Vol. 4 No. (1) (June 2026) 72-80

Journal of Data Insights

http://journalnew.unimus.ac.id/index.php/jodi

Hate Speech Detection on X Using K-Nearest Neighbor with TF–IDF and Cosine Similarity

Faiq Madani[*1], Arvanida Feizal Permana[2], Abdul Karim[3], Riyagung Nuryusufa Tranggono Adi Prasetya[4], Wendy Sarasjati[5]

[1]Department of Artificial Intelligence, Universitas Muhammadiyah Semarang, Indonesia
[2]Prospective Technology of Electrical Engineering and Computer Science, National Chin Yi University of Technology, Taiwan
[3]College of Information ScienceAI-X, Hallym University, Chuncheon, South Korea
[4]Graduate Institute of Information Engineering, Kun Shan University, Taiwan
[5]Master of Informatics, Universitas Muhammadiyah Semarang, Indonesia

DOI: https://doi.org/10.26714/jodi.v4i1.1149

Article Info

Article history:
Received May 24, 2026
Revised June 16, 2026
Accepted June 20, 2026

Keywords:
Cosine Similarity; Hate Speech Detection; K-Nearest Neighbor; TF–IDF; Text Classification.

Abstract

The widespread use of social media has facilitated the rapid dissemination of hate speech, posing significant challenges to social harmony, public safety, and online communication environments. Hate speech can promote discrimination, incite hostility, and negatively affect individuals and communities, making its detection an important issue in digital society. As one of the largest social networking platforms, X (formerly Twitter) enables users to share opinions publicly, creating a large volume of user-generated content that requires effective monitoring mechanisms. Therefore, automated hate speech detection systems have become essential for supporting content moderation and reducing the spread of harmful online content. This research proposes a hate speech classification approach using the K-Nearest Neighbor (KNN) algorithm combined with Term Frequency–Inverse Document Frequency (TF–IDF) weighting and Cosine Similarity. The dataset consists of 900 social media posts collected through the platform API and manually labeled into hate speech and non-hate speech categories, consisting of 675 training data and 225 testing data. Prior to classification, text preprocessing techniques including tokenization, stopword removal, and stemming were applied to improve text quality. Model evaluation was conducted using 10-fold cross validation to assess classification performance. Experimental results showed that the KNN algorithm with Cosine Similarity distance measurement and K=3 parameter achieved an accuracy of 78.22% in hate speech detection tasks. The findings indicate that KNN combined with TF–IDF and Cosine Similarity provides a reliable approach for social media text classification and can support automated hate speech detection systems.

✉ Correspondence Address:
E-mail: faiqmadani@unimus.ac.id

e-ISSN: 2988 - 2109

INTRODUCTION

The rapid advancement of information technology has significantly transformed communication patterns in modern society. Social media platforms have become one of the primary communication channels, enabling users to exchange information, opinions, and ideas without geographical limitations. However, alongside these advantages, social media growth has also increased the spread of harmful online content, including hate speech, cyberbullying, and offensive language. Hate speech is generally defined as expressions intended to attack, insult, provoke, or discriminate against individuals or groups based on race, ethnicity, religion, gender, nationality, or other social identities [1]. The widespread dissemination of hate speech on social media can negatively affect social harmony and public interaction, making automated hate speech detection increasingly important.

X (*formerly Twitter*) is one of the largest social networking platforms widely used for public discussions and information sharing. The platform allows users to express opinions through short textual content, creating large volumes of user-generated data daily. While this openness supports information exchange, it also creates challenges in content moderation because harmful content can spread rapidly and reach large audiences within a short time. Manual moderation approaches often become inefficient due to the scale of online activity, creating the need for intelligent automated systems capable of identifying hate speech content effectively [2].

