the Confusion Matrix used to evaluate the classification results:

Table 2. Confusion Matrix

Actual	Predicted	
	Positive	Negative
Positive Class	TP (True Positive)	FN (False Negative)
Negative Class	FP (False Positive)	FP (False Positive)

This table helps in understanding the classification performance by providing insights into the number of correct and incorrect classifications made by the model.

3. Result and Discussion

3.1 Results

In this study, the K-Nearest Neighbors (KNN) algorithm was employed with odd values of K ranging from 1 to 13. The primary objective was to evaluate whether the model's performance remained stable or exhibited variation across different K values [11]. The experiments conducted assessed the accuracy, recall, precision, and processing time of the KNN algorithm in classifying SMS messages as either spam or ham. The results of these evaluations are illustrated in Figure 9.

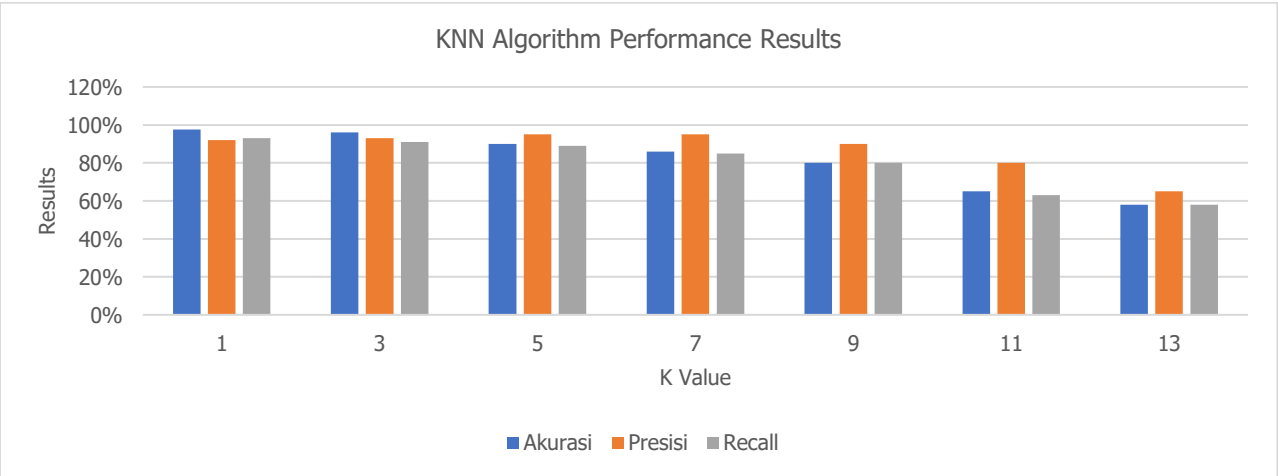

Figure 9. KNN Algorithm Performance Results

Figure 9 presents the findings, revealing that the accuracy, precision, and recall metrics do not align perfectly but instead overlap, indicating a strong correlation between accuracy and recall. The high accuracy

values suggest that the model reliably classified the SMS messages based on the labels provided by the users. The results demonstrated that among the tested values, K=1 was the most optimal choice. Specifically, the model achieved an accuracy of 97.5%, precision of 92%, and recall of 93%. These percentages were obtained through ten iterations, with the data being randomized for each test. From the graphical analysis in Figure 9, it is observed that K=1 and K=3 required approximately 98 seconds to process the SMS spam data, whereas higher values of K necessitated longer processing times, ranging between 107 and 208 seconds. This finding highlights a significant difference in the time efficiency and classification accuracy depending on the selected K value.

According to the results from the KNN classification method, as depicted in Figure 9, higher K values generally resulted in reduced accuracy, precision, and recall [13]. The study further indicates that as the value of K increases, the time required for data processing also increases. However, lower K values tend to yield better classification results [14]. To achieve optimal performance at lower K values within the KNN framework, further optimization was carried out. In this case, the Term Frequency-Inverse Document Frequency (TF-IDF) weighting process was applied to the 5,566 documents in the dataset, resulting in word occurrences ranging from 100 to 1,586 times [15]. Following this, the data was clustered based on frequency distribution, with the aim of categorizing the documents according to the frequency range of word occurrences in each document.

