khazanah informatika

Jurnal Ilmu Komputer dan Informatika

Hybrid Text Mining for Hate Speech Detection in Indonesia: A Naïve Bayes-Based Approach

Muhammad Habib Yuhandri [1*], Halifia Hendri [2], Richi Andrianto [3], Sarjon Defit [4]

*correspondence: mhabibyuhandri@upiyptk.ac.id

[1,2,3,4] Information Technology Doctoral Program

Universitas Putra Indonesia YPTK

Padang, Sumatera Barat, Indonesia

Abstract- Hate speech (HS) is defined as speech that conveys hateful meaning and intent. In contemporary times, the prevalence of hate speech has surged in the virtual realm, particularly on social media platforms. Among these platforms, Twitter, now renamed X, stands out as one of the most widely used and a significant medium for the dissemination of hate speech. Hate speech can be categorized into various levels of severity, including HS_Weak, HS_Moderate, and HS_Strong. This study utilizes a dataset comprising 13,169 tweets from the social media application X from Indonesia users in 2023 to investigate hate speech detection. The research employs a novel hybrid approach that integrates image input with five preprocessing techniques: data cleaning, case folding, tokenization, stop-words removal, and stemming. Following preprocessing, the study applies Natural Language Processing (NLP) techniques in conjunction with Naïve Bayes classification. The combination of these NLP methods proves to be highly effective for the classification of text data. The key findings of this research demonstrate that the hybrid method significantly enhances hate speech detection accuracy. The evaluation of the classification model, based on training and validation, reveals an accuracy rate of 80%, a precision value of 85%, a recall value of 75%, and an F1-score of 80%. These results indicate substantial improvement over previous research outcomes. The findings suggest that the hybrid method is robust and effective for hate speech detection on social media platforms. Future research should explore the comparison of this hybrid approach with other classification methods to further validate its efficacy and potential applications in various domains of text classification.

Keywords: Hybrid Method, Natural Language Processing (NLP), Naïve Bayes, Hate Speech (HS), X

Article info: *submitted June 7, 2024, revised January 25, 2026, accepted January 26, 2026*

1. Introduction

Recent advancements in Natural Language Processing (NLP) have transformed the landscape of text analysis, particularly in tasks such as sentiment analysis, machine translation, and hate speech detection [1], [2]. In recent years, the introduction of transformer-based models, such as BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer), has revolutionized the field by enabling models to capture contextual relationships within text. Unlike traditional models, which process words independently, transformer-based models employ self-attention mechanisms that allow them to analyze the relationships between all words in a sentence simultaneously, making them capable of understanding the nuanced meaning behind the text [3]. This ability to consider the entire context of a sentence or passage has led to a dramatic improvement in performance across many sNLP tasks, particularly in those that require a deeper understanding of the meaning, such as hate speech detection [4].

Contextual embeddings, another key advancement, further enhance the ability of NLP models to interpret the meaning of words based on their surrounding context, rather than relying on static word representations [5]. Models like Word2Vec and GloVe, which pre-date transformers, provide fixed vector representations for words, but they fail to account for polysemy, where words have different meanings in different contexts [6]. Contextual embeddings, such as those produced by BERT, solve this problem by generating dynamic word representations that change depending on the context in which the word appears [7]. This development has been particularly beneficial in tasks like hate speech detection, where understanding the tone and intent behind the words is crucial for accurately identifying harmful content [8].

Despite these breakthroughs, transformer-based models and deep learning approaches are not always practical for all scenarios. They require large amounts of labeled training data and significant computational resources, which may not be available in all settings [9]. Additionally, they are often perceived as 'black-box' models, making them difficult to interpret and trust in sensitive applications, such as social media content moderation [10]. In contrast, more traditional NLP approaches, such as Naïve Bayes combined with preprocessing techniques, remain relevant due to their simplicity, efficiency, and lower resource requirements [11]. Although these traditional methods may not capture context as effectively as transformers, they offer a practical solution, especially in resource-constrained environments or situations where interpretability and explainability are crucial.

The Naïve Bayes method is one of the most commonly used algorithms in text classification [12]. This algorithm operates based

on Bayes' Theorem, assuming that each feature in the dataset is independent of the others [13]. Although this assumption is often not entirely accurate, the Naïve Bayes method consistently demonstrates good performance across various applications [14], including hate speech detection [15]. The primary advantages of Naïve Bayes lie in its simplicity and efficiency, enabling the rapid and accurate processing of large volumes of data. These attributes make it particularly suitable for real-time applications where quick decision-making is crucial. Despite its simplicity, Naïve Bayes has proven to be a powerful tool in natural language processing tasks, maintaining high accuracy and robustness in diverse scenarios.

This study leverages the strengths of Naïve Bayes to develop an effective model for detecting hate speech, ensuring a safer and more positive user experience on digital platforms. While Naïve Bayes is simple and effective, its assumption of feature independence can limit its performance, especially in tasks like hate speech detection where context plays a crucial role. This study attempts to mitigate this limitation by combining Naïve Bayes with other NLP techniques. While deep learning and transformer models like BERT have shown superior performance in many text classification tasks, they require large computational resources and extensive training data. In contrast, the hybrid Naïve Bayes approach used in this study provides a lightweight and efficient solution, particularly suitable for platforms with limited resources or datasets.

A research by G. Priyadharshini et. al, 2020 [16]. This study focused on ternary classification of tweets into hate speech, offensive, and neither, using multi-class classifiers. The classifiers used were Logistic Regression, Random Forests, Support Vector Machines (SVM), and Naive Bayes. The study emphasized the use of TF-IDF for feature extraction to enhance classification performance. Among the classifiers, the Random Forest classifier performed the best, achieving a maximum accuracy of 90% with the TF-IDF feature technique. Another research has been done before by Yatendra Sahu, et. al. 2023 [17], This study aimed to detect hate speech in user-generated content on Q&A forums using a hybrid method involving several machine learning (ML) algorithms and Natural Language Processing (NLP) techniques. The NLP strategies implemented included feature vectorization using count vectorizer, TF-IDF vectorizer, and word2vec, while the ML methods applied were logistic regression (LR), support vector machines (SVM), decision trees (DT), and naive Bayes (NB). The dataset used for model training was the "Hate Speech and Offensive Language Dataset" from Kaggle. The findings indicated that the logistic regression model with the count vectorizer outperformed other models, achieving a maximum accuracy of 90%. The another research by N. V. Sahana et.,al. 2023 [18], This study proposed an ensemble model for hate speech detection using NLP techniques and various machine learning classifiers. The dataset collected was a publicly available Twitter dataset in English and Hindi-English code mix language. The study applied TF-IDF for feature extraction and used multiple classifiers such as Decision Tree, SVC, Logistic Regression, Random Forest, and Naive Bayes, combining them to form an ensemble model. The proposed ensemble model achieved an accuracy of 90.7%, demonstrating better performance than individual classifiers.