Various machine learning and deep learning approaches have been proposed to address hate speech classification problems. Recent studies have reported promising results using deep learning architectures and transformer-based models for detecting harmful online content[10]. However, these approaches often require large-scale labeled datasets, substantial computational resources, and complex model training procedures, which may limit their applicability in resource-constrained environments. In addition, the performance of advanced models may not always be optimal when applied to relatively small or domain-specific datasets. Traditional machine learning methods remain relevant because they provide lower computational complexity, easier implementation, and greater interpretability while maintaining competitive classification performance. Therefore, investigating the effectiveness of lightweight classification approaches, such as K-Nearest Neighbor (KNN) combined with TF–IDF and Cosine Similarity, remains important for developing practical hate speech detection systems. Previous studies have implemented machine learning and deep learning methods for text classification tasks on social media platforms [3]. Deep learning architectures have demonstrated strong performance but generally require substantial computational resources and large datasets [4]. Traditional machine learning approaches remain relevant because they provide computational efficiency and easier interpretability while maintaining competitive classification performance [5]. One of the classification algorithms frequently utilized in text mining is K-Nearest Neighbor (KNN), a supervised learning algorithm that classifies data instances based on the similarity of neighboring data points [6].

Text classification tasks require textual information to be transformed into numerical representations before classification can be performed. Term Frequency–Inverse Document Frequency (TF–IDF) is a commonly used weighting technique to represent textual features numerically, while Cosine Similarity can be applied to measure similarity between document vectors [7]. Previous studies indicate that integrating TF–IDF weighting and similarity-based classification methods contributes positively to text classification performance, particularly in hate speech identification tasks [8], [9].

This research proposes an automated hate speech detection framework for X social media using the K-Nearest Neighbor algorithm integrated with TF–IDF feature weighting and Cosine Similarity distance measurement. The dataset consists of social media posts categorized into hate speech and non-hate speech classes, which were processed through tokenization, stopword removal, and stemming before classification. Model performance was evaluated using 10-fold cross validation. The scientific contribution of this study lies in providing an empirical evaluation of the effectiveness of

combining KNN, TF–IDF, and Cosine Similarity for hate speech detection on Indonesian social media textual data, particularly under limited dataset conditions. In addition, this research demonstrates how a lightweight and interpretable machine learning approach can achieve competitive classification performance without relying on computationally intensive deep learning architectures. From a practical perspective, the proposed framework can serve as a cost-effective solution for supporting automated content moderation and hate speech monitoring systems on social media platforms. Experimental results demonstrate that the proposed approach achieved an accuracy of 78.22% using Cosine Similarity distance measurement with K=3 parameter settings.

METHOD

2.1 Dataset Collection

This research utilized social media textual data collected from X (*formerly Twitter*) using the Twitter API integrated with Python programming libraries. Social media platforms provide valuable textual datasets for hate speech detection because they contain large-scale public communication patterns and linguistic variations commonly analyzed in Natural Language Processing (NLP) studies [11]. The dataset consisted of 900 social media posts related to public discussions containing potential hate speech expressions. Each collected text was manually annotated and categorized into two classes: Hate Speech (HS) and Non-Hate Speech (NonHS). The annotation process was conducted based on the semantic content of each post. A text was labeled as Hate Speech if it contained expressions intended to insult, attack, provoke, discriminate against, or incite hostility toward an individual or group based on characteristics such as ethnicity, religion, race, nationality, gender, or other social identities. Conversely, texts that did not contain such expressions were categorized as Non-Hate Speech. To ensure labeling consistency, the annotation process followed predefined classification guidelines derived from previous hate speech detection studies and relevant social media content moderation criteria. The resulting labeled dataset was subsequently used for training and evaluating the classification model. The dataset was divided into training and testing data with a ratio of 75:25, resulting in 675 training instances and 225 testing instances.

2.2 Text Preprocessing

Text preprocessing is a crucial stage in text mining to improve data quality and reduce noise that may affect classification performance. Before feature extraction and classification processes, several preprocessing techniques were applied to normalize textual information and generate more representative features[12]. The preprocessing workflow implemented in this research consists of case folding, tokenization, stopword removal, and stemming, as illustrated in Figure 1.

