CEPAT Journal of Computer Engineering: Progress, Application and Technology
Vol. 5, No. 2, November 2026, pp. 192~205
ISSN: 2963-6728, DOI: https://10.25124/cepat.v5i02.12451

Preprocessing Choices and N-gram Representation in Indonesian Product Review Sentiment Classification

Akmal Faiz Agiputra[1], Jumanto Unjung[1]
[1]Department of Computer Science, Universitas Negeri Semarang, Semarang, Indonesia

Article Info

Article history:

Received October 28, 2026
Revised November 07, 2026
Accepted November 09, 2026

Keywords:

Indonesian Sentiment Analysis
Text Preprocessing
TF-IDF
N-gram
Multilayer Perceptron

ABSTRACT

The large volume of unstructured Indonesian product reviews on e-commerce platforms creates the need for effective sentiment classification methods. Previous studies have shown that the effectiveness of preprocessing techniques, Term Frequency-Inverse Document Frequency (TF-IDF) n-gram representations, and classification models may vary across datasets and experimental settings. This study analyzes the interaction between preprocessing scenarios, TF-IDF feature representations, and classification models for Indonesian product review sentiment classification. Experiments were conducted on the PRDECT-ID dataset using eight preprocessing scenarios, two TF-IDF n-gram representations, and Logistic Regression and Multilayer Perceptron (MLP) classifiers, evaluated with five-fold StratifiedGroupKFold cross-validation, with normalized duplicate reviews assigned to the same fold. The best configuration combined case folding and text normalization, unigram-bigram features, and MLP-3, achieving a mean F1-score of 94.83%. Statistical testing using the Nadeau–Bengio corrected resampled t-test with Holm adjustment found significant improvements for unigram-bigram over unigram and for case folding plus normalization over basic cleaning. Adding stopword removal or stemming did not yield significant improvements. These findings indicate that feature representation and selected normalization steps contributed more consistently to classification performance than additional preprocessing operations or increased MLP architecture complexity.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Akmal Faiz Agiputra
Department of Computer Science
Universitas Negeri Semarang
Semarang, Indonesia
Email: faizagiputra@students.unnes.ac.id

1. INTRODUCTION

Online product reviews provide consumers with information about product quality, performance, and value, and the information they contain can influence purchasing decisions [1]. These reviews generally consist of large amounts of user-generated, unstructured text and reflect diverse consumer experiences, opinions, and expressions [2]. This characteristic makes the analysis process challenging, particularly when opinions must be extracted from massive volumes of comments, as efficiently analyzing such data can be difficult without automated methods [3]. Therefore, an automated approach is needed to process and classify user opinions more efficiently, one of which is sentiment analysis, a natural language processing technique for recognizing and classifying sentiment expressed in textual data [4].

The application of sentiment analysis to product review data is not without its challenges. User-generated text may contain noise and irrelevant elements, including URLs, hashtags, mentions, and special characters, which can complicate text processing and sentiment classification [5]. In this context, preprocessing is a crucial step to improve data quality and consistency before classification [6]. This stage typically includes cleaning and normalizing the text to remove irrelevant elements and standardize textual

inputs, before the data is transformed into a numerical representation that can be processed by a machine learning model [7], [8].

However, the effectiveness of preprocessing is not always consistent and depends on the technique used. Variations in preprocessing techniques can affect model performance, as more aggressive text cleaning may reduce classification performance by removing important emotional and lexical signals [9]. In addition, the effect of the same preprocessing technique on model performance can vary across classifiers, as some classifiers may benefit from a technique while others may experience reduced performance [10]. This condition shows that preprocessing results can vary on certain models and data conditions, so its effectiveness cannot be generalized and needs to be analyzed systematically in various combinations of techniques and models. In addition to preprocessing factors, sentiment analysis performance is also influenced by the feature representation method and classification algorithm used [11], [12].

Several previous studies have evaluated the effect of preprocessing techniques on sentiment analysis performance on text data. Research by [13] shows that certain preprocessing combinations can improve classification performance, especially on noisy informal text data. In the Indonesian context, [14] experimentally compared four preprocessing configurations across different feature extraction methods and classification models. Their findings showed that the configuration without additional preprocessing achieved the highest average F1-score, indicating that techniques such as stopword removal and stemming do not necessarily improve classification performance. However, research by [13] also found that certain preprocessing techniques, such as stemming and stopword removal, can actually degrade model performance under certain conditions because they affect the quality of feature representation and the information extraction process in text. Furthermore, research by [15] reports that preprocessing has a relatively small impact on several contextual embedding-based deep learning models. These differences in results indicate that the effectiveness of preprocessing can vary depending on the data characteristics, feature representation, and classification model used.

In addition to preprocessing, feature representation is also an important factor in sentiment analysis because it influences how information in text is represented numerically before being used in the classification process. One feature representation method widely used in text classification is TF-IDF [16]–[18]. Several studies have shown that TF-IDF still provides competitive performance compared to other representation methods such as Bag of Words and Word2Vec in certain text classification tasks [19], [20].

In TF-IDF applications, n-gram-based feature representations are also widely used to capture word patterns in text, including the use of unigrams, bigrams, and trigrams [21], [22]. Research by [21] shows that different n-gram configurations can produce different classification performance, with the optimal n-gram configuration depending on the classification setting and dataset characteristics. Meanwhile, research by [22] shows that bigram and trigram representations are able to show differences between sentiment classes more clearly than unigrams, although the addition of these features does not always significantly improve classification accuracy. In addition, research by [23] used a combination of unigrams, bigrams, and trigrams as contextual feature sets in a machine learning model and showed that the combination of unigrams and bigrams was able to provide competitive performance on sentiment analysis tasks. These results indicate that the effectiveness of n-gram representation is still influenced by data characteristics and the classification method used.