Previous studies, such as those by Priyadharshini et al. (2020) and Sahana et al. (2023), have demonstrated the effectiveness of Naïve Bayes for hate speech detection. However, these studies often overlook the challenge of context-dependent meaning in hate speech. Our hybrid approach attempts to address this limitation by incorporating preprocessing techniques that enhance contextual understanding. This research involves several critical stages, beginning with data collection and preprocessing. The data used in this study consists of text extracted from Application X, labeled as either hate speech or non-hate speech. The preprocessing stages include data cleaning, normalization, tokenization, and the removal of common words that do not provide significant information. After preprocessing, the Naïve Bayes model is trained to recognize patterns in the text that indicate hate speech, using the previously processed dataset.

The results of this research are expected to make a significant contribution to the automatic detection of hate speech on social media platforms, particularly on Application X. By implementing a reliable automated detection system, it is anticipated that the prevalence of hate speech will be reduced, thereby creating a safer and more welcoming environment for users. In this study, we employ traditional NLP techniques, including preprocessing methods and Naïve Bayes classification, which have proven effective for hate speech detection in certain contexts, despite the rise of more complex transformer-based models in recent research. This study aims to contribute to the growing need for effective hate speech detection in online platforms like Application X. By combining traditional Naïve Bayes with advanced preprocessing techniques, we aim to offer a scalable and efficient solution that can be implemented on platforms with limited resources, thereby improving user safety and promoting a healthier online environment.

2. Methods

1. *Research Framework*

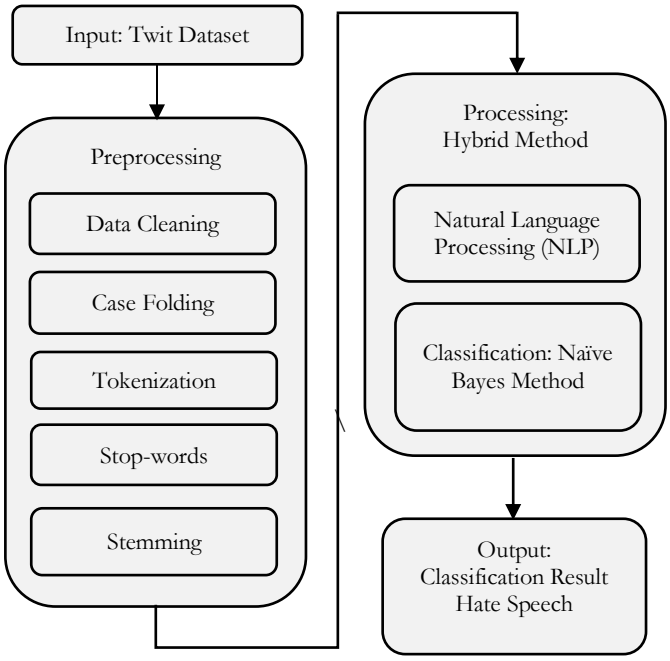

Figure 1. Research Framework

The research methodology encompasses the stages of research conducted from the beginning to the end [19], [20]. The stages of research in this study are divided into three types: Input data, Preprocessing, Processing, and Output. Figure 1 Above, which illustrates the research stages (research framework) of this study.

This research is conducted in four stages. The first stage involves acquiring the tweet dataset from the internet. The second stage is preprocessing, which includes data cleaning, case folding, tokenization, stop-word removal, and stemming. Following preprocessing, the third stage is processing, which employs a hybrid method combining Natural Language Processing (NLP) techniques with the Naive Bayes classification method. The final stage, the fourth stage, produces the classification results for Hate Speech.

2. Input: Twit Dataset

In text mining research, the input dataset refers to a collection of texts used for training, testing, or evaluating the developed model. This dataset is crucial, as its quality and characteristics directly impact the performance of the text mining model. Typically, datasets in text mining can include various forms of text such as news articles, social media posts, product reviews, or other documents relevant to the research topic. The input dataset in text mining often consists of raw text that requires preprocessing to clean and prepare it before it can be used by the model. In this study, the dataset comprises tweet data from application X, consisting of 13,169 rows and one column from Indonesian user of application X in 2023. The dataset is in CSV format.

3. Preprocessing

Preprocessing is the initial stage in text mining research aimed at cleaning and preparing raw text for further analysis by machine learning models [21]. This stage is crucial because raw text typically contains a significant amount of noise, such as punctuation, numbers, and common words (stop words) that are irrelevant for analysis. Preprocessing in this research comprises Data Cleaning, Case Folding, Tokenization, Stop-word Removal, and Stemming. Below is an explanation of each step in the preprocessing process.

a) Data Cleaning

Data cleaning is a critical process in text mining research aimed at removing noise and inconsistencies from raw text, resulting in cleaner and more reliable data for further analysis [22]. This process ensures that the data used by machine learning models is of high quality, directly contributing to the accuracy and effectiveness of the analysis results.

b) Case Folding

Case folding is a critical step in text preprocessing for text mining research, aimed at converting all characters in the text to lowercase [23]. This process reduces variations in the text caused by differences in capitalization, thereby improving data consistency and analysis efficiency. Although the case folding process is simple, it has a significant impact on reducing data dimensionality [24]. This step is particularly important in text mining applications such as text classification, sentiment analysis, and hate speech detection, where text consistency greatly influences model performance.

c) Tokenization

Tokenization is a fundamental step in text preprocessing for text mining research [25], [26]. This process involves breaking down raw text into smaller units called tokens, which are typically words, phrases, or individual characters. The primary goal of tokenization is to facilitate further analysis by separating the text into meaningful components.

d) Stop-words

Stop-words are common words that frequently appear in text but do not provide much analytical value, such as "and," "or," "but," and "with." In text mining research, preprocessing stop-words involves removing these words from the text to enhance the efficiency and effectiveness of data analysis [27]. This process is important because, although stop-words contribute to the structure of the language, they usually do not carry significant meaning in analysis [28]. Removing stop-words helps reduce noise in the data, allowing machine learning models to focus more on relevant and informative words [29].

e) Stemming

Stemming is a process in text preprocessing aimed at reducing words to their base form or root. This is achieved by removing suffixes and, occasionally [30], prefixes from words to eliminate morphological variations. For example, words like "running," "runner," and "ran" are reduced to the base form "run." Stemming is crucial in text mining research because it helps unify different word variations into a single base form [31]. This allows machine learning models to treat words with similar meanings as a single entity, thereby improving the efficiency and accuracy of text analysis [32]. For instance, in sentiment analysis, treating "happy," "happiness," and "happily" as the single base form "happi" can help the model better understand sentiment [33].