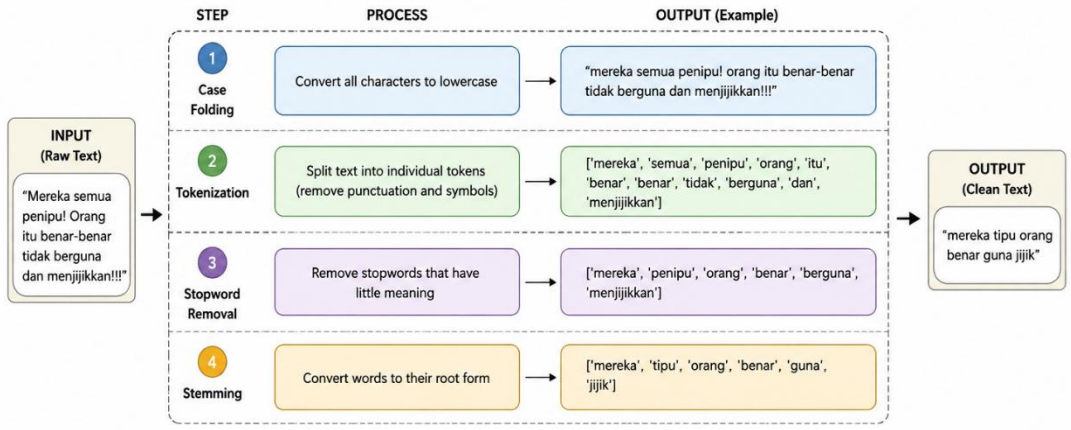

Figure 1. Text Preprocessing Workflow

Case folding converts textual information into lowercase representation to improve feature consistency. Tokenization separates sentence structures into individual lexical units. Stopword removal eliminates common words with limited semantic contribution, while stemming converts inflected words into root forms to reduce vocabulary dimensionality [13]. These preprocessing procedures produce cleaner textual representations and improve classification performance in textual mining applications [14].

2.3 Feature Extraction Using TF–IDF

After preprocessing, textual data were transformed into numerical representations using Term Frequency–Inverse Document Frequency (TF–IDF). TF–IDF is one of the most widely utilized weighting methods in information retrieval and text mining because it measures term importance within documents by considering both term occurrence frequency and document distribution [15]. Term Frequency (TF) calculates the frequency of occurrence of a term within a document, while Inverse Document Frequency (IDF) reduces the influence of highly frequent terms appearing across multiple documents [16]. The TF–IDF weighting formulation is defined as follows :

$$TFIDF(t,d) = TF(t,d) \times IDF(t)$$

where:

$TF(t,d)$= frequency of term t in document d

$IDF(t)$= inverse document frequency

t= term

d= document

The resulting TF–IDF vectors were used as feature inputs for classification.

2.4 Similarity Measurement Using Cosine Similarity

Cosine Similarity was applied to calculate similarity between document vectors generated by TF–IDF weighting. Cosine Similarity measures angular distance between vector representations and is frequently utilized in textual similarity measurement because it effectively captures semantic proximity within vector space representations [17].

$$CosSim(A,B) = \frac{A \cdot B}{\| A \| \times \| B \|}$$

where:

A= vector representation of testing document

B= vector representation of training document

Higher cosine similarity values indicate stronger document similarity.

2.5 K-Nearest Neighbor Classification

K-Nearest Neighbor (KNN) is a supervised learning algorithm that classifies testing instances according to similarity relationships among neighboring training samples [18]. KNN remains widely utilized because of its simplicity, interpretability, and competitive performance in classification problems involving textual data [19]. In this research, KNN was integrated with TF–IDF feature weighting and Cosine Similarity distance measurement to classify social media textual content into hate speech and non-hate speech categories. The classification workflow consists of feature representation, similarity calculation, ranking neighboring documents, selecting K nearest neighbors, and assigning class labels using majority voting mechanisms. The overall classification workflow consists of feature representation, similarity calculation, neighbor selection, and majority voting processes. The complete KNN classification procedure implemented in this research is illustrated in Figure 2.