Various classification algorithms have been used in sentiment analysis, ranging from traditional machine learning methods to artificial neural network-based models. Traditional machine learning models such as Logistic Regression are still widely used in text classification because they are able to provide competitive performance on TF-IDF-based sentiment analysis tasks [23], [24]. In the context of sentiment analysis of Indonesian language product reviews, research by [22] shows that conventional machine learning methods are still able to provide competitive performance compared to deep learning models under certain dataset conditions. In addition, research by [23] also reports that traditional machine learning models are able to produce better performance than some deep learning models under certain dataset conditions, especially when the dataset size is not large enough to support the learning process of the deep learning model optimally. On the other hand, neural network-based models such as MLP have also been widely used in sentiment classification tasks. Research by [11] states that MLP is still a viable approach to use in sentiment analysis despite its limitations in capturing hierarchical relationships and long-term dependencies in text data. In the Indonesian context, research by [25] also shows that a combination of TF-IDF and MLP can be used to classify sentiment on Indonesian text data. The differences in results across these studies indicate that the relationship between preprocessing, feature representation, and classification methods still requires further analysis in sentiment analysis tasks.

However, previous research has shown that the effectiveness of preprocessing, n-gram representation, and classification methods in sentiment analysis still yields varying results across data characteristics and models. Some studies report that more comprehensive preprocessing can improve

classification performance, while others show that techniques such as stemming and stopword removal do not always provide consistent performance improvements. Similar variations in results are also found when using n-gram representation and comparing classification methods, where specific feature and model combinations can yield different performance across different datasets. Nevertheless, most previous studies still use fixed preprocessing pipelines or focus solely on evaluating the final model performance without analyzing the gradual influence of preprocessing in combination with feature representation and classification methods in an integrated manner. Therefore, research is needed that can more systematically analyze the relationship between preprocessing, TF-IDF and n-gram-based feature representation, and classification model variations in sentiment analysis of Indonesian-language product reviews.

Based on these issues, this study aims to analyze the influence of preprocessing stages on sentiment classification performance in Indonesian-language product reviews using TF-IDF feature representation and machine learning classification models. This study also evaluates the effect of using unigram representation and unigram-bigram combination, compares the performance of MLP with Logistic Regression as a baseline, and analyzes the effect of MLP architecture variations on classification results. The evaluation is conducted using accuracy, precision, recall, and F1-score metrics to gain a more systematic understanding of the relationship between preprocessing, feature representation, and classification models in sentiment analysis.

2. METHOD

This research was conducted through several stages: text preprocessing, feature extraction using TF-IDF, classification using Logistic Regression and MLP, and model performance evaluation using predetermined evaluation metrics. The general research flow is shown in Figure 1.

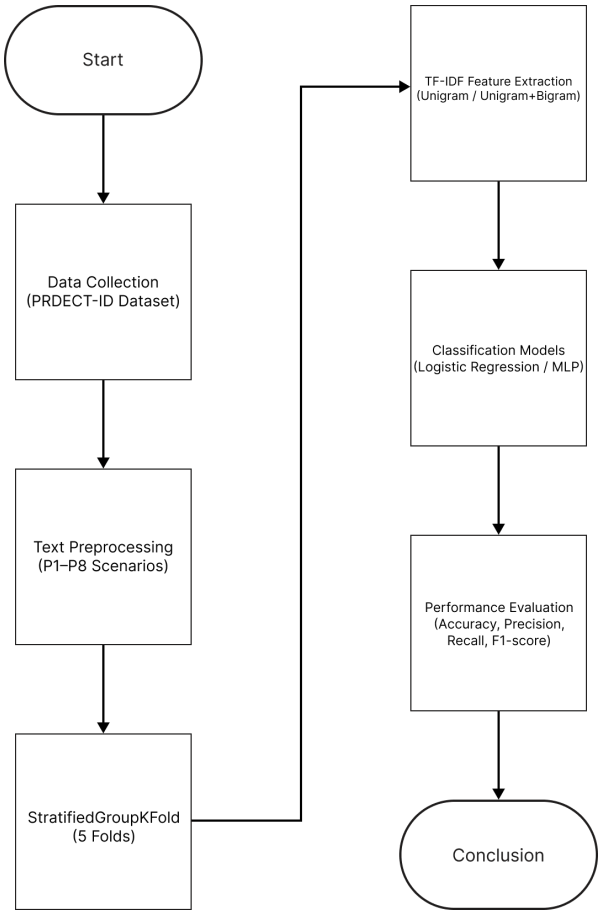


Figure 1. Research Flowchart

2.1. Dataset

This study uses the publicly available PRDECT-ID dataset, which contains 5400 Indonesian product reviews collected from 29 product categories on Tokopedia. The dataset provides two annotation fields: five

emotion categories (Anger, Fear, Happy, Love, and Sadness) and binary sentiment labels (Negative and Positive). The dataset was annotated by three annotators, with emotion annotation guided by criteria developed with an expert in clinical psychology. The annotations underwent peer review and random checks [26]. In the dataset file used in this study, Anger, Fear, and Sadness are consistently associated with Negative sentiment, whereas Happy and Love are associated with Positive sentiment. This study directly uses the provided Sentiment column as the classification target, encoded as Negative = 0 and Positive = 1, without deriving sentiment labels from the Emotion column.

The dataset was chosen because it represents the characteristics of user reviews on e-commerce platforms, which generally contain informal language, abbreviations, spelling variations, and noisy text, all relevant to the preprocessing process in sentiment analysis. A duplicate review audit was conducted to identify repeated review texts. The exact-text audit identified 83 distinct duplicated review texts involving 178 records. After converting the review texts to lowercase, removing leading and trailing spaces, and collapsing consecutive whitespace, 91 normalized duplicate groups involving 203 records were identified. The higher number of normalized duplicate groups resulted from different raw text variations becoming identical after normalization. No normalized duplicate group contained conflicting sentiment labels. All records were retained in the dataset, and duplicate reviews were not removed. Instead, normalized review texts were used to define groups during cross-validation to prevent identical reviews from being distributed across training and testing folds. The characteristics of the dataset used in this study are shown in Table 1.

Table 1. Dataset Description

Attribute	Description
Dataset	PRDECT-ID
Language	Indonesian
Domain	Product Reviews
Dataset Type	Public Dataset
Total Data	5400
Positive Reviews	2579
Negative Reviews	2821
Data Characteristics	Informal text, abbreviations, spelling variations, and noisy text

2.2. Text Preprocessing

The preprocessing stage was performed to reduce noise and improve the consistency of text data before the feature extraction process. In this study, preprocessing consisted of case folding, text normalization, stopword removal, and stemming.