Table 3. Frequency Distribution Clustering

Range of Word Frequency	Number of Documents
D1: 1 – 320	4000
D2: 321 – 640	766
D3: 641 – 960	410
D4: 961 – 1280	290
D5: 1281 – 1600	100
Total	5566

Table 3 provides the results of the data clustering based on frequency distribution. The table illustrates five frequency ranges (D1-D5), each with an interval of 320. The clustering is further represented visually through a histogram, as shown in Figure 10.

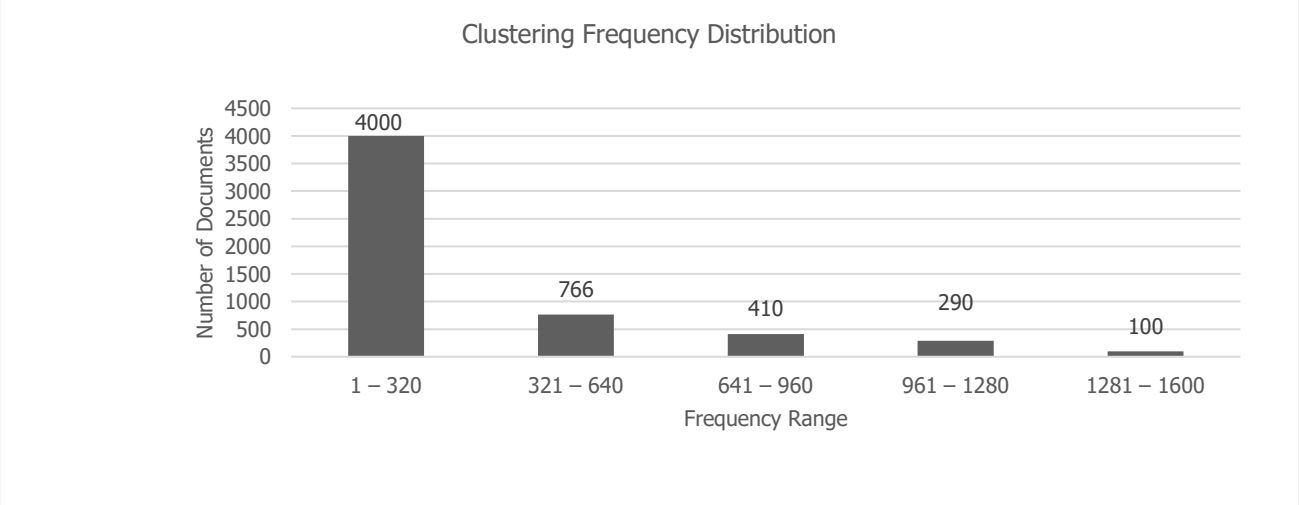

Figure 10. Frequency Distribution Clustering Histogram

Using the existing frequency distribution, the research team identified five effective models for grouping data based on the frequency ranges shown in Table 3. These models led to the formulation of nine research scenarios aimed at optimizing the smallest K value (K=1) within the KNN algorithm to enhance accuracy. The scenarios include:

1) Scenario 1: D1
2) Scenario 2: D2
3) Scenario 3: D3
4) Scenario 4: D4
5) Scenario 5: D5

6) Scenario 6: D1, D2
7) Scenario 7: D1, D2, D3
8) Scenario 8: D1, D2, D3, D4
9) Scenario 9: D1, D2, D3, D4, D5

These scenarios represent the outcomes of classification using the KNN algorithm with the smallest possible K value.