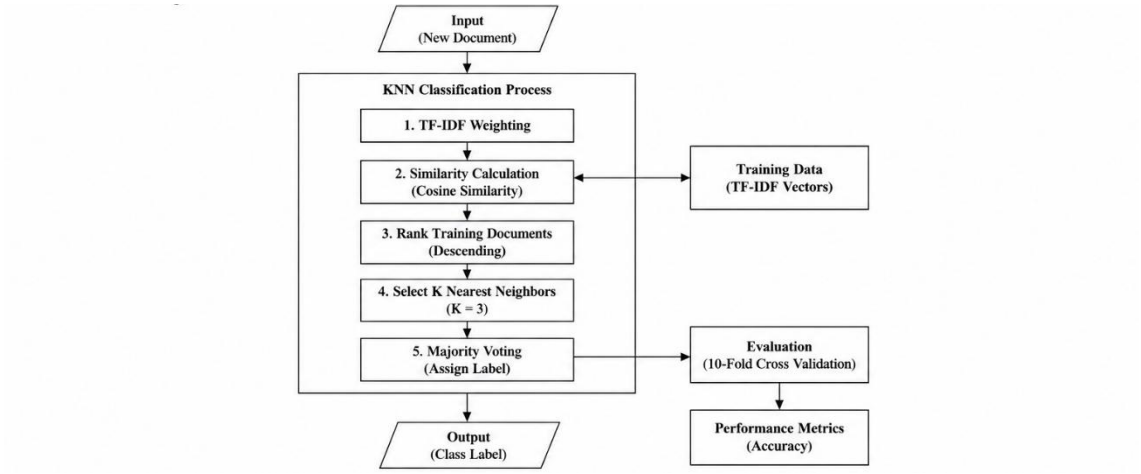

Figure 2. KNN Classification Procedure

Figure 2 illustrates the classification workflow used in this research. Initially, textual documents are transformed into numerical vector representations using TF–IDF weighting. Subsequently, Cosine Similarity is calculated between testing documents and training documents to determine similarity values. The training documents are then ranked based on similarity scores in descending order. After ranking, the K nearest neighboring documents are selected according to the predetermined K parameter value. Finally, class labels are assigned using a majority voting mechanism among selected neighbors. In this research, model performance evaluation was conducted using 10-fold cross validation, where experimental results showed that the optimal performance was achieved at K=3..

2.6 Performance Evaluation

Model evaluation was conducted using a combination of train-test split and 10-fold cross-validation strategies. Initially, the dataset consisting of 900 social media posts was divided into training and testing sets using a 75:25 ratio, resulting in 675 training instances and 225 testing instances. The training dataset was subsequently evaluated using 10-fold cross validation, where the data were partitioned into ten equal subsets. In each iteration, nine subsets were used for training and one subset was used for validation, and the process was repeated ten times so that each subset served as a validation set once. The average performance obtained from the ten iterations was used to determine the optimal K parameter for the K-Nearest Neighbor classifier. After the optimal parameter configuration was identified, the final model was evaluated using the independent testing dataset consisting of 225 unseen instances.. Cross validation is commonly utilized in machine learning studies to estimate model generalization capability and reduce performance bias caused by specific dataset partitioning [20]. Classification performance evaluation utilized Accuracy, Recall, Precision, and F1-Score metrics. Accuracy measures overall classification correctness, Recall evaluates the capability to identify positive instances, Precision measures prediction reliability, and F1-Score balances Precision and Recall performance [21]. The dataset was divided into ten subsets, where one subset was used for testing and the remaining subsets were utilized for training iteratively.

Classification performance was measured using accuracy:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100\%$$

where:
TP = True Positive
TN = True Negative
FP = False Positive
FN = False Negative

Experimental evaluation demonstrated that the proposed approach achieved an accuracy value of 78.22% using Cosine Similarity distance measurement and K=3 parameter settings.

RESULTS AND DISCUSSION

3.1 Dataset Distribution

The dataset used in this research consisted of 900 social media posts collected from X (*formerly Twitter*) and manually categorized into hate speech and non-hate speech classes. The experimental dataset was divided into training and testing sets using a 75:25 ratio, resulting in 675 training data and 225 testing data. The training data were used to build the classification model, while testing data were utilized to evaluate model performance. Table 1 presents dataset distribution used in this research.

Table 1. Dataset Distribution

Dataset Type	Number of Data
Training Data	675
Testing Data	225
Total Data	900

The dataset division aims to ensure that the classification model can learn patterns from training instances while maintaining independent evaluation using unseen testing data.

3.2 Text Preprocessing Result

Text preprocessing was conducted to improve textual quality before the feature extraction and classification processes. Raw textual data collected from X (*formerly Twitter*) often contain various forms of noise, including uppercase and lowercase inconsistencies, punctuation marks, special characters, irrelevant words, and vocabulary variations that may affect classification performance. Therefore, preprocessing procedures were applied to standardize textual information and generate cleaner representations for subsequent analysis.

The preprocessing stage consisted of case folding, tokenization, stopword removal, and stemming. Case folding transformed all characters into lowercase form to ensure consistency in textual representation. Tokenization separated sentence structures into individual tokens by removing punctuation symbols and unnecessary characters. Stopword removal eliminated commonly occurring words that contribute limited semantic information and may introduce noise into the classification process. Finally, stemming converted words into their root forms to reduce vocabulary variation and improve term consistency across documents.

Table 2 presents an example of the preprocessing output obtained from each preprocessing stage. The example demonstrates how raw textual content is progressively transformed into cleaner and more representative textual features. The preprocessing pipeline reduced irrelevant information while preserving meaningful contextual information required for hate speech classification.

Table 2. Example of Text Preprocessing Result

Process	Output
Raw Text	"Mereka semua penipu! Orang itu benar-benar tidak berguna."
Case Folding	"mereka semua penipu orang itu benar benar tidak berguna"
Tokenization	[mereka, semua, penipu, orang, benar, benar, tidak, berguna]
Stopword Removal	[penipu, orang, benar, berguna]
Stemming	[tipu, orang, benar, guna]

The preprocessing procedures contributed to improving textual quality before feature extraction using TF–IDF weighting. Cleaner textual representations facilitate more effective similarity measurement during Cosine Similarity computation and improve classification performance in the K-Nearest Neighbor algorithm. Consequently, preprocessing plays a critical role in reducing noise and enhancing the effectiveness of hate speech detection in social media textual data.

3.3 Classification Performance

Performance model evaluation was conducted using 10-fold cross validation to assess the effectiveness of the proposed hate speech detection model. The classification process utilized K-Nearest Neighbor (KNN) integrated with TF–IDF weighting and Cosine Similarity distance measurement. Since the K parameter directly influences neighbor selection during classification, different K values were evaluated to identify the optimal parameter configuration.

The K parameter determines how many neighboring instances contribute to assigning the class label of testing data. Smaller K values generally provide more specific local decision boundaries but may increase sensitivity to noise, while larger K values tend to produce smoother decision boundaries but may reduce classification specificity. Therefore, selecting an appropriate K value becomes an important factor in optimizing model performance.

Table 3 presents classification results obtained from different K parameter configurations using Cosine Similarity distance measurement.

Table 3. Example of Text Preprocessing Result

Parameter K	Accuracy	Recall	Precision	F1-Score
K = 1	76.23%	77.87%	77.75%	76.34%
K = 3	78.22%	78.76%	78.07%	78.41%
K = 10	69.77%	57.52%	76.47%	65.65%
K = 20	60.44%	37.16%	70.00%	48.55%
K = 50	59.11%	32.47%	69.81%	44.57%

The experimental results demonstrate that parameter selection significantly influences K-Nearest Neighbor classification performance. Based on Table 3, different K parameter values produced varying classification outcomes in terms of accuracy, recall, precision, and F1-score. The highest performance was achieved when K=3, yielding an accuracy of 78.22%, recall of 78.76%, precision of 78.07%, and F1-score of 78.41%.

From a theoretical perspective, the value of K determines the balance between local sensitivity and generalization capability in KNN classification. A relatively small K value allows the classifier to focus on the most similar neighboring instances, thereby preserving local data characteristics. In hate speech detection tasks, textual patterns often exhibit high variability and context dependency, making local neighborhood information particularly important. When K=3, the classifier can effectively capture the similarity patterns of documents represented by TF–IDF vectors while reducing the influence of noisy or irrelevant neighboring samples.

In contrast, larger K values such as K=10, K=20, and K=50 introduce a broader set of neighboring instances into the majority voting process. Although larger K values generally reduce sensitivity to noise, they may also include training instances that are less semantically similar to the testing document. In high-dimensional text classification problems, where document vectors are often sparse, excessive neighborhood sizes can blur class boundaries and reduce the discriminative capability of the classifier. Consequently, the model becomes less sensitive to distinctive hate speech patterns, leading to lower recall and overall classification performance.

These findings suggest that K=3 provides an appropriate balance between preserving local textual similarity and maintaining classification stability. Therefore, the selected parameter is better suited for the characteristics of the dataset used in this study, where hate speech and non-hate speech documents exhibit nuanced linguistic differences that are more effectively captured through smaller neighborhood structures.

3.4 Limitations and Future Research

Despite achieving promising classification performance, this study has several limitations. First, the dataset consists of only 900 social media posts, which may not fully represent the diversity of hate speech expressions found on X and other social media platforms. Second, the classification framework relies on TF–IDF feature representation, which primarily captures term frequency information and may not adequately represent contextual or semantic relationships between words. Third, the study focuses exclusively on binary classification (hate speech and non-hate speech), whereas real-world online content often contains more complex categories such as offensive language, cyberbullying, sarcasm, and implicit hate speech.

Future research may address these limitations by utilizing larger and more diverse datasets collected from multiple social media platforms. In addition, advanced text representation techniques such as word embeddings, contextual embeddings, and transformer-based language models may be explored to better capture semantic information within textual data. Further studies may also investigate multiclass hate speech classification, ensemble learning approaches, and hybrid machine learning frameworks to improve classification performance and enhance the applicability of automated hate speech detection systems in real-world environments.

CONCLUSION

This research proposed a hate speech detection approach on X (*formerly Twitter*) social media using the K-Nearest Neighbor (KNN) algorithm integrated with TF–IDF feature weighting and Cosine Similarity distance measurement. The classification process involved text preprocessing stages consisting of case folding, tokenization, stopword removal, and stemming before feature extraction and classification. Experimental evaluation was conducted using 900 social media text data and assessed using 10-fold cross validation with multiple K parameter configurations. The experimental results indicate that the K parameter significantly affects classification performance, where K=3 achieved the best performance with 78.22% accuracy, 78.76% recall, 78.07% precision, and 78.41% F1-score.

The findings demonstrate that integrating TF–IDF weighting, Cosine Similarity, and K-Nearest Neighbor classification provides an effective approach for hate speech detection in social media textual data while maintaining computational efficiency and model interpretability. The proposed framework offers a practical solution for organizations, researchers, and social media platforms seeking to implement automated hate speech monitoring systems without requiring extensive computational resources or complex model architectures. By utilizing a lightweight machine learning approach, the framework can support content moderation processes, assist in identifying harmful online interactions, and contribute to creating safer digital communication environments. These results indicate that traditional machine learning methods remain a viable alternative for hate speech detection, particularly in scenarios with limited datasets and computational constraints.

REFERENCES

[1] M. S. Jahan and M. Oussalah, "A systematic review of hate speech automatic detection using natural language processing," arXiv preprint arXiv:2106.00742, 2021.

[2] R. Cao, R. K. W. Lee, and T. A. Hoang, "DeepHate: Hate speech detection via multi-faceted text representations," arXiv preprint arXiv:2103.11799, 2021.

[3] E. Utami, Rini, A. F. Iskandar, and S. Raharjo, "Multi-label classification of Indonesian hate speech detection using one-vs-all method," in Proc. International Conference on Information Technology Systems and Innovation (ICITSI), 2021, pp. 78–82.

[4] S. Akuma, T. Lubem, and I. T. Adom, "Comparing bag of words and TF-IDF with different models for hate speech detection from live tweets," Int. J. Inf. Technol., vol. 14, no. 7, pp. 3629–3635, 2022.

[5] M. Subramanian, V. E. Sathiskumar, G. Deepalakshmi, J. Cho, and G. Manikandan, "A survey on hate speech detection and sentiment analysis using machine learning and deep learning models," Egyptian Informatics Journal, vol. 24, 2023.
[6] R. Raut and F. Spezzano, "Enhancing hate speech detection with user characteristics," Int. J. Data Sci. Anal., vol. 16, pp. 131–145, 2023.
[7] A. Toktarova, D. Syrlybay, B. Myrzakhmetova, G. Anuarbekova, G. Rakhimbayeva, and B. Zhylanbayeva, "Hate speech detection in social networks using machine learning and deep learning methods," Int. J. Adv. Comput. Sci. Appl., vol. 14, no. 5, pp. 396–406, 2023.
[8] K. Hadi and E. Utami, "Analysis of K-NN with the integration of bag of words, TF-IDF, and n-grams for hate speech classification on Twitter," JUITA: Jurnal Informatika, vol. 12, no. 2, pp. 289–298, 2024.
[9] X. Yang, "Diagnosing hate speech classification: Where do humans and machines disagree, and why?," arXiv preprint arXiv:2410.10153, 2024.
[10] S. A. Zikrina and Fitriyani, "Advancing hate speech detection in Indonesian language using graph neural networks and TF-IDF," Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), vol. 9, no. 1, pp. 137–145, 2025.
[11] H. Al-Azani and E. El-Alfy, "Using Word Embedding and Ensemble Learning for Highly Imbalanced Data Sentiment Analysis in Short Arabic Text," Procedia Computer Science, vol. 109, pp. 359–366, 2021.
[12] S. Vijayarani, M. J. Ilamathi, and M. Nithya, "Preprocessing Techniques for Text Mining: An Overview," Int. J. Comput. Sci. Commun. Networks, 2021.
[13] C. D. Manning, P. Raghavan, and H. Schütze, Introduction to Information Retrieval. Cambridge University Press.
[14] J. Kaur and V. Gupta, "Effective Text Preprocessing for Machine Learning Applications," Int. J. Data Mining Knowl. Manag. Process., 2022.
[15] G. Salton and C. Buckley, "Term Weighting Approaches in Automatic Text Retrieval," Inf. Process. Manage.
[16] H. Ramos, "Using TF-IDF to Determine Word Relevance in Document Queries," Proc. First Instructional Conference on Machine Learning, 2021.
[17] A. Huang, "Similarity Measures for Text Document Clustering," Proc. NZCSRSC, 2021.
[18] T. Cover and P. Hart, "Nearest Neighbor Pattern Classification," IEEE Trans. Information Theory.
[19] S. Zhang, X. Li, M. Zong, X. Zhu, and R. Wang, "Efficient KNN Classification for Big Data," IEEE Trans. Systems, Man, and Cybernetics, 2022.
[20] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning.
[21] D. Powers, "Evaluation Metrics in Machine Learning Classification Problems," J. Machine Learning Technologies.