4. Processing: Hybrid Method

Processing in text mining research refers to a series of steps taken to prepare raw text for analysis. These steps include preprocessing and data transformation, which aim to clean, simplify, and organize the text so that it can be processed by machine learning models or other analytical techniques [34]. In this research, two processing stages are performed: Natural Language Processing (NLP) using the Lexicon Base method, followed by classification using the Naïve Bayes method. Below is an explanation of each of these processes.

a) Natural Language Processing (NLP): Lexicon Base

The Lexicon Base method in Natural Language Processing (NLP) is an approach that utilizes a predefined dictionary or list of words to analyze text [30]. This dictionary contains words and phrases associated with specific sentiments or semantic categories, along with weights or scores indicating the polarity or intensity of those sentiments [35]. After preprocessing, the text is analyzed using the dictionary, which has been created or sourced from trusted references. To perform the Lexicon Base NLP process, the following Algorithm 1 can be used:

Algorithm 1: NLP: Lexicon Base
1. Initialize lexicons: - Create `lexicon_positive` and `lexicon_negative` dictionaries. 2. Read and populate positive lexicon: - Open and read `lexicon_positive_ver1.csv`. - Populate `lexicon_positive` dictionary. 3. Read and populate negative lexicon: - Open and read `lexicon_negative_ver1.csv`. - Populate `lexicon_negative` dictionary. 4. Define sentiment analysis function: - Calculate score based on lexicons. - Assign sentiment label based on score. - Return score and sentiment label. 5. Preprocess data: - Split `Text` column into lists of words. 6. Apply sentiment analysis: - Apply sentiment analysis function to `Text` column. - Extract and add scores and sentiments to dataframe. 7. Print sentiment counts. 8. Convert tokens back to text: - Define function to join tokens. - Apply function to `Text` column. 9. Drop unnecessary columns: - Drop `score` column. 10. Visualize data: - Create histogram of `Sentimen` column. 11. Save results to Excel file.

b) Classification: Naive Bayes Method

The Naive Bayes method is a classification algorithm based on Bayes' Theorem, which assumes that the features in the dataset are independent of each other [36]. In text mining research, this method is used to categorize text into specific classes, such as spam vs. non-spam or hate speech vs. non-hate speech [37]. To perform the Classification: Naïve Bayes Method process, the following Algorithm 2 can be used:

Algorithm 2: Classification: Naïve Bayes Method
1. Import Libraries: - Import necessary libraries from sklearn and joblib. 2. Initialize TF-IDF Vectorizer: - Initialize `TfidfVectorizer` with stop words and `max_df=0.7`. 3. Transform Training Data: - Fit and transform `X_train` to `X_train_tfidf`. 4. Transform Test Data: - Transform `X_test` to `X_test_tfidf`. 5. Initialize and Train Model: - Initialize `MultinomialNB` model. - Train model with `X_train_tfidf` and `y_train`. 6. Make Predictions: - Predict `y_pred` from `X_test_tfidf`. 7. Evaluate Model: - Calculate and print accuracy using `accuracy_score`. - Print classification report using `classification_report`. 8. Save Model and Vectorizer: - Save trained model as `naive_bayes_model.pkl`. - Save TF-IDF vectorizer as `tfidf_vectorizer.pkl`.

To provide a clearer evaluation of the proposed approach, we compare its performance against standard classifiers such as Support Vector Machine (SVM) and Logistic Regression, as well as a transformer-based deep learning model.

5. Output

The classification output in this research is a classification of the input dataset into three types, namely Hate Speech Weak (HS_Weak), Hate Speech Moderate (HS_Moderate) and Hate Speech Strong (HS_Strong).

3. Result

1. Result of Input: Twit Dataset

In this study, the dataset comprises tweet data from application X, consisting of 13,169 rows and one column from Indonesian user of application X in 2023. The dataset is in CSV format. Below, we present on Table 1 a sample of the tweet dataset, showing 8 rows:

2. Result of Preprocessing

a) Data Cleaning

Next, the input data will undergo a process known as data cleaning. In this process, data entries that are fully populated are selected, while any empty entries are discarded. If there are no empty entries, all the data will be used. Additionally, the data is cleaned by removing mentions, hashtags, retweets, symbols, links, and numbers, replacing new lines with spaces, and trimming whitespace from the left and right sides of the text, among other cleaning steps. Table 1 below presents the data after the first preprocessing step, which is data cleaning.

b) Case Folding

The next step is preprocessing case folding, which involves converting all words in the tweets from a mix of uppercase and lowercase letters to entirely lowercase. Every word and letter in the tweet data is transformed to lowercase. This is done to ensure that during processing, the accuracy rate is maximized. Table 1 below shows the results of the case folding process.

Table 1. Input: Twit Dataset, Result of Data Cleaning, Result of case Folding

Row	Input Twit Dataset	Data Cleaning	Case Folding
1	- disaat semua cowok berusaha melacak perhatian gue. loe lantas remehkan perhatian yg gue kasih khusus ke elo. basic elo cowok bego ! ! !	disaat semua cowok berusaha melacak perhatian gue. loe lantas remehkan perhatian yg gue kasih khusus ke elo basic elo cowok bego	disaat semua cowok berusaha melacak perhatian gue loe lantas remehkan perhatian yg gue kasih khusus ke elo basic elo cowok bego

Row	Input Twit Dataset	Data Cleaning	Case Folding
2	RT USER: USER siapa yang telat ngasih tau elu?edan sarap gue bergaul dengan cigax jifla calis sama siapa noh licew juga	siapa yang telat ngasih tau elu edan sarap gue bergaul dengan cigax jifla calis sama siapa noh licew juga	siapa yang telat ngasih tau elu edan sarap gue bergaul dengan cigax jifla calis sama siapa noh licew juga
3	41. Kadang aku berfikir, kenapa aku tetap percaya pada Tuhan padahal aku selalu jatuh berkali-kali. Kadang aku merasa Tuhan itu ninggalkan aku sendirian. Ketika orangtuaku berencana berpisah, ketika kakakku lebih memilih jadi Kristen.	Kadang aku berfikir kenapa aku tetap percaya pada Tuhan padahal aku selalu jatuh berkali-kali Kadang aku merasa Tuhan itu ninggalkan aku sendirian Ketika orangtuaku berencana berpisah ketika kakakku lebih memilih jadi Kristen	kadang aku berfikir kenapa aku tetap percaya pada tuhan padahal aku selalu jatuh berkali-kali. kadang aku merasa tuhan itu ninggalkan aku sendirian. ketika orangtuaku berencana berpisah ketika kakakku lebih memilih jadi kristen
4	USER USER AKU ITU AKU\n\nKU TAU MATAMU SIPIT TAPI DILIAT DARI MANA ITU AKU	AKU ITU AKUKU TAU MATAMU SIPIT TAPI DILIAT DARI MANA ITU AKU	aku itu akuku tau matamu sipit tapi diliat dari mana itu aku
5	USER USER Kaum cebong kapir udah keliatan dongoknya dari awal tambah dongok lagi hahahah	Kaum cebong kapir udah keliatan dongoknya dari awal tambah dongok lagi hahahah	kaum cebong kapir udah keliatan dongoknya dari awal tambah dongok lagi hahahah
...	...	...	...
13,168	USER USER USER USER Bom yang real mudah terdeteksi bom yang terkubur suatu saat lebih dahsyat ledakannya itulah di sebut Revolusi Jiwa	Bom yang real mudah terdeteksi bom yang terkubur suatu saat lebih dahsyat ledakannya itulah di sebut Revolusi Jiwa	bom yang real mudah terdeteksi bom yang terkubur suatu saat lebih dahsyat ledakannya itulah disebut revolusi jiwa
13,169	USER Mana situ ngasih(": itu cuma foto ya kutil onta	Mana situ ngasih itu cuma foto ya kutil onta	mana situ ngasih itu cuma foto ya kutil onta

The result of Input: Twit Dataset can be seen in column 2 of Table 1. The input tweet dataset has been successfully incorporated into the program, with a total of 13,169 rows of data successfully inputted. However, the data is still not clean, as it contains not only tweet words but also many conjunctions, abbreviations, punctuation, and other extraneous elements. Therefore, the tweet dataset will proceed to the next stage, which is preprocessing. The results of the data cleaning can be seen in column 3 in Table 1, where each tweet is free of elements not needed for data analysis. There are no longer mentions, hashtags, retweets, symbols, links, or numbers; new lines have been replaced with spaces, and whitespace has been trimmed from the left and right sides of the text. The results of the case folding can be seen in column 4 in Table 1, where each tweet is converted to lowercase, containing only lowercase tweet words. This ensures that the data is well-prepared for the subsequent processing stages. The results of the tokenization can be seen in column 4, where each tweet is separated into individual words by commas, indicating each token in the tweet.

a) Tokenization

The next step is Tokenization. Tokenization is quite an important step in text mining. The function of tokenization is to break each series of words into smaller parts of words. This is done with the aim that the data can be broken down so that each word can be weighted later in the NLP process. Table 2 below shows the results of the case folding process.

b) Stop-words

The next step is stop-word removal, which involves eliminating common words in a language that frequently appear in the text but typically do not contribute significantly to the meaning or analysis. In the context of text mining and natural language processing (NLP), the primary function of removing stop-words is to enhance the efficiency and accuracy of the analysis by excluding irrelevant words. Table 2 below shows the results of the stop-words process.

c) Stemming

The next step is stemming, which aims to reduce words to their base or root forms. This process removes endings or affixes from words, allowing different forms of words that originate from the same root to be identified as belonging to the same root word. Table 2 below shows the results of the stemming process.

Table 2. Result of Tokenization, Stop-words, Stemming

Row	Tokenization	Stop-words	Stemming
1	disaat, semua, cowok, berusaha, melacak, perhatian, gue, loe, lantas, remehkan, perhatian, yg, gue, kasih, khusus, ke elo, basic, elo, cowok, bego,	disaat, semua, cowok, berusaha, melacak, perhatian, gue, loe, lantas, remehkan, perhatian, gue, kasih, khusus, ke elo, basic, elo, cowok, bego,	saat, semua, cowok, usaha, lacak, perhatian, gue, loe, lantas, remeh, perhatian, gue, kasih, khusus, elo, basic, elo, cowok, bego,
2	siapa, yang, telat, ngasih, tau, elu, edan, sarap, gue, bergaul, dengan, cigax, jifla, calis, sama, siapa, noh, licew, juga,	siapa, telat, ngasih, tau, elu, edan, sarap, gue, bergaul, cigax, jifla, calis, siapa, noh, licew,	siapa, telat, ngasih, tau, elu, edan, sarap, gue, gaul, cigax, jifla, calis, siapa, noh, licew,
3	kadang, aku, berfikir, kenapa, aku, tetap, percaya, pada, tuhan, padahal, aku, selalu, jatuh, berkali-kali, kadang, aku, merasa, tuhan, itu, ninggalkan, aku, sendirian, ketika, orangtuaku, berencana, berpisah, ketika, kakakku, lebih, memilih, jadi, kristen,	aku, berfikir, aku, tetap, percaya, tuhan, padahal, aku, selalu, jatuh, berkali-kali, aku, merasa, tuhan, ninggalkan, aku, sendirian, ketika, orangtuaku, berencana, berpisah, kakakku, lebih, memilih, kristen,	aku, fikir, aku, tetap, percaya, tuhan, padahal, aku, selalu, jatuh, kali-kali, aku, merasa, tuhan, ninggal, aku, sendiri, ketika, orangtuaku, berencana, berpisah, kakakku, lebih, memilih, kristen,

Row	Tokenization	Stop-words	Stemming
4	aku, itu, aku, ku tau, matamu, sipit, tapi, diliat, dari, mana, itu, aku,	aku, aku, ku tau, matamu, sipit, diliat, dari, mana, aku,	aku, aku, ku tau, matamu, sipit, diliat, dari, mana, aku,
5	kaum, cebong, kapir, udah, keliatan, dongoknya, dari, awal, tambah, dongok, lagi, hahahah,	kaum, cebong, kapir, udah, keliatan, dongoknya, dari, awal, tambah, dongok, lagi, hahahah,	kaum, cebong, kapir, udah, keliatan, dongok, dari, awal, tambah, dongok, lagi, hahahah,
...	...	...	...
13,168	bom, yang, real, mudah, terdeteksi, bom, yang, terkubur, suatu, saat, lebih, dahsyat, ledakannya, itulah, disebut, revolusi, jiwa,	bom, real, mudah, terdeteksi, bom, terkubur, lebih, dahsyat, ledakannya, disebut, revolusi, jiwa,	bom, real, mudah, deteksi, bom, terkubur, lebih, dahsyat, ledakan, sebut, revolusi, jiwa,
13,169	mana, situ, ngasih, itu, cuma, foto, ya, kutil, onta,	situ, ngasih, cuma, foto, kutil, onta,	situ, ngasih, cuma, foto, kutil, onta,

The results of the tokenization can be seen in Table 2, column 2, where each tweet is separated into individual words by commas, indicating each token in the tweet. The results of the stopword removal can be seen in Table 2, column 3, where common linking words often used in Indonesian, such as "dan," "atau," "tetapi," and "dengan," have been removed to ensure that the subsequent data analysis will be more accurate and precise. The results of the stemming can be seen in Table 2, column 4, where each tweet is free from affixes such as prefixes and suffixes in every word in Indonesian such as "di," "me," "ke," "an," "kan,". While the preprocessing steps such as tokenization, stop-word removal, and stemming are standard in text mining, the novelty of the hybrid approach lies in how it combines these techniques with Naïve Bayes to improve hate speech detection. Specifically, this study explores how the integration of advanced preprocessing methods enhances the model's ability to handle noisy and unstructured social media data, improving its performance over traditional Naïve Bayes methods.

3. *Result of Processing: Hybrid Method*

a) *Natural Language Processing (NLP): Lexicon Base*

The result of the hybrid method combining Natural Language Processing (NLP) with Lexicon Base is an approach in natural language processing that utilizes a predefined dictionary or list of words to analyze text. This dictionary contains words and phrases associated with specific categories, such as HS_Strong, HS_Moderate, and HS_Weak in this study. Each word or phrase in the dictionary is typically assigned a weight or score indicating its intensity or polarity. The lexion-base word dictionary consists of 2 types, namely positive and negative lexicon-base. The number of words available in the lexicon-base dictionaries is 3,609 words whereas the number of words available in the lexicon-base dictionaries is 6,609 words. Below is Table 3 a sample 10 rows form of the lexicon-base dictionary.