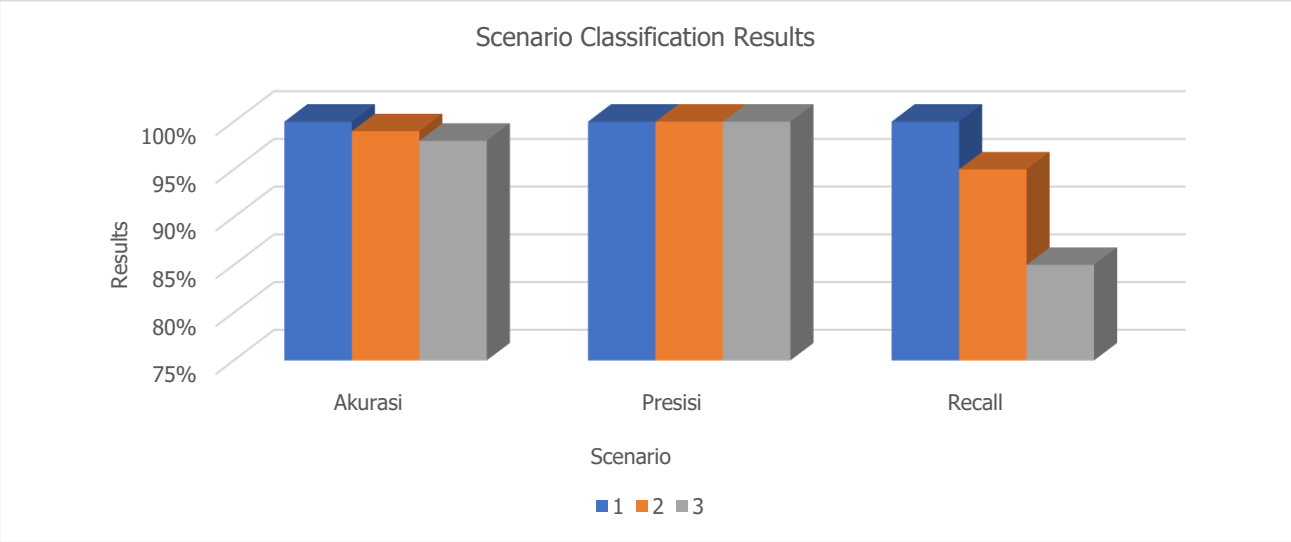

Figure 11. Scenario Classification Histogram

Figure 11 shows that Scenario 1 achieved higher levels of accuracy, precision, and recall compared to previous KNN classification results. In terms of precision, all scenarios achieved an outstanding 100%, indicating that the system effectively identified spam. For recall, Scenario 1 yielded the best result at 99.86%, while the average accuracy across all scenarios was 99.16%, demonstrating the system's high capability in accurately identifying spam.

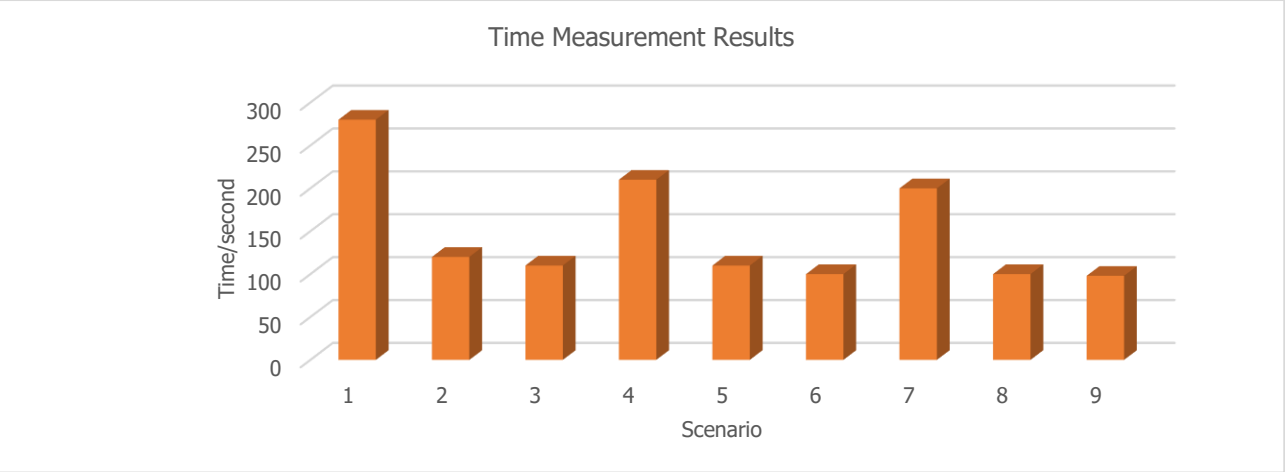

Figure 12. Processing Time Measurement Histogram

Figure 12 indicates that Scenario 1 required the most processing time compared to the other scenarios. In contrast, Scenario 9 had the fastest processing time. These findings suggest that while the KNN algorithm is highly effective for classifying SMS spam and ham, with an overall accuracy rate of 99.16%, the processing time varies depending on the scenario. In addition to these results, a Confusion Matrix was generated, as shown in Figure 13.

	Predicted 0	Predicted 1
Actual 0	1559	28
Actual 1	17	235

Figure 13. Confusion Matrix

The Confusion Matrix in Figure 13 reveals that there were 1,559 true positives and 235 true negatives, while the number of false positives and false negatives were only 18 and 28, respectively. This indicates that the model was highly effective in correctly categorizing most of the messages as either spam or ham. Furthermore, the accuracy score of 97.5% underscores the model's efficiency.

3.2 Discussion

The findings from this study indicate that the K-Nearest Neighbors (KNN) algorithm is a highly effective method for classifying SMS messages into spam and ham categories, particularly when optimized with appropriate values of K and supplemented by advanced preprocessing techniques like TF-IDF weighting and frequency distribution clustering. The experiment conducted with varying K values (from K=1 to K=13) revealed that the algorithm's performance is highly sensitive to the choice of K. The results showed that lower values of K, particularly K=1, yielded the best performance in terms of accuracy, precision, and recall. Specifically, K=1 achieved an accuracy of 97.5%, precision of 92%, and recall of 93%. These metrics are critical in the context of spam detection, where the ability to accurately identify spam messages (high recall) without misclassifying legitimate messages as spam (high precision) is essential. The findings align with the theoretical understanding of KNN, where a lower K value tends to capture the local structure of the data more effectively, leading to better classification results in cases where the data points are well separated. However, as K increases, the model starts to consider more neighbors, which can introduce noise into the classification process, leading to reduced accuracy and increased processing time. This was evident in the study, where higher K values (e.g., K=13) resulted in longer processing times and lower classification performance.

The introduction of frequency distribution clustering further enhanced the classification accuracy. By grouping the data based on the frequency of word occurrences, the model was able to focus on the most relevant features for each cluster, thereby improving the overall effectiveness of the KNN algorithm. The clustering process led to the identification of five distinct models, each corresponding to a specific frequency range (D1-D5). These models were used to create nine different scenarios, with Scenario 1 (which used the D1 cluster) achieving the highest recall of 99.86% and a precision of 100%. This approach highlights the importance of data preprocessing and feature selection in text classification tasks. The TF-IDF weighting method used in this study was crucial in transforming the text data into a numerical format that could be effectively processed by the KNN algorithm. By assigning weights to words based on their frequency and importance, TF-IDF helps to differentiate between commonly used words (which are less informative) and more distinctive terms that are critical for accurate classification.

Another key finding from the study is the relationship between the K value and the processing time required for classification. Lower K values, while yielding higher accuracy, were also associated with shorter processing times. This is an important consideration for real-time applications, where the speed of classification is as critical as its accuracy. Scenario 9, which utilized all five frequency clusters, demonstrated the fastest processing time, suggesting that clustering not only improves accuracy but also enhances computational efficiency by reducing the complexity of the classification task.

However, the trade-off between processing time and classification accuracy must be carefully managed, especially in large-scale applications where the volume of data can significantly impact performance. The study's results suggest that an optimal balance can be achieved by carefully selecting the K value and employing effective clustering techniques. The Confusion Matrix provided additional insights into the model's performance. With 1,559 true positives and 235 true negatives, the model demonstrated a high level of accuracy in correctly identifying both spam and ham messages. The low number of false positives (18) and false negatives (28) further supports the effectiveness of the KNN algorithm in this context. The accuracy score of 97.5% underscores the robustness of the model, particularly when used in combination with the optimization techniques explored in this study.

When compared to previous studies, such as those by Jain *et al.* (2019) and Jindal and Liu (2007), which employed more complex models like LSTM and deep learning-based methods, the KNN algorithm's performance is competitive, especially considering its simplicity and lower computational requirements [2][3]. While advanced neural networks can achieve slightly higher accuracy, they often require more extensive

resources and longer training times. The KNN algorithm, on the other hand, offers a more accessible and efficient alternative, particularly for applications with limited computational resources.

The results of this study have significant implications for the deployment of SMS spam filters in real-world settings. The high accuracy, precision, and recall achieved with the optimized KNN model suggest that it could be effectively implemented in mobile communication systems to protect users from spam. Moreover, the relatively low computational burden makes it suitable for integration into existing systems without the need for substantial hardware upgrades. Despite the promising results, there are several limitations to this study that should be addressed in future research. Firstly, the dataset used in this study was limited to English-language SMS messages, which may not generalize well to other languages or dialects. Future studies could explore the application of the KNN algorithm to multilingual datasets to assess its versatility. Additionally, while the study focused on optimizing the K value and using frequency distribution clustering, other optimization techniques, such as feature selection or dimensionality reduction, could be investigated to further enhance performance. Another area for future research is the exploration of hybrid models that combine KNN with other algorithms, such as Support Vector Machines (SVM) or neural networks, to leverage the strengths of multiple approaches. This could lead to the development of more robust spam detection systems that can handle a wider variety of spam types and formats. This study demonstrates that the KNN algorithm, when properly optimized, is a powerful tool for SMS spam classification. The combination of TF-IDF weighting, frequency distribution clustering, and careful selection of the K value resulted in high accuracy, precision, and recall, making it a viable option for real-world spam detection applications. The findings also highlight the importance of preprocessing and optimization in enhancing the performance of text classification algorithms, offering valuable insights for future research and development in this field.

4. Related Work

The detection and classification of spam, particularly within SMS communications, has been a focal point of research in the field of text classification and machine learning. Various methods and algorithms have been developed to enhance the accuracy and efficiency of spam detection systems. This section reviews the relevant literature, focusing on the approaches applied to SMS spam classification, including machine learning techniques, deep learning models, and hybrid methods. Machine learning has been instrumental in spam detection research due to its ability to learn from data and improve model performance over time. Traditional algorithms such as Naive Bayes (NB), Support Vector Machines (SVM), and Decision Trees have been widely used for classifying text data, including SMS messages.

Naive Bayes has been a popular choice due to its simplicity and effectiveness in handling large datasets. Ling *et al*. (2014) demonstrated the application of Naive Bayes in sentiment analysis, showing that it can be effective in classification tasks where text features are well-represented [8]. However, the model's assumption of word independence can limit its effectiveness in more complex scenarios where word relationships are important. Support Vector Machines (SVM) have also been extensively studied for spam classification. SVMs are known for their robustness in high-dimensional spaces, which is advantageous in text classification tasks where the feature space can be vast due to a large vocabulary. Jain *et al*. (2019) applied SVM in optimizing semantic LSTM for spam detection, achieving high accuracy by effectively separating spam and non-spam messages in the feature space [2]. Although SVMs offer strong performance, they require significant computational resources, especially with large datasets, which can be a drawback for SMS spam detection on mobile devices with limited processing power.

The advent of deep learning has introduced more sophisticated models to address the limitations of traditional machine learning algorithms. Long Short-Term Memory (LSTM) networks, a type of recurrent neural network (RNN), have shown great potential in handling sequential data like text. Jain *et al*. (2019) employed LSTM networks for SMS spam detection, leveraging the model's ability to capture long-term dependencies in text. Their approach involved using word embeddings such as Word2Vec, WordNet, and ConcepNet to represent words as vectors in a continuous space, which were then fed into the LSTM model. This method achieved an accuracy of 99.01%, demonstrating the effectiveness of deep learning in improving spam detection [2]. However, the computational complexity of LSTM models can be a limiting factor, particularly for deployment on mobile devices with constrained computational resources. Another deep learning approach involves Convolutional Neural Networks (CNNs), traditionally used in image processing but also adapted for text classification tasks. Roul *et al*. (2018) introduced a modified TF-IDF term weighting strategy in combination with CNNs to improve text categorization, demonstrating the adaptability of CNNs in capturing local dependencies in text [5]. This approach can be particularly useful in spam detection, where specific

phrases or word combinations are key indicators of spam. However, CNNs' performance can vary depending on the choice of hyperparameters and the quality of the word embeddings used.

Hybrid approaches that combine multiple algorithms or techniques have also been explored to leverage the strengths of different models. For example, Jindal and Liu (2007) proposed a hybrid model that combined rule-based methods with machine learning algorithms for review spam detection [3]. This approach could be adapted for SMS spam by incorporating features specific to mobile communication, such as message length, sender information, and frequency of message receipt. In SMS spam detection, hybrid models have been less extensively studied, but there are promising directions. The combination of traditional machine learning models with deep learning techniques, as suggested by Tamil and Andhra (2020), can provide a more robust approach to spam detection by capturing both the statistical properties of text and the complex patterns that deep learning models can identify [9].

Several studies have conducted comparative analyses of different spam detection algorithms to identify the most effective approaches. Christanti *et al.* (2018) compared K-Nearest Neighbor (KNN) with Neighbor Weighted K-Nearest Neighbor (NW-KNN) on sentiment analysis tasks, highlighting the trade-offs between accuracy, processing time, and computational efficiency [6]. Similar studies in SMS spam detection have emphasized the importance of balancing accuracy with resource consumption, particularly in mobile environments where computational power may be limited. Moreover, the development of publicly available datasets, like the UCI SMS Spam Collection, has facilitated the benchmarking of different algorithms and the reproducibility of research findings. These datasets provide a standardized basis for comparing the performance of various models and have contributed to the advancement of the field [6].

Despite the progress made in SMS spam detection, several challenges remain. One of the primary challenges is the adaptability of spam detection models to new and evolving spam tactics. As spammers continuously change their strategies to evade detection, models need to be regularly updated with new data and potentially retrained to maintain their effectiveness [10][11]. Another challenge is the deployment of these models on mobile devices with limited computational resources. While deep learning models offer high accuracy, their implementation on mobile platforms can be hindered by their resource requirements. Future research could explore the use of lightweight models or model compression techniques to address this issue [12][13]. Lastly, the multilingual nature of SMS communication poses a challenge for spam detection models, which are often trained on datasets in a single language. Expanding research to include multilingual datasets and developing models capable of handling multiple languages would significantly enhance the applicability of spam detection systems worldwide [14][15]. The field of SMS spam detection has seen significant advancements through the application of both traditional machine learning algorithms and modern deep learning techniques. While each approach has its strengths and limitations, hybrid models and comparative studies offer valuable insights into optimizing spam detection systems for real-world use. The ongoing challenges underscore the need for continued research and innovation to keep pace with the evolving nature of spam and the growing diversity of SMS communication.

5. Conclusion

This study demonstrates that the K-Nearest Neighbors (KNN) method is effective for classifying SMS messages as either spam or ham. The results indicate that using K=1 yields an accuracy of 97.5%, which is higher compared to larger K values, along with a precision of 92% and a recall of 93%, with a processing time of 98 seconds. Optimizing the K value in KNN can further enhance accuracy, one approach being to categorize data into several clusters before the classification process. Frequency distribution clustering is a technique that can be employed for this optimization. The findings reveal that the first scenario, optimized using frequency distribution clustering, achieved an accuracy of 100%, while the second and third scenarios reached 99% and 98% accuracy, respectively. In terms of processing time, the results indicate that as the complexity of the model combinations in each scenario increases, so does the processing time required. Based on these findings, it can be concluded that scenarios employing frequency distribution clustering optimization yield highly favorable results.