Case folding is a preprocessing technique that converts text into lowercase to reduce token variation and vocabulary sparsity caused by capitalization differences [27], [28]. Text normalization was used to convert noisy and non-standard words into standard lexical forms, including variations caused by abbreviations and informal language use [29]. Stopword removal was used to eliminate high-frequency words that contribute little semantic or informational value to text documents [30]. However, negation words such as “tidak”, “bukan”, “kurang”, “jangan”, and “belum” were excluded from the stopword list because they contain important sentiment information in Indonesian sentiment analysis tasks. Stemming was performed to reduce affixed words into their root forms using the Sastrawi stemming library, a commonly used approach in Indonesian NLP tasks [27]. Examples of each technique can be seen in Table 2.

Table 2. Examples of Text Preprocessing

Technique	Example
Case Folding	“Produk BAGUS” → “produk bagus”
Normalization	“gk” → “tidak”
Stopword Removal	“produk ini bagus” → “produk bagus”
Stemming	“pengiriman” → “kirim”

In addition to being used individually, several preprocessing techniques were also combined in several experimental scenarios to analyze the effect of preprocessing on sentiment classification performance. Before applying each preprocessing scenario, all text data underwent a basic cleaning stage consisting of URL removal, mention and hashtag removal, number removal, punctuation removal, repeated character reduction, extra space, and whitespace normalization. Repeated characters occurring three or more times were reduced to two characters. This cleaning stage was consistently applied across all preprocessing scenarios to reduce textual noise and maintain data consistency.

Text normalization was performed using the new_kamusalay dictionary released by Ibrohim and Budi [31]. The dictionary contains typos and slang expressions mapped to their corresponding formal forms. The dictionary was originally developed for Indonesian Twitter text normalization by combining resources

from previous studies with additional entries collected from the authors' dataset. In this study, it was reused as a lexical normalization resource for Indonesian product reviews. Since the dictionary was not specifically developed for e-commerce reviews, its coverage may not include all informal expressions found in the PRDECT-ID dataset.

Eight preprocessing scenarios (P1 – P8) were designed to compare selected combinations of case folding, text normalization, stopword removal, and stemming. The scenarios include individual techniques and selected combinations to examine how different preprocessing choices relate to classification performance. All preprocessing scenarios were applied consistently before TF-IDF feature extraction. Details of these scenarios can be seen in Table 3.

Table 3. Preprocessing Scenarios

Technique	Example
P1	Basic Cleaning
P2	Basic Cleaning + Case Folding
P3	Basic Cleaning + Text Normalization
P4	Basic Cleaning + Stopword Removal
P5	Basic Cleaning + Stemming
P6	Basic Cleaning + Case Folding + Text Normalization
P7	Basic Cleaning + Case Folding + Text Normalization + Stopword Removal
P8	Basic Cleaning + Case Folding + Text Normalization + Stopword Removal + Stemming

2.3. Feature Extraction

After the preprocessing stage is carried out, the text data is transformed into a numerical representation using the TF-IDF method. TF-IDF is used to measure the importance of a word in a document by considering its frequency within a document and its relevance across the entire corpus [32].

In this study, TF-IDF feature representation was used with two variations of n-grams: unigram and a unigram-bigram combination. Unigram representation represents each word individually, while bigram represents pairs of two consecutive words to capture local context in the text.

TF-IDF feature extraction was implemented using the TfidfVectorizer from the scikit-learn library. The lowercase parameter in TfidfVectorizer was disabled (lowercase=False) to ensure that case folding effects were evaluated explicitly through preprocessing scenarios rather than automatically applied during feature extraction. The number of TF-IDF features was limited using the max_features parameter of 5000 to maintain a balance between information representation and computational complexity. Examples of unigram and bigram can be seen in Table 4.

Table 4. N-gram Representation Example

Sentence	Representation
produk tidak bagus	unigram: produk, tidak, bagus
produk tidak bagus	bigram: produk tidak, tidak bagus

2.4. Classification Models

This study used two classification methods: Logistic Regression and MLP. Logistic Regression was used as the baseline classifier because it is a widely used machine learning method for text classification and has demonstrated strong performance in TF-IDF-based text classification tasks [33]. Logistic Regression was implemented using the scikit-learn library with the maximum iteration parameter set to 1000 and random state to 42 to ensure convergence during training.

Meanwhile, MLP is a feedforward artificial neural network consisting of an input layer, one or more hidden layers, and an output layer [33]. In this study, MLP was implemented using the scikit-learn library with ReLU activation and Adam optimizer. ReLU activation was selected because of its computational efficiency and its ability to reduce vanishing gradient problems during training. The maximum number of iterations was set to 200 and random state to 42 for all experiments to maintain training consistency while avoiding excessive computational cost. Several hidden layer configurations were evaluated to analyze the effect of model complexity on sentiment classification performance. Early stopping was enabled with n_iter_no_change set to 10 to reduce overfitting and terminate training when no significant performance improvement was observed. Details of the evaluated MLP architecture variations are presented in Table 5, while the general network structure is illustrated in Figure 2.

Table 5. MLP Architecture Variations

Model	Hidden Layer Architecture
MLP-1	50
MLP-2	100
MLP-3	100-50
MLP-4	100-100

Table 5 summarizes the hidden layer configurations evaluated in this study. Four MLP architectures with different numbers of neurons and hidden layers were designed to examine the effect of network complexity on sentiment classification performance.

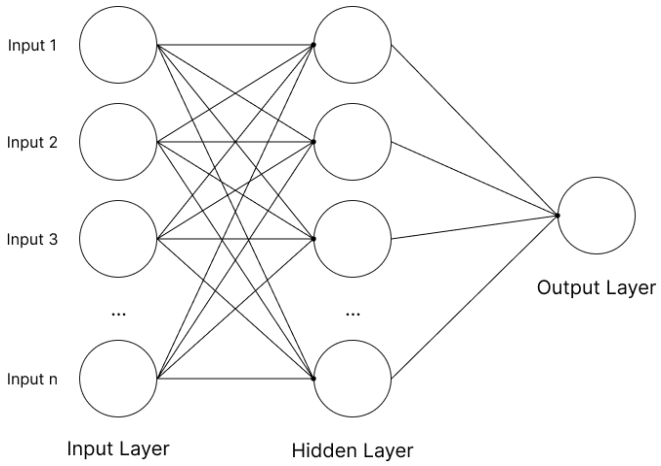


Figure 2. General Architecture of the Evaluated MLP Models

Figure 2 illustrates the general architecture of the MLP models used in this study. The network consists of an input layer representing TF-IDF features, one or more hidden layers, and an output layer for binary sentiment classification. Depending on the experimental configuration, the number of hidden layers and neurons varied across the evaluated architectures. The specific configurations are summarized in Table 5, including single-hidden-layer and multi-hidden-layer models. This design enables the investigation of the impact of network complexity on sentiment classification performance.

2.5. Experimental Setup

The experiments combined eight preprocessing scenarios, two TF-IDF feature representations, and five classification configurations, resulting in 80 experimental configurations. Model evaluation was performed using StratifiedGroupKFold with five folds to maintain an approximately balanced distribution of sentiment classes while ensuring that identical normalized review texts were assigned to the same fold.

The grouping key was created from the original review text by converting it to lowercase, removing leading and trailing whitespace, and replacing consecutive whitespace characters with a single space. This normalization was used only to identify review groups for cross-validation. It did not replace or modify the text produced by the preprocessing scenarios.

A preliminary duplicate leakage audit using conventional StratifiedKFold identified 71 normalized duplicate groups involving 163 records that were distributed across different folds. This finding motivated the use of group-aware cross-validation. In the final StratifiedGroupKFold split, the dataset contained 5288 unique normalized review groups, and no group was shared between the training and testing sets in any fold. All 5400 records were included in the testing sets exactly once.

The five folds used training sets containing 4311 to 4326 reviews and testing sets containing 1074 to 1,089 reviews. The proportion of positive reviews in the testing folds ranged from 46.19% to 49.16%. A fixed random state of 42 was used when generating the folds to support reproducibility. The same fold assignments were used across all 80 experimental configurations to ensure consistent comparisons. Within each fold, the TF-IDF vectorizer was fitted exclusively on the training data and subsequently used to transform the corresponding testing data. This procedure was implemented using a scikit-learn Pipeline to prevent information from the testing fold from influencing feature extraction.

Details of the experimental configuration and experiment combinations are presented in Table 6 and Table 7, respectively.

Table 6. Experimental Configuration

Component	Configuration
Validation Method	StratifiedGroupKFold (5 folds)
Grouping Strategy	Normalized Review Text
Feature Extraction	TF-IDF
N-gram Variation	Unigram, Unigram+Bigram
Max Features	5000
Baseline Model	Logistic Regression
Neural Network Model	Multilayer Perceptron
Activation Function	ReLU
Optimizer	Adam
MLP Maximum Iterations	200
MLP Early Stopping	Enabled
MLP Patience	10

Table 7. Master Experiment

Experiment Component	Description
Preprocessing Scenarios	P1 - P8
Feature Representation	Unigram, Unigram+Bigram
Classification Models	LR, MLP-1, MLP-2, MLP-3, MLP-4
Total Experiment Combinations	80

All experiments were implemented in Python using the scikit-learn library for TF-IDF feature extraction and classification models. Indonesian stemming was performed using the Sastrawi library. Model evaluation was conducted using StratifiedGroupKFold (5 folds). To improve experiment reproducibility, a random state value of 42 was used in the classification models and stratified group process.

2.6. Evaluation Metrics

Model performance was evaluated using accuracy, precision, recall, and F1-score metrics. The use of multiple evaluation metrics aims to provide a more comprehensive analysis of classification performance.

Accuracy measures the proportion of correctly classified instances among all instances in the dataset. Precision measures the proportion of correctly predicted positive instances among all instances predicted as positive, while recall measures the model's ability to correctly identify all actual positive instances. The F1-score represents the harmonic mean of precision and recall.

The metrics were calculated separately for each testing fold. Their mean values were then obtained by averaging the five fold-level scores. Standard deviation was also calculated across the five fold scores to describe performance variability. The standard deviation was computed using the population standard deviation convention (ddof=0).

2.7. Statistical Testing

Four planned statistical comparisons were conducted using fold-level F1-scores. For the n-gram comparison (Unigram+Bigram vs. Unigram), the F1-scores of each representation were averaged across the eight preprocessing scenarios and five classifiers within each fold. For the preprocessing comparisons (P6 vs. P1, P7 vs. P6, and P8 vs. P7), the scores were averaged across the two feature representations and five classifiers within each fold. The resulting five paired fold-level differences were evaluated using the Nadeau–Bengio corrected resampled t-test [34]. The average test-to-training sample-size ratio was used in the variance correction. The four resulting p-values were adjusted using Holm's procedure [35], with statistical significance assessed at a significance level of 0.05. A positive mean difference indicates that the first condition in each comparison achieved a higher F1-score. Pairing was maintained by comparing results from the same fold assignments across the two conditions in each comparison.

3. RESULTS AND DISCUSSION

A total of 80 experimental configurations were evaluated using five-fold StratifiedGroupKFold cross-validation. Model performance was evaluated using accuracy, precision, recall, and F1-score metrics. Although all evaluation metrics were calculated, the discussion primarily focuses on F1-score because it provides a balanced representation of precision and recall performance across classification models. The top 3 configurations for each model result are presented in Table 8.

Table 8 presents the top-performing configurations for each classification model based on the F1-score metric. The results show that different preprocessing scenarios and feature representation combinations appeared among the highest-performing configurations across Logistic Regression and MLP models. The unigram-bigram representation was frequently observed among top-performing configurations, while several preprocessing combinations consistently appeared in high-ranking results.

Table 8. Top 3 Configurations for Each Model

Preprocessing	Feature Representation	Model	F1 Mean	F1 Std
P6	Unigram+Bigram	MLP-3	0.948288	0.007194
P7	Unigram+Bigram	MLP-3	0.947928	0.004854
P6	Unigram+Bigram	MLP-2	0.947258	0.008008
P7	Unigram+Bigram	MLP-4	0.947006	0.004671
P7	Unigram+Bigram	MLP-2	0.946102	0.006302
P7	Unigram+Bigram	LR	0.945138	0.008295
P8	Unigram+Bigram	MLP-3	0.945062	0.005187
P6	Unigram+Bigram	MLP-4	0.944333	0.010592
P7	Unigram+Bigram	MLP-1	0.944214	0.004809
P6	Unigram+Bigram	MLP-1	0.944093	0.006686
P6	Unigram+Bigram	LR	0.943782	0.006620
P8	Unigram+Bigram	MLP-4	0.943203	0.005161
P3	Unigram+Bigram	MLP-2	0.942381	0.008537
P8	Unigram+Bigram	LR	0.941705	0.005926
P8	Unigram+Bigram	MLP-1	0.940575	0.002960

The highest overall performance was achieved by MLP-3 using P6 preprocessing and unigram-bigram representation, with a mean F1-score of 0.948288 and a standard deviation of 0.007194. This configuration also achieved an average accuracy of 0.951282, precision of 0.960863, and recall of 0.936445. The second highest configuration was MLP-3 with P7 preprocessing and unigram-bigram representation, achieving a mean F1-score of 0.947928. The difference between these two configurations was relatively small. All three highest-performing configurations for each classifier also used the unigram-bigram representation, indicating that this representation frequently appeared among the better-performing configurations.

3.1. Analysis of Preprocessing Scenarios

This section analyzes the effect of preprocessing scenarios on sentiment classification performance across different classification models and feature representations. The analysis was conducted to examine whether specific preprocessing techniques consistently improved classification performance under different experimental settings. The values represent the average F1-score across both unigram and unigram+bigram feature representations.

Table 9. Preprocessing and Model

Preprocessing	LR	MLP-1	MLP-2	MLP-3	MLP-4
P1	0.926578	0.930123	0.931038	0.932357	0.933396
P2	0.934839	0.934508	0.938331	0.939651	0.937902
P3	0.933047	0.928962	0.933829	0.934514	0.934899
P4	0.924479	0.927350	0.930228	0.932121	0.932555
P5	0.932985	0.935303	0.935895	0.937965	0.933562
P6	0.940754	0.937369	0.941969	0.942615	0.940465
P7	0.941290	0.936021	0.939629	0.940629	0.939381
P8	0.937060	0.933253	0.933700	0.938162	0.936347

Table 9 shows the relationship between preprocessing scenarios and classification models. The results indicate that classification performance varied across preprocessing scenarios for both Logistic Regression and MLP models. Several preprocessing configurations produced relatively higher F1-scores across multiple models, while other preprocessing scenarios showed lower performance trends.

P6 produced the highest average F1-score among the evaluated preprocessing scenarios for all four MLP architectures, whereas P7 achieved the highest value for Logistic Regression. These findings indicate that the most favorable preprocessing configuration depended on the classifier used.

When averaged across all five classifiers, P6 obtained the highest overall value of 0.940634, followed by P7 with 0.939390. In contrast, P4 produced the lowest overall average of 0.929347. The relatively lower performance of P4 should be interpreted in relation to its implementation, as stopword removal in this scenario was applied without prior case folding. Consequently, stopword matching followed case-sensitive lookup behavior, which may have affected the words removed during preprocessing.

The comparison between P7 and P8 also shows that adding stemming did not improve the average F1-score in this experiment. P8 produced lower average values than P7 across all five classifiers, even though it included all preprocessing operations in P7 with the additional stemming stage. This result suggests that applying more preprocessing operations did not necessarily lead to better classification performance. However, the findings are limited to the evaluated preprocessing configurations and should not be interpreted as evidence that stemming is universally ineffective for Indonesian text classification.

Table 10. Preprocessing and N-gram

Preprocessing	Unigram	Unigram+Bigram
P1	0.928690	0.932707
P2	0.934951	0.939141
P3	0.927720	0.938380
P4	0.925572	0.933120
P5	0.931218	0.939066
P6	0.935717	0.945551
P7	0.932702	0.946078
P8	0.929021	0.942387

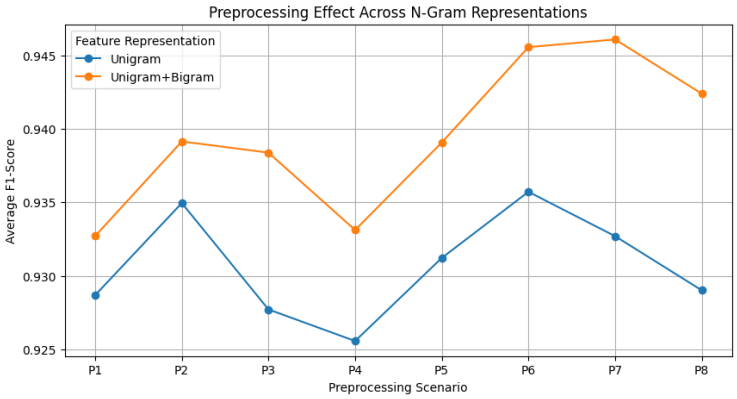


Figure 3. Line Chart of Preprocessing and N-gram

Table 10 and Figure 3 present the comparison between unigram and unigram-bigram representations across the eight preprocessing scenarios. The results show that unigram-bigram achieved a higher average F1-score than unigram in all eight scenarios. Across the preprocessing scenarios, the overall average F1-score increased from 0.930699 for unigram to 0.939554 for unigram-bigram, corresponding to an improvement of approximately 0.008855.

The magnitude of this improvement varied across preprocessing scenarios. The largest differences were observed in P7 and P8, with increases of 0.013376 and 0.013366, respectively. In contrast, P1 showed the smallest difference, with an increase of 0.004017. The highest average F1-score for unigram-bigram was obtained by P7, reaching 0.946078, while P6 achieved the highest average for unigram at 0.935717.

The improvement obtained from unigram-bigram may be related to its ability to represent adjacent word combinations in addition to individual words. Such combinations can preserve local lexical patterns, including short expressions whose meaning may depend on the relationship between consecutive words. However, the present experiment did not examine individual feature contributions; therefore, this explanation should be considered a possible interpretation rather than a direct attribution of the observed performance improvement.

3.2. Analysis of N-gram Representation

This section analyzes the effect of TF-IDF n-gram representation on sentiment classification performance. The analysis compares unigram representation with the unigram-bigram combination across different classification models. The comparison results are presented in Table 11 and Figure 4.

Table 11. N-gram and Model

Model	Unigram	Unigram+Bigram	Bigram Gain
LR	0.931093	0.936665	0.005571
MLP-1	0.927553	0.938169	0.010616
MLP-2	0.930662	0.940493	0.009831
MLP-3	0.932502	0.942001	0.009499
MLP-4	0.931685	0.940442	0.008757

Table 11 and Figure 4 further examine the relationship between n-gram representation and classifier performance by averaging the results across all eight preprocessing scenarios. The unigram-bigram representation achieved a higher average F1-score than unigram for all five classifiers. The largest improvement was obtained by MLP-1, with an F1 difference of 0.010615, while Logistic Regression showed the smallest improvement of 0.005571. This indicates that the magnitude of the improvement associated with unigram-bigram representation varied across classifiers.

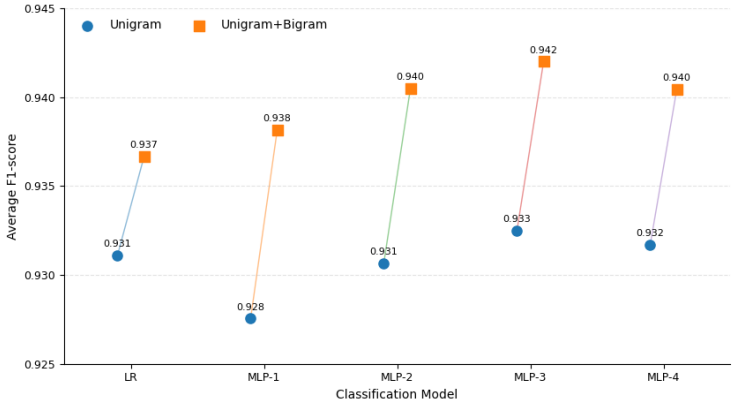


Figure 4. Dot Plot of Feature Representation and Models

Regarding the MLP architectures, MLP-3 achieved the highest average F1-score for both unigram and unigram-bigram representations, with values of 0.932502 and 0.942001, respectively. MLP-4, which used a larger second hidden layer than MLP-3, did not achieve higher average performance. This observation indicates that increasing the number of hidden units did not consistently improve classification performance within the evaluated architectures. In addition, Logistic Regression achieved a higher average F1-score than MLP-1 and MLP-2 when using unigram features, whereas all MLP configurations achieved higher average scores than Logistic Regression when using unigram-bigram features.

3.3. Overall Discussion

Overall, the results demonstrate that classification performance varied according to the combination of preprocessing scenario, feature representation, and classifier. Preprocessing combinations involving case folding and text normalization, particularly P6 and P7, generally produced higher average scores, although their relative performance depended on the classifier. The addition of stemming in P8 did not improve the average results compared with P7. In contrast, unigram-bigram representation showed a more consistent improvement in average F1-score across the evaluated preprocessing scenarios and classifiers. Nevertheless, the magnitude of performance differences varied across experimental configurations. Since descriptive rankings alone do not establish statistical significance, planned statistical comparisons were conducted to further examine the average effects of n-gram representation and selected preprocessing combinations, as presented in Section 3.4. Therefore, the findings should be interpreted as descriptive evidence of performance differences within the PRDECT-ID dataset and the evaluated experimental configurations.

3.4. Statistical Significance and Error Analysis

Statistical significance was evaluated using the Nadeau–Bengio corrected resampled t-test, followed by Holm adjustment for the four planned comparisons. The results in Table 12 show that Unigram+Bigram significantly outperformed Unigram, with a mean F1-score improvement of 0.008855 (Holm-adjusted p=0.005682). Similarly, P6 significantly outperformed P1, with a mean improvement of 0.009936 (Holm-adjusted p=0.000774). However, the addition of stopword removal (P7 vs. P6) and stemming (P8 vs. P7) did not produce statistically significant improvements. These comparisons represent marginal effects averaged across the evaluated classifiers and feature representations.

Table 12. Statistical comparison results

Comparison	Mean ΔF1	Corrected SE	Holm-adjusted p-value	Result
Unigram+Bigram vs. Unigram	+0.008855	0.001217	0.005682	Significant
P6 vs. P1	+0.009936	0.000756	0.000774	Significant
P7 vs. P6	−0.001244	0.002824	0.682279	Not significant
P8 vs. P7	−0.003686	0.002386	0.394511	Not significant

The out-of-fold (OOF) predictions of the best configuration (P6, Unigram+Bigram, and MLP-3) covered all 5400 reviews and yielded an accuracy of 95.13%. Table 13 presents the confusion matrix and class-wise performance. The model correctly classified 2722 negative reviews and 2415 positive reviews, while 99 negative reviews were incorrectly classified as positive and 164 positive reviews were classified as

negative. The recall for the Negative class (96.49%) was higher than that of the Positive class (93.64%), indicating that the model missed more positive reviews than negative reviews.

Table 13. Confusion Matrix and Class-wise Performance Based on OOF Predictions

Class	Pred. Negative	Pred. Positive	Precision	Recall	F1-score
Negative	2,722	99	0.9432	0.9649	0.9539
Positive	164	2,415	0.9606	0.9364	0.9484

The OOF predictions contained 263 misclassified reviews, consisting of 164 false negatives and 99 false positives. An inspection of selected misclassified examples revealed difficulties involving informal expressions, negation, and concise sentiment expressions. For example, the negative review "No good!!! Tidak puassss" was classified as positive, whereas the positive review "mantap. saya sudah beberapa kali order ulang" was classified as negative. These cases illustrate that TF-IDF features did not always capture the sentiment conveyed by the complete review. Since the best configuration was selected based on the same cross-validation results, the OOF metrics are presented as post-selection diagnostic results rather than as an independent test-set estimate. The statistical comparisons were based on a single five-fold cross-validation partition. Therefore, the significance findings should be interpreted within the evaluated validation setup and dataset.

4. CONCLUSION

This study analyzed the influence of preprocessing scenarios, TF-IDF feature representations, and classification models on Indonesian product review sentiment classification. The results showed that preprocessing effectiveness depended on the combination of techniques and classifiers. The best configuration, combining P6 preprocessing, unigram-bigram representation, and MLP-3, achieved a mean F1-score of 0.948288. Statistical testing showed that unigram-bigram representation significantly outperformed unigram, while P6 also significantly outperformed P1. However, adding stopword removal and stemming did not produce statistically significant marginal improvements. These findings suggest that feature representation and selected normalization techniques contributed more consistently to performance than additional preprocessing operations or increased MLP architectural complexity.

The OOF predictions of the selected configuration achieved an accuracy of 95.13%, although these results represent post-selection diagnostics rather than an independent test-set estimate. This study was limited to a single dataset, TF-IDF feature representation, and the evaluated classifier configurations. Furthermore, statistical comparisons were based on a single five-fold cross-validation partition. Future work may investigate additional datasets, contextual embedding methods, broader hyperparameter optimization, and repeated validation procedures to further examine the generalizability of the findings.

REFERENCES

[1] S.-Q. Gu, S.-T. Liu, Y. Liu, and J.-M. Ding, "A novel product ranking approach considering sentiment intensity distribution of online reviews," *International Journal of Computational Intelligence Systems*, vol. 17, no. 1, Nov. 2024, Art. no. 277, doi: 10.1007/s44196-024-00688-1.

[2] L. G. Atlas *et al*., "A modernized approach to sentiment analysis of product reviews using BiGRU and RNN based LSTM deep learning models," *Scientific Reports*, vol. 15, no. 1, May 2025, Art. no. 16642, doi: 10.1038/s41598-025-01104-0.

[3] Y. Mao, Q. Liu, and Y. Zhang, "Sentiment analysis methods, applications, and challenges: A systematic literature review," *Journal of King Saud University Computer and Information Sciences*, vol. 36, no. 4, Apr. 2024, Art. no. 102048, doi: 10.1016/j.jksuci.2024.102048.

[4] M. Kumar, L. Khan, and H.-T. Chang, "Evolving techniques in sentiment analysis: A comprehensive review," *PeerJ Computer Science*, vol. 11, Jan. 2025, Art. no. e2592, doi: 10.7717/peerj-cs.2592.

[5] L. P. Mudarakola, R. K. Gatla, A. S. N. Raju, A. Y. Jaffar, A. Alzahrani, and A. A. Dessalegn, "Multi stage sentiment analysis for product reviews on Twitter using optimized machine learning algorithm," *Scientific Reports*, vol. 15, no. 1, Art. no. 39777, Nov. 2025, doi: 10.1038/s41598-025-23451-8.

[6] R. K. Mohapatra, L. Jolly, and S. P. Dakua, "From raw to refined: The influence of preprocessing on ML performance for liver disease detection," *Discover Artificial Intelligence*, vol. 6, no. 1, Art. no. 27, Dec. 2025, doi: 10.1007/s44163-025-00659-1.

[7] P. S. Ghatora, S. E. Hosseini, S. Pervez, M. J. Iqbal, and N. Shaukat, "Sentiment analysis of product reviews using machine learning and pre-trained LLM," *Big Data and Cognitive Computing*, vol. 8, no. 12, Art. no. 199, 2024, doi: 10.3390/bdcc8120199.

[8] K. Taha, P. D. Yoo, C. Yeun, D. Homouz, and A. Taha, "A comprehensive survey of text classification techniques and their research applications: Observational and experimental insights," *Computer Science Review*, vol. 54, Nov. 2024, Art. no. 100664, doi: 10.1016/j.cosrev.2024.100664.

[9] R. Anadra, H. Wijayanto, and K. Sadik, "Sentiment analysis of Tokopedia customer reviews using BiLSTM and IndoBERT with comparative analysis of preprocessing and labeling methods," *International Journal of Advances in Data and Information Systems*, vol. 6, no. 3, pp. 773–788, Dec. 2025, doi: 10.59395/ijadis.v6i3.1458.

[10] M. Maree, M. Eleyat, and E. Mesqali, "Optimizing machine learning-based sentiment analysis accuracy in bilingual sentences via preprocessing techniques," *The International Arab Journal of Information Technology*, vol. 21, no. 2, pp. 257–270, Mar. 2024, doi: 10.34028/iajit/21/2/8.

[11] M. F. Alam, A. Nuryaman, P. H. Khotimah, A. Parlina, and A. Sihombing, "Optimizing Multi-Layer Perceptron Performance in

Sentiment Classification through Neural Network Feature Extraction," *Baca. j. dok. inf.*, vol. 46, no. 1, pp. 1–14, Jul. 2025, doi: 10.55981/baca.2025.8240.
[12] J. R. Jim, M. A. R. Talukder, P. Malakar, M. M. Kabir, K. Nur, and M. F. Mridha, "Recent advancements and challenges of NLP-based sentiment analysis: A state-of-the-art review," *Natural Language Processing Journal*, vol. 6, Mar. 2024, Art. no. 100059, doi: 10.1016/j.nlp.2024.100059.
[13] Y. S. Mehanna and M. Mahmuddin, "The effect of pre-processing techniques on the accuracy of sentiment analysis using bag-of-concepts text representation," *SN Computer Science*, vol. 2, no. 4, Apr. 2021, Art. no. 237, doi: 10.1007/s42979-021-00453-7.
[14] S. Samsudin and L. Putriani, "Optimal preprocessing and feature engineering combination for DANA app sentiment classification: A full factorial study," *Scientific Journal of Informatics*, vol. 13, no. 4, pp. 767–782, Nov. 2026, doi: 10.15294/sji.v13i4.58535.
[15] A. Kurniasih and L. P. Manik, "On the role of text preprocessing in BERT embedding-based DNNs for classifying informal texts," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 6, pp. 927–934, 2022, doi: 10.14569/IJACSA.2022.01306109.
[16] K. Alemerien, A. Al-Ghareeb, and M. Z. Alksasbeh, "Sentiment Analysis of Online Reviews: A Machine Learning Based Approach with TF-IDF Vectorization," *Journal of Mobile Multimedia*, vol. 20, no. 5, pp. 1089–1116, 2024, doi: 10.13052/jmm1550-4646.2055.
[17] K. P. Harmandini and K. M. L., "Analysis of TF-IDF and TF-RF Feature Extraction on Product Review Sentiment," *Sinkron: Jurnal dan Penelitian Teknik Informatika*, vol. 8, no. 2, pp. 929–937, Apr. 2024, doi: 10.33395/sinkron.v8i2.13376.
[18] H. Liu, X. Chen, and X. Liu, "A Study of the Application of Weight Distributing Method Combining Sentiment Dictionary and TF-IDF for Text Sentiment Analysis," *IEEE Access*, vol. 10, pp. 32280–32289, 2022, doi: 10.1109/ACCESS.2022.3160172.
[19] R. Mulyawan, H. Naparin, and W. M. Fatihia, "Comparison of text vectorization methods for IMDB movie review sentiment analysis using SVM," *Journal of Applied Informatics and Computing*, vol. 9, no. 5, pp. 2270–2277, Oct. 2025, doi: 10.30871/jaic.v9i5.10372.
[20] C. A. Nurhaliza Agustina, R. Novita, Mustakim, and N. E. Rozanda, "The implementation of TF-IDF and Word2Vec on booster vaccine sentiment analysis using support vector machine algorithm," *Procedia Computer Science*, vol. 234, pp. 156–163, Jan. 2024, doi: 10.1016/j.procs.2024.02.162.
[21] M. B. Kurniawan, Hanafi, R. Hikmianto, and I. Muslihah, "Hyperparameter optimization of TF-IDF and SVM via grid search for sentiment analysis of Traveloka customer reviews," *Khazanah Informatika: Jurnal Ilmu Komputer dan Informatika*, vol. 11, no. 2, pp. 68–76, Jun. 2026, doi: 10.23917/khif.v11i2.4784.
[22] A. Romadhony, S. al Faraby, R. Rismala, U. N. Wisesti, and A. Arifianto, "Sentiment Analysis on a Large Indonesian Product Review Dataset," *Journal of Information Systems Engineering and Business Intelligence*, vol. 10, no. 1, pp. 167–178, 2024, doi: 10.20473/jisebi.10.1.167-178.
[23] B. Erkantarci and G. Bakal, "An empirical study of sentiment analysis utilizing machine learning and deep learning algorithms," *J Comput Soc Sci*, vol. 7, no. 1, pp. 241–257, Apr. 2024, doi: 10.1007/s42001-023-00236-5.
[24] A. Riswawan, "Implementing TF-IDF and Logistic Regression for Sentiment Analysis of YouTube Comments on the iPhone 16," *Jurnal Teknologi dan Open Source*, vol. 8, no. 2, pp. 604–611, Dec. 2025, doi: 10.36378/jtos.v7i2.4753.
[25] A. Arasy, S. Agustian, L. Handayani, and I. Iskandar, "Klasifikasi Sentimen Menggunakan Metode Multilayer Perceptron dengan Fitur TF-IDF," *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 5, no. 3, pp. 908–919, Jun. 2025, doi: 10.57152/malcom.v5i3.2052.
[26] R. Sutoyo, S. Achmad, A. Chowanda, E. W. Andangsari, and S. M. Isa, "PRDECT-ID: Indonesian product reviews dataset for emotions classification tasks," *Data in Brief*, vol. 44, Oct. 2022, Art. no. 108554, doi: 10.1016/j.dib.2022.108554.
[27] R. Rianto, A. B. Mutiara, E. P. Wibowo, and P. I. Santosa, "Improving the accuracy of text classification using stemming method, a case of non-formal Indonesian conversation," *Journal of Big Data*, vol. 8, no. 1, p. 26, Jan. 2021, doi: 10.1186/s40537-021-00413-1.
[28] M. Siino, I. Tinnirello, and M. la Cascia, "Is text preprocessing still worth the time? A comparative survey on the influence of popular preprocessing methods on Transformers and traditional classifiers," *Inf Syst*, vol. 121, Mar. 2024, doi: 10.1016/j.is.2023.102342.
[29] J. Khan, K. Ahmad, S. K. Jagatheesaperumal, and K.-A. Sohn, "Textual variations in social media text processing applications: Challenges, solutions, and trends," *Artificial Intelligence Review*, vol. 58, no. 3, Mar. 2025, Art. no. 89, doi: 10.1007/s10462-024-11071-z.
[30] A. S. Bashir, A. A. Bichi, and A. Adamu, "Automatic construction of generic Hausa language stop words list using term frequency-inverse document frequency," *Journal of Electrical Systems and Information Technology*, vol. 11, no. 1, Art. no. 58, Dec. 2024, doi: 10.1186/s43067-024-00187-5.
[31] M. O. Ibrohim and I. Budi, "Multi-label hate speech and abusive language detection in Indonesian Twitter," in *Proceedings of the Third Workshop on Abusive Language Online*, Florence, Italy, Aug. 2019, pp. 46–57, doi: 10.18653/v1/W19-3506.
[32] M. Mujahid et al., "Data oversampling and imbalanced datasets: An investigation of performance for machine learning and feature engineering," *Journal of Big Data*, vol. 11, no. 1, Jun. 2024, Art. no. 87, doi: 10.1186/s40537-024-00943-4.
[33] E. Fikadu, T. Urgessa, and M. Das, "MLP-SVM: a hybrid approach for improving the performance of the classification model for health-related documents from social media using multi-layer perceptron and support vector machine," *Discover Applied Sciences*, vol. 7, no. 4, p. 320, Apr. 2025, doi: 10.1007/s42452-025-06851-3.
[34] C. Nadeau and Y. Bengio, "Inference for the Generalization Error," *Machine Learning*, vol. 52, pp. 239–281, 2003, doi: 10.1023/A:1024068626366.
[35] S. Holm, "A Simple Sequentially Rejective Multiple Test Procedure," *Scandinavian Journal of Statistics*, vol. 6, no. 2, pp. 65–70, 1979.