Translation of article 10.22146/ jnteti.v14i2.17117

Semantic Similarity Measurement Evaluation of KBBI Synonyms Using a Word Embedding Approach

Muhammad Rafli Aditya H.[1], Muhammad Ilham[1], Dewi Fatmarani Surianto[1], Abdul Muis Mappalotteng[2]

[1] Departemen Teknik Informatika dan Komputer, Fakultas Teknik, Universitas Negeri Makassar, Makassar, Sulawesi Selatan 90224, Indonesia
[2] Departemen Pendidikan Teknik Elektro, Fakultas Teknik, Universitas Negeri Makassar, Makassar, Sulawesi Selatan 90224, Indonesia

[Submitted: 7 November 2024, Revised: 9 February 2025, Accepted: 16 April 2025]
Corresponding Author: Dewi Fatmarani Surianto (email: dewifatmaranis@unm.ac.id)

ABSTRACT — Kamus Besar Bahasa Indonesia (KBBI) is a primary resource for data in research on determining word-meaning similarity in Indonesian. This study investigates the effectiveness of word embedding methods and the term frequency–inverse document frequency (TF-IDF) weighting technique in assessing the semantic similarity of synonym pairs. The objective is to measure the similarity of synonym word pairs listed in KBBI by applying cosine similarity, leveraging TF-IDF weighting, various word embedding models, and latent semantic analysis (LSA). The methodology involved data collection, followed by a text preprocessing stage consisting of case folding, stopword removal, stemming, and tokenization. The processed data were transformed into vector representations using word embedding models, including Word2Vec, fastText, GloVe, and sentence-bidirectional encoder representations from transformers (S-BERT), and TF-IDF. LSA was employed for dimensionality reduction of the vectors before similarity testing using cosine similarity, with final evaluation of the results. The findings revealed that fastText significantly improved the similarity scores between synonym pairs, achieving an average similarity score of 0.901 for 30 synonym pairs. Evaluation results indicated an accuracy of 0.88, a recall of 1.00, a precision of 0.81, and an F1 score of 0.90. These results suggest that fastText is more effective in enhancing the accuracy of synonym meaning similarity measurements. Future research is encouraged to expand the corpus and further explore the use of word embedding for semantic similarity tasks. This study contributes to the natural language processing advancement and provides a potential foundation for more accurate language-based applications that assess word meaning similarity in KBBI.

KEYWORDS — KBBI, Word2Vec, Cosine Similarity, FastText, GloVe, Sentence-BERT, Semantic Similarity Measurement.

I. INTRODUCTION

The Kamus Besar Bahasa Indonesia (KBBI) is one of the primary resources for determining semantic similarity between words in the Indonesian language. Semantic similarity refers to the condition in which two or more words are not identical but share the same or closely related meanings [1]. Such words are known as synonyms, as they exhibit semantic similarity across various word forms [2]. Recognizing similarity between words constitutes an essential initial step in assessing similarity at higher levels of textual structure, such as sentences, paragraphs, and documents. Similarity is defined as the degree of resemblance between two text segments [3]. To measure semantic proximity between sentences, inter-sentence similarity must be calculated [4]. Semantic similarity is a linguistic concept that refers to situations in which two words differ phonetically or morphologically but possess identical or nearly equivalent meanings.

Natural language processing (NLP) is a subfield of computer science and artificial intelligence that focuses on the interaction between natural human languages and computers [5]. Within NLP, measuring semantic similarity is a fundamental task that underpins various applications. For example, in machine translation, accurate translation requires systems capable of understanding phrases and synonyms that differ in form but share meaning. Similarly, in sentiment analysis, recognizing semantic similarity enables systems to comprehend the various lexical and phrasal expressions of the same emotional content, thereby enhancing the relevance and accuracy of NLP-based applications [5].

This study investigates how the word embedding method and the term frequency-inverse document frequency (TF-IDF) weighting technique can be utilized to evaluate the semantic similarity of synonym pairs in Indonesian. The data used for this study were derived from the definitions of selected words in the KBBI. Therefore, the primary focus of this research is on computing the semantic similarity between words that are synonymous.

Numerous computational approaches have been developed for measuring word similarity, ranging from rule-based systems to machine learning techniques. In this study, cosine similarity was employed as the primary technique for computing the similarity between vectors representing words via embedding models [6]. Cosine similarity is widely used in text similarity tasks, including sentence-level similarity assessments [7]. The word embedding models used in this research included Word2Vec, fastText, GloVe, and sentence-bidirectional encoder representations from transformers (S-BERT), all of which process unstructured textual data by taking a collection of words as input and producing corresponding word vectors as output.