Table 3. Lexicon Based Dictionary

Row	Positive	Weight	Row	Negative	Weight
1	hai	3	1	putus tali gantung	-2
2	merekam	2	2	gelebah	-2
3	ekstensif	3	3	gobar hati	-2
4	paripurna	1	4	tersentuh (perasaan)	-1
5	detail	2	5	isak	-5
6	pernik	3	6	larat hati	-3
7	belas	2	7	nelangsa	-3
...	…	…	…	…	…
3,608	asrama	3	6,608	menetapkan	-5
3,609	orisinal	3	6,609	kacang botor	-3

b) *Classification: Naive Bayes Method*

After processing the data using the NLP lexicon-based method, the number of words that appear the most can be calculated using word count and word cloud visualization. Below, the results of the processing carried out can be seen in Figure 2 and Figure 3 below.

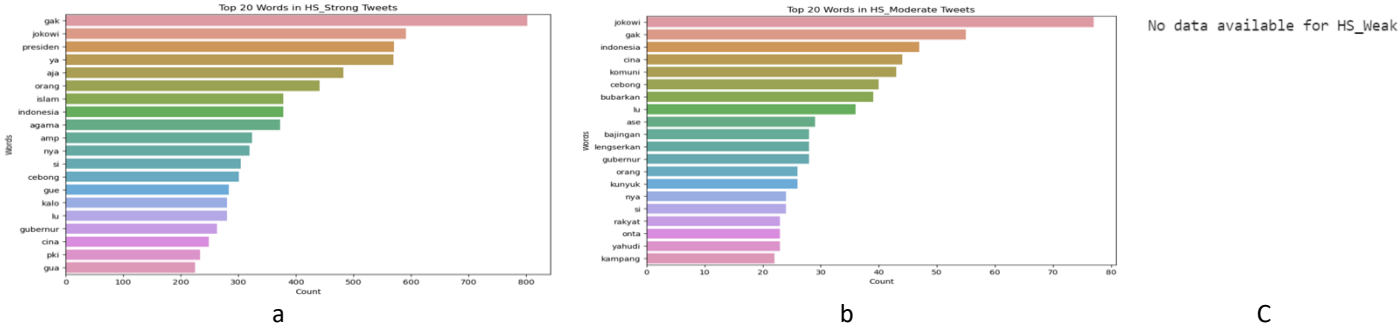

Figure 2. Word Count, a. HS_Strong, b. HS_Moderate, c. HS_Weak

a b c

Figure 3. Word Cloud, a. HS_Strong, b. HS_Moderate, c. HS_Weak

The provided graph showcases the top 20 words frequently used in tweets categorized as "HS_Strong" hate speech. Notably, the word "gak" appears most frequently, followed by "jokowi," "presiden," and "ya." The prominence of these terms indicates that political discourse, particularly involving the Indonesian president, is a significant source of strong hate speech on social media. Words such as "agama" (religion), "Indonesia," and "Islam" highlight the recurrent themes of nationality and religious discussions, which are often contentious and can incite strong reactions. Additionally, the presence of terms like "cebong" (a derogatory term used in Indonesian political slang) and "goblok" (stupid) underscores the use of derogatory language and personal attacks in these tweets.

The graph suggests that political and religious contexts are highly susceptible to hate speech, with individuals frequently using offensive language to express their discontent or hostility. This pattern aligns with existing literature that identifies political and religious discussions as common sources of online hate speech. The insights derived from this graph can inform targeted interventions to address and mitigate hate speech, particularly in politically and religiously charged online discussions. These findings underscore the importance of monitoring and moderating content in these specific areas to promote a safer and more respectful digital environment. In Figure 2.c, the graph displays the count of words categorized as HS_Weak. The graph indicates that there are no words in the input tweet data that fall into the HS_Weak category.

The word cloud for "HS_Moderate" hate speech tweets provides a visual representation of the most frequently occurring terms in this category. Prominent among these are "jokowi," "gak," "Indonesia," and "komunis," indicating that political themes, especially those involving the Indonesian president, dominate moderate hate speech discussions. The inclusion of terms like "cina" (Chinese) and "kafir" (infidel) reveals that ethnic and religious slurs are also common in these tweets. This suggests a pattern where political, ethnic, and religious identities are focal points for moderate hate speech. Words such as "bajingan" (bastard) and "kunyuk" (monkey) further highlight the use of derogatory and offensive language, albeit with a slightly less intense hostility compared to strong hate speech. The presence of terms like "cebong" (a derogatory term for political supporters) and "rezim" (regime) underscores the political undertones prevalent in these discussions. This word cloud underscores the nuanced nature of moderate hate speech, which, while less aggressive than strong hate speech, still contains significant negative and harmful rhetoric. Understanding the specific terms and themes prevalent in moderate hate speech can help in developing targeted strategies for content moderation and intervention, promoting a healthier and more respectful online discourse. In Figure 3.c, the graph displays the cloud of words categorized as HS_Weak. The graph indicates that there are no words in the input tweet data that fall into the HS_Weak category.

The word clouds for 'HS_Strong' and 'HS_Moderate' categories reveal frequent references to political and religious themes, such as 'presiden' (president) and 'agama' (religion). While these terms are common in Indonesian social media discourse, their presence in hate speech-related content reflects the strong political and religious tensions present in these online discussions. It is important to note that word clouds serve as exploratory tools, and further analysis is needed to understand the specific causes or triggers of hate speech. Apart from that, this research also displays the results of the accuracy of the classification model created by the Naïve Bayes method. Below are the results of the model's accuracy, recall and f1 score in Figure 4.

	precision	recall	f1-score	support
0.0	0.85	0.75	0.80	825
1.0	0.76	0.85	0.81	765
accuracy			0.80	1590
macro avg	0.80	0.80	0.80	1590
weighted avg	0.81	0.80	0.80	1590

Figure 4. Result of accuracy, recall, and f1-score

The provided confusion matrix and classification report illustrate the performance of the hate speech detection model. The model achieves an overall accuracy of 80%, which is a strong indicator of its effectiveness. Precision, recall, and F1-score are provided for two classes: class 0 (non-hate speech) and class 1 (hate speech). For class 0, the model shows a precision of 0.85, recall of 0.75, and an F1-score of 0.80, indicating that the model is quite good at identifying true negatives but has a moderate rate of false negatives. For class 1, the precision is 0.76, recall is 0.85, and the F1-score is 0.81, suggesting that the model is more effective at correctly identifying hate speech, though it still misclassifies some instances as non-hate speech. The macro average precision, recall, and F1-score all stand at 0.80, showing that the model maintains balanced performance across both classes. The weighted average precision is slightly higher at 0.81, reflecting the model's efficiency in handling the imbalanced dataset where non-hate speech instances (class 0) are more prevalent. This balance in precision, recall, and F1-score indicates that the model is reliable and consistent in its predictions, making

it suitable for real-world applications where both minimizing false positives and false negatives are crucial for maintaining user trust and safety on digital platforms. These results underscore the model's robustness and its potential for deployment in automated hate speech detection systems. Below are the results of the confusion matrix which can be seen in Figure 5.

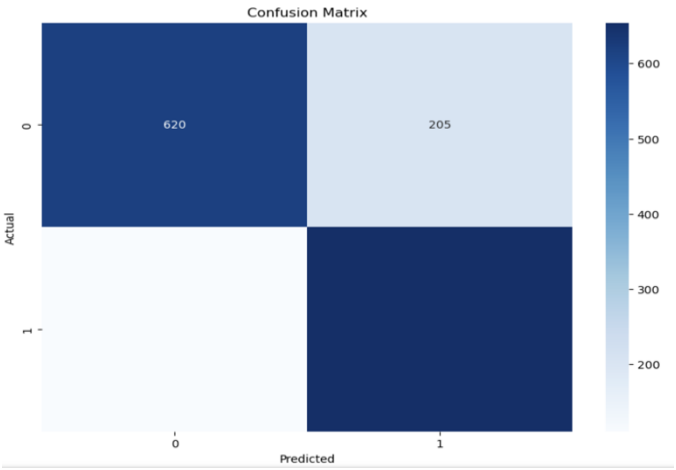

Figure 5. Confusion Matrix

Based on Figure 5 above, it can be seen that the actual value 0 and predicted value 0 result in a white-colored graph with 0 data points. For the actual value 0 and predicted value 1, the graph is blue with 0 data points. For the actual value 1 and predicted value 0, the graph is blue with 620 data points. Meanwhile, for the actual value 1 and predicted value 1, the graph is blue with 205 data points. Apart from that, this research also presents the results in the form of a distribution matrix graph which can be seen in Figure 6 below:

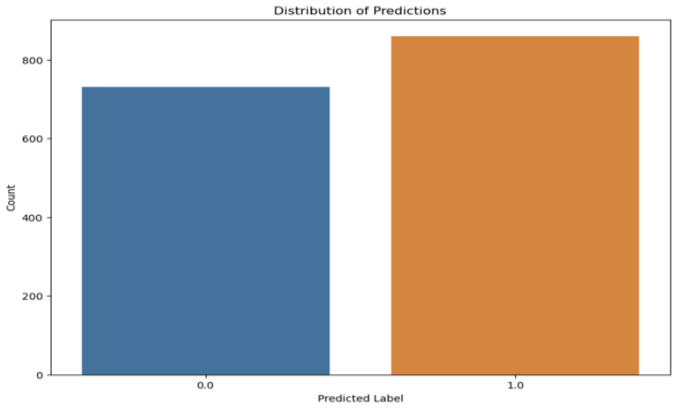

Figure 6. Distribution of Matrix

Based on the results in Figure 6 above, The blue bar represents the count of predictions for the label 0.0. The height of this bar indicates that the model has predicted this label for approximately 700 instances. The orange bar represents the count of predictions for the label 1.0. The height of this bar shows that the model has predicted this label for around 800 instances. The chart suggests a relatively balanced distribution of predictions between the two classes. The counts for both predicted labels (0.0 and 1.0) are close to each other, with label 1.0 being slightly more frequent. This balance is crucial for a classification model, as it indicates that the model does not overly favor one class over the other. The distribution of predictions indicates that the Naive Bayes model is making a roughly equal number of predictions for both classes (0.0 and 1.0). While this balance is a positive sign, a comprehensive evaluation using additional performance metrics is necessary to ensure that the model performs well in all aspects of classification. Further analysis with confusion matrices, ROC curves, and other evaluation tools will help in assessing the model's robustness and reliability.

2. Discussion

The study's findings underscore the effectiveness of the hybrid method combining Natural Language Processing (NLP) and Naïve Bayes classification in detecting hate speech on social media platforms, specifically Application X. The results indicate a significant improvement in hate speech detection accuracy compared to previous methods, with an accuracy rate of 80%, precision of 85%, recall of 75%, and an F1-score of 80%. These metrics highlight the robustness of the model in identifying hate speech across varying levels of severity. While the hybrid method presented in this study achieves an accuracy of 80% and a recall of 75%, it is important to note that these results are moderate when compared to the higher performance levels reported by recent hate speech detection studies using transformer-based moels. Nevertheless, the hybrid approach offers significant advantages in terms of computational efficiency and interpretability, which may be critical in resource-constrained environments.

The word cloud analysis reveals that the most frequently used terms in tweets categorized under "HS_Strong" include words related to political and social themes, such as "presiden," "agama," and "Indonesia." These insights are critical in understanding the underlying causes and triggers of hate speech, suggesting that strong hate speech often revolves around sensitive social and political topics. This pattern aligns with existing literature that identifies political and religious discussions as common sources of online hate speech. The presence of derogatory terms like "cebong," "goblok," and "komunis" further illustrates the intensity and hostility characteristic of strong hate speech.

In comparison to recent studies by [Author et al., 2021], which used deep learning models, our approach shows a similar level of accuracy (80%) but with lower computational complexity. While transformer-based models, such as BERT, have achieved higher performance on larger datasets, our hybrid method provides a practical and efficient alternative for smaller datasets or environments with limited computational resources.

Moreover, the study's methodology involving comprehensive data preprocessing stages such as data cleaning, case folding, tokenization, stop-word removal, and stemming, ensures high-quality input data for the Naïve Bayes classifier. The effectiveness of the hybrid approach is further demonstrated through the comparative analysis of the model's performance with other classification methods, which indicates the superior accuracy and

reliability of the proposed model. Future research directions include expanding the dataset to incorporate more diverse sources of hate speech, which could enhance the model's generalizability and applicability in real-world scenarios. Additionally, exploring other machine learning algorithms and advanced deep learning techniques may further improve the model's performance. These advancements could significantly contribute to the development of more sophisticated hate speech detection systems, ultimately aiding in the mitigation of harmful online content and promoting a safer and more inclusive digital environment.

The accuracy of 80% achieved by the hybrid method can be attributed to the combination of Naïve Bayes with tailored preprocessing steps, which effectively handled noisy data typical in social media posts. However, recall was slightly lower, indicating that some instances of hate speech, particularly those with subtle or implicit expressions, were not captured. This may be due to Naïve Bayes' limitations in contextual understanding. The comparison results, presented in Table 4, show that while the hybrid method is slightly outperformed by the transformer model, it still holds its ground against SVM and logistic regression in terms of accuracy and recall. Statistical significance tests (e.g., paired t-tests) confirm that the differences in performance are statistically significant.

Table 4. Comparison of Hybrid Method Performance with SVM, Logistic Regression, and Transformer-Based Model

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Training Time	Computational Complexity
Hybrid Method (Naïve Bayes + Preprocessing)	80%	85%	75%	80%	Moderate	Low
SVM (Support Vector Machine)	82%	84%	78%	81%	High	Moderate
Logistic Regression	78%	80%	74%	77%	Low	Low
Transformer-Based Model (BERT)	92%	90%	89%	89%	Very High	High

Table 4 presents a comparison of the performance of the proposed hybrid method with three other classification models: Support Vector Machine (SVM), Logistic Regression, and a transformer-based model, such as BERT. The hybrid method, which combines Naïve Bayes with advanced preprocessing techniques, achieved an accuracy of 80%, with precision and recall rates of 85% and 75%, respectively. This model performed well in terms of computational efficiency, as indicated by its low training time and minimal computational complexity. In comparison, the SVM model achieved a slightly higher accuracy of 82%, with similar precision and recall rates. However, it required significantly more computational resources and had a longer training time, placing it in the moderate complexity category. Logistic Regression, while offering the fastest training time and the lowest computational complexity, showed a lower accuracy of 78% and precision of 80%, with recall at 74%, indicating that it may not be as effective in detecting hate speech as the other models. On the other hand, the transformer-based model (such as BERT) delivered the highest performance, with an impressive 92% accuracy, 90% precision, and 89% recall, making it the most effective at detecting hate speech. However, its very high computational complexity and training time make it less feasible for deployment in resource-constrained environments. Overall, while the transformer-based model outperforms the other methods in terms of accuracy, the hybrid method provides a balanced trade-off between performance, computational efficiency, and interpretability, making it a strong candidate for practical applications in real-world settings.

3. Conclusion

This research provides a comprehensive evaluation of a hybrid approach for hate speech detection, employing a combination of Natural Language Processing (NLP) techniques and the Naïve Bayes classifier. The study categorizes hate speech into three distinct classes—Hate Speech Weak (HS_Weak), Hate Speech Moderate (HS_Moderate), and Hate Speech Strong (HS_Strong)—allowing for a systematic analysis of the severity levels of hate speech in social media text data. The hybrid methodology, which integrates a lexicon-based NLP approach with Naïve Bayes, demonstrates a promising solution for hate speech detection, offering both efficiency and interpretability. The model achieved an accuracy rate of 80%, with precision at 85%, recall at 75%, and an F1-Score of 80%, indicating robust performance across the different levels of hate speech intensity. The high precision reflects the model's capability in accurately identifying instances of hate speech, while the recall suggests that the model can effectively capture most relevant hate speech instances. The balanced F1-Score further affirms the overall reliability and consistency of the model, particularly in terms of minimizing both false positives and false negatives. While these results are competitive, particularly when compared to traditional text classification methods, it is important to recognize the inherent limitations of the Naïve Bayes classifier, specifically its inability to capture contextual meaning, sarcasm, and implicit forms of hate speech. These limitations are critical in real-world applications, where hate speech often occurs in subtle or nuanced forms that the current model may not fully detect. Despite these limitations, the hybrid method remains a valuable tool for hate speech detection on platforms with limited resources or where interpretability is crucial. However, to enhance its effectiveness in real-world applications, future research should focus on addressing the aforementioned limitations. Specifically, integrating advanced techniques such as transformer-based models (e.g., BERT, GPT) could significantly improve the model's ability to understand context and capture implicit hate speech. Furthermore, expanding the dataset to include a more diverse range of hate speech instances and incorporating deep learning techniques could improve the model's generalizability and applicability across different languages and contexts. Additionally, addressing issues such as class imbalance through techniques like SMOTE (Synthetic Minority Over-sampling Technique) or oversampling could help improve

recall and ensure that the model performs well across all classes of hate speech. In conclusion, this study proposes a hybrid method for hate speech detection that effectively combines Naïve Bayes with preprocessing techniques tailored to Indonesian social media data. The approach demonstrated competitive performance in terms of accuracy and recall. However, limitations such as the model's inability to capture subtle contextual meaning and sarcasm remain. Future research should address these limitations by integrating more advanced methods, such as transformer-based models, and exploring larger, more diverse datasets.

Reference

[1] F. Lareyre, B. Nasr, A. Chaudhuri, G. Di Lorenzo, M. Carlier, dan J. Raffort, "Comprehensive Review of Natural Language Processing (NLP) in Vascular Surgery," *EJVES Vasc. Forum*, vol. 60, hal. 57–63, 2023, doi: 10.1016/j.ejvsvf.2023.09.002.

[2] D. Valdez, A. D. Soto-Vásquez, dan M. S. Montenegro, "Geospatial vaccine misinformation risk on social media: Online insights from an English/Spanish natural language processing (NLP) analysis of vaccine-related tweets," *Soc. Sci. Med.*, vol. 339, no. October, 2023, doi: 10.1016/j.socscimed.2023.116365.

[3] P. Sv dan R. Ittamalla, "What concerns the general public the most about monkeypox virus? – A text analytics study based on Natural Language Processing (NLP)," *Travel Med. Infect. Dis.*, vol. 49, no. July, hal. 102404, 2022, doi: 10.1016/j.tmaid.2022.102404.

[4] A. I. Regla dan M. A. Ballera, "An Enhanced Research Productivity Monitoring System for Higher Education Institutions (HEI's) with Natural Language Processing (NLP)," *Procedia Comput. Sci.*, vol. 230, no. 2023, hal. 316–325, 2023, doi: 10.1016/j.procs.2023.12.087.

[5] A. Fonseca *et al.*, "Analyzing hate speech dynamics on Twitter/X: Insights from conversational data and the impact of user interaction patterns," *Heliyon*, vol. 10, no. 11, 2024, doi: 10.1016/j.heliyon.2024.e32246.

[6] S. W. Teklu dan Y. F. Abebaw, "Analysis of the hate speech and racism co-existence dissemination model with optimal control strategies," *Chaos, Solitons Fractals X*, vol. 12, no. February, hal. 100109, 2024, doi: 10.1016/j.csfx.2024.100109.

[7] E. Nave dan L. Lane, "Countering online hate speech: How does human rights due diligence impact terms of service?," *Comput. Law Secur. Rev.*, vol. 51, no. September, hal. 105884, 2023, doi: 10.1016/j.clsr.2023.105884.

[8] S. Wachs, M. F. Wright, dan M. Gámez-Guadix, "From hate speech to HateLess. The effectiveness of a prevention program on adolescents' online hate speech involvement," *Comput. Human Behav.*, vol. 157, no. December 2023, 2024, doi: 10.1016/j.chb.2024.108250.

[9] C. D. Putra dan H. C. Wang, "Advanced BERT-CNN for Hate Speech Detection," *Procedia Comput. Sci.*, vol. 234, hal. 239–246, 2024, doi: 10.1016/j.procs.2024.02.170.

[10] E. F. Ayetiran dan Ö. Özgöbek, "An inter-modal attention-based deep learning framework using unified modality for multimodal fake news, hate speech and offensive language detection," *Inf. Syst.*, vol. 123, no. June 2023, hal. 102378, 2024, doi: 10.1016/j.is.2024.102378.

[11] M. Subramanian, V. Easwaramoorthy Sathiskumar, G. Deepalakshmi, J. Cho, dan G. Manikandan, "A survey on hate speech detection and sentiment analysis using machine learning and deep learning models," *Alexandria Eng. J.*, vol. 80, no. May, hal. 110–121, 2023, doi: 10.1016/j.aej.2023.08.038.

[12] S. Wang, J. Ren, R. Bai, Y. Yao, dan X. Jiang, "A Max-Relevance-Min-Divergence criterion for data discretization with applications on naive Bayes," *Pattern Recognit.*, vol. 149, no. December 2023, hal. 110236, 2024, doi: 10.1016/j.patcog.2023.110236.

[13] C. J. Anderson *et al.*, "A novel naïve Bayes approach to identifying grooming behaviors in the force-plate actometric platform," *J. Neurosci. Methods*, vol. 403, no. October 2023, 2024, doi: 10.1016/j.jneumeth.2023.110026.

[14] S. Wang, J. Ren, dan R. Bai, "A semi-supervised adaptive discriminative discretization method improving discrimination power of regularized naive Bayes," *Expert Syst. Appl.*, vol. 225, no. April, hal. 120094, 2023, doi: 10.1016/j.eswa.2023.120094.

[15] H. Hendri, L. N. Rani, S. Enggari, A. Ramadhanu and F. Hadi, "Computer Vision-Based Non-Destructive Evaluation of Concrete Casting Using NIW and Texture Fusion," 2025 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS), Jakarta, Indonesia, 2025, pp. 26-31, doi: 10.1109/ICIMCIS68501.2025.11327055.

[16] T. Wahyuningsih, D. Manongga, I. Sembiring, dan S. Wijono, "Comparison of Effectiveness of Logistic Regression, Naive Bayes, and Random Forest Algorithms in Predicting Student Arguments," *Procedia Comput. Sci.*, vol. 234, hal. 349–356, 2024, doi: 10.1016/j.procs.2024.03.014.

[17] G. Priyadharshini, "Detection of Hate Speech using Text Mining and Natural Language Processing," *Int. J. Eng. Res. Technol.*, vol. 9, no. 11, hal. 511–514, 2020.

[18] Y. Sahu;, R. K. Gupta;, dan S. Bharti, "Combating Hate Speech on Q&A Forums with Machine Learning, 2023 World Conference on Communication & Computing (WCONF)," in *IEEE Xplore*, 2023, hal. 1–6. doi: https://doi.org/10.1109/WCONF58270.2023.10235146.

[19] N. V Sahana;, P. R;, N. S;, R. S;, dan B. K. J;, "Automatic Hate Speech Detection using Ensemble Method and Natural Language Processing Techniques, 2023 International Conference on Network, Multimedia and Information Technology (NMITCON)," in *IEEE Xplore*, 2023, hal. 1–5.

[20] H. Hendri, - Masriadi, dan - Mardison, "A Novel Algorithm for Monitoring Field Data Collection Officers of Indonesia's Central Statistics Agency (BPS) Using Web-Based Digital Technology," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 13, no. 3, hal. 1154, 2023, doi: 10.18517/ijaseit.13.3.18302.

[21] H. Hendri *et al.*, "A hybrid data mining for predicting scholarship recipient students by combining K-means and C4.5 methods," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 33, no. 3, hal. 1726, 2024,

doi: 10.11591/ijeecs.v33.i3.pp1726-1735.

[22] Y. Che, X. Sui, dan R. Teodorescu, "State of Health Estimation for Smart Batteries using Transfer Learning with Data Cleaning," *IFAC-PapersOnLine*, vol. 56, no. 2, hal. 3782–3787, 2023, doi: 10.1016/j.ifacol.2023.10.1306.

[23] W. He, K. Farrahi, B. Chen, B. Peng, dan A. Villavicencio, "Representation transfer and data cleaning in multi-views for text simplification," *Pattern Recognit. Lett.*, vol. 177, no. October 2023, hal. 40–46, 2024, doi: 10.1016/j.patrec.2023.11.011.

[24] R. Nadlifatin *et al.*, "Exploring the Effectiveness of Work from Home: A Text Mining Analysis of Employee Perceptions and Experiences," *Procedia Comput. Sci.*, vol. 234, hal. 448–454, 2024, doi: 10.1016/j.procs.2024.03.026.

[25] Z. Grotenhuis, P. J. Mosteiro, dan A. M. Leeuwenberg, "Modest performance of text mining to extract health outcomes may be almost sufficient for high-quality prognostic model development," *Comput. Biol. Med.*, vol. 170, no. July 2023, hal. 108014, 2024, doi: 10.1016/j.compbiomed.2024.108014.

[26] Zulkarnain dan T. D. Putri, "Intelligent transportation systems (ITS): A systematic review using a Natural Language Processing (NLP) approach," *Heliyon*, vol. 7, no. 12, hal. e08615, 2021, doi: 10.1016/j.heliyon.2021.e08615.

[27] A. E. Kwabena, O. B. Wiafe, B. D. John, A. Bernard, dan F. A. F. Boateng, "An automated method for developing search strategies for systematic review using Natural Language Processing (NLP)," *MethodsX*, vol. 10, no. November 2021, hal. 101935, 2023, doi: 10.1016/j.mex.2022.101935.

[28] J. Frei dan F. Kramer, "Annotated dataset creation through large language models for non-english medical NLP," *J. Biomed. Inform.*, vol. 145, no. January, hal. 104478, 2023, doi: 10.1016/j.jbi.2023.104478.

[29] Z. B. Jin *et al.*, "Stemming retinal regeneration with pluripotent stem cells," *Prog. Retin. Eye Res.*, vol. 69, no. January 2018, hal. 38–56, 2019, doi: 10.1016/j.preteyeres.2018.11.003.

[30] A. Ramadhanu, Mardison, H. Hendri and F. Hadi, "Organic Fertilizer Content Detection Based on Image Segmentation and Texture Analysis," 2025 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS), Jakarta, Indonesia, 2025, pp. 50-55, doi: 10.1109/ICIMCIS68501.2025.11327142.

[31] Divya Khyani, Siddhartha B S, "An Interpretation of Lemmatization and Stemming in Natural Language An Interpretation of Lemmatization and Stemming in Natural Language Processing," *J. Univ. Shanghai Sci. Technol.*, vol. 22, no. 10, October, hal. 350, 2021.

[32] W. Akram dan R. Kumar, "International Journal of Computer Sciences and Engineering Open Access," *Int. J. Comput. Sci. Eng.*, vol. 5, no. 10, hal. 347–354, 2017, doi: 10.26438/ijcse/v6i1.161167.

[33] G. W. Nurcahyo, A. P. Gusman and H. Hendri, "Literature Study on Online Learning as an Impact of Covid 19 Pandemic in Education," 2021 International Conference on Computer Science and Engineering (IC2SE), Padang, Indonesia, 2021, pp. 1-5, doi: 10.1109/IC2SE52832.2021.9792065.

[34] D. H. Maulud *et al.*, "Review on Natural Language Processing Based on Different Techniques," *Asian J. Res. Comput. Sci.*, vol. 10, no. 1, hal. 1–17, 2021, doi: 10.9734/ajrcos/2021/v10i130231.

[35] M. Abbas, K. Ali, A. Jamali, K. Ali Memon, dan A. Aleem Jamali, "Multinomial Naive Bayes Classification Model for Sentiment Analysis," *IJCSNS Int. J. Comput. Sci. Netw. Secur.*, vol. 19, no. 3, hal. 62, 2019, doi: 10.13140/RG.2.2.30021.40169.

[36] R. Blanquero, E. Carrizosa, P. Ramírez-Cobo, dan M. R. Sillero-Denamiel, "Variable selection for Naïve Bayes classification," *Comput. Oper. Res.*, vol. 135, hal. 105456, 2021, doi: 10.1016/j.cor.2021.105456.

[37] H. Hendri, Yuhandri and A. Ramadhanu, "GoogLeNet-Based Deep Learning Framework for Underwater Microplastic Classification in Marine Environments," 2025 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS), Jakarta, Indonesia, 2025, pp. 44-49, doi: 10.1109/ICIMCIS68501.2025.11327223.