References

[1] Nanja, M., & Purwanto, P. (2015). Metode K-Nearest Neighbor berbasis forward selection untuk prediksi harga komoditi lada. *Pseudocode, 2*(1), 53–64. https://doi.org/10.33369/pseudocode.2.1.53-64

[2] Jain, G., Sharma, M., & Agarwal, B. (2019). Optimizing semantic LSTM for spam detection. *International Journal of Information Technology*, *11*, 239-250. https://doi.org/10.1007/s41870-018-0157-5.

[3] Jindal, N., & Liu, B. (2007, May). Review spam detection. In *Proceedings of the 16th international conference on World Wide Web* (pp. 1189-1190).

[4] Jiang, M., Cui, P., & Faloutsos, C. (2016). Suspicious behavior detection: Current trends and future directions. *IEEE intelligent systems*, *31*(1), 31-39. https://doi.org/10.1109/MIS.2016.5.

[5] Roul, R. K., Sahoo, J. K., & Arora, K. (2018). Modified TF-IDF term weighting strategies for text categorization. In *2017 14th IEEE India Council International Conference (INDICON)* (no. October). https://doi.org/10.1109/INDICON.2017.8487593

[6] Martha, M., Christanti, V., Naga, D. S., & Rompas, P. T. D. (2018). Perbandingan Pengklasifikasi k-Nearest Neighbor dan Neighbor-Weighted k-Nearest Neighbor Pada Sistem Analisis Sentimen dengan Data Microblog. *FRONTIERS: JURNAL SAINS DAN TEKNOLOGI*, *1*(1). https://doi.org/10.36412/frontiers/001035e1/april201801.08

[7] Irfa, A. A., Adiwijaya, A., & Mubarok, M. S. (2018). Klasifikasi Topik Berita Berbahasa Indonesia Menggunakan k-Nearest Neighbor. *eProceedings of Engineering*, *5*(2).

[8] Ling, J., Kencana, I. P. E. N., & Oka, T. B. (2014). Analisis sentimen menggunakan metode Naïve Bayes Classifier dengan seleksi fitur Chi Square. *E-Jurnal Matematika, 3*(3), 92. https://doi.org/10.24843/mtk.2014.v03.i03.p070

[9] Tamil, N., & Andhra, P. (2020). Classification of social media text spam using VAE-CNN and LSTM mode. *Ingénierie des Systèmes d'Information, 25*(6), 747-753.

[10] Widyasanti, N. K., Putra, I. D., & Rusjayanthi, N. D. (2018). Seleksi Fitur Bobot Kata dengan Metode TFIDF untuk Ringkasan Bahasa Indonesia. *J. Ilm. Merpati (Menara Penelit. Akad. Teknol. Informasi)*, *6*(2), 119.

[11] Zuviyanto, E., Adji, T. B., & Setiawan, N. A. (2018). Perbandingan Algoritme-algoritme Pembelajaran Mesin pada Klasifikasi SMS Spam. *Prosiding SENIATI*, *4*(3), 20-26. https://doi.org/10.36040/seniati.v4i3.1350.

[12] Muzakki, M. A. (2020). Klasifikasi dan Analisa Sentimen Kuesioner Fasilitas dan Layanan untuk Universitas Qomaruddin Gresik. *Journal of Computer Science and Visual Communication Design*, *5*(2), 68-76.

[13] Ramadhan, R., Sari, Y. A., & Adikara, P. P. (2021). Perbandingan Pembobotan Term Frequency-Inverse Document Frequency dan Term Frequency-Relevance Frequency terhadap Fitur N-Gram pada Analisis Sentimen. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, *5*(11), 5075-5079.

[14] Herwijayanti, B., Ratnawati, D. E., & Muflikhah, L. (2018). Klasifikasi Berita Online dengan menggunakan Pembobotan TF-IDF dan Cosine Similarity. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, *2*(1), 306-312.

[15] Pramartha, G. S., Shaufiah, S., & Bijaksana, M. A. (2015). Analisis Dan Implementasi Algoritma Graph-basedk-nearest Neighbour Untuk Klasifikasi Spam Pada Pesan Singkat. *eProceedings of Engineering*, *2*(2).