Word2Vec is an NLP algorithm designed to generate vector representations of words based on textual data [8]. Developed by Mikolov et al. in 2013, this model uses a neural network with a hidden layer to convert words into dense vector representations. One of its main advantages is its ability to handle synonyms and homonyms more effectively than traditional sparse representations [9], [10].

FastText is a predictive word embedding method that extracts features from words in the form of real-valued vectors, building upon the continuous bag of words (CBOW) approach [11]. FastText offers rapid training capabilities on large datasets and supports out-of-vocabulary word representations by decomposing words into n-grams to generate embeddings for previously unseen terms [9].

FastText is a method for extracting features from words in the form of real numbers using the concept of prediction-based word embedding. This method is a development of the CBOW technique [11]. The advantage of fastText lies in its ability to quickly train models on large datasets and provide representations for words that are not present in the training data. If a word does not appear during training, it can be split into n-grams to generate its embedding vector [9].

In contrast to Word2Vec and fastText, GloVe is a count-based model that captures semantic relationships between words by computing co-occurrence frequencies in a corpus. This model employs global matrix factorization to encode broader semantic relationships than neural-network-based models like Word2Vec [9]. Meanwhile, S-BERT is a modification of BERT that incorporates Siamese and triplet network architectures to efficiently compare semantic similarity between sentences. Through pooling techniques, S-BERT can generate fixed-size sentence embeddings, making it more efficient than BERT in similar sentence retrieval tasks [12].

Additionally, this study applied latent semantic analysis (LSA) to reduce the dimensionality of the word vectors generated by the embedding models. LSA is a statistical method used to identify and represent the semantic similarity between words and texts by analyzing large-scale textual data [13]. This approach is expected to improve the accuracy of semantic similarity measurements based on the KBBI.

Several prior studies have examined similar topics. For instance, semantic similarity between sentences based on KBBI was evaluated using latent semantic indexing (LSI), yielding accuracy scores of 75.9% with traditional TF-IDF and 80% when combining LSI and TF-IDF [14]. Another study applied a preprocessing technique using number-to-word conversion and cosine similarity to test sentence similarity in Indonesian, reporting that 12 out of 13 tests (92.30%) showed improved similarity scores compared to those without such preprocessing [15]. Furthermore, a separate study assessed word-level semantic similarity using cosine similarity and a synonym database, finding that 24 out of 25 test cases yielded improved similarity scores. The average similarity value using ID-based vectors reached 94.48%, while the comparison method yielded 69.96% [16]. These studies primarily employed cosine similarity and LSI, but did not utilize more recent embedding techniques such as Word2Vec, fastText, GloVe, S-BERT, or dimensionality reduction via LSA.

Other studies have explored Word2Vec applications in various contexts. For example, Word2Vec has been used in topic classification for Twitter data using random forest classifiers and feature expansion based on Word2Vec. This study tested three different corpora (tweets, news articles, and a combination of both) and three feature expansion levels (Top 1, Top 5, Top 10), with the best model achieving an accuracy of 99.49% using Top 5 expansion [17]. Another study applied Word2Vec in sentiment analysis of movie reviews using long short-term memory (LSTM) deep learning models. Results showed an accuracy of 88.17% with 100-dimensional word vectors and a minimum accuracy of 85.86% with 500-dimensional vectors [18]. Additionally, Word2Vec has been used for sentiment analysis in online transportation services, with Gojek achieving 87% accuracy, 93% precision, and 84% recall, while Grab achieved 82% accuracy, 89% precision, and 83% recall using support vector machine (SVM) algorithms [19]. A different approach employed Word2Vec to recommend cross-language songs based on lyrics, identifying ten songs with high lyrical similarity to a given input song. TF-IDF-based and nearest-neighbor methods yielded the highest average precision-at-10 scores [20]. Although these studies applied Word2Vec, they did so in different contexts and with different research objectives.

Prior studies have investigated semantic similarity measurement in Indonesian using various approaches. Some of them use rule-based approaches, cosine similarity techniques, and LSI in the context of measuring semantic similarity. Although these methods have their own advantages, previous research has not integrated various word embedding techniques, such as Word2Vec, fastText, GloVe, and S-BERT in one framework to see a more comprehensive performance comparison.

Moreover, although word embedding techniques have seen significant development in recent years, most studies have focused on other languages or different applications, such as sentiment analysis or text classification. The application of word embedding techniques specifically for measuring word-level semantic similarity in Indonesian remains relatively unexplored, despite the language's unique linguistic characteristics [21].

To address this research gap, the present study contributes by integrating multiple word embedding techniques (Word2Vec, fastText, GloVe, and S-BERT) into a unified framework for evaluating their effectiveness in measuring synonym-based semantic similarity in Indonesian. Additionally, LSA was employed to reduce vector dimensionality, thereby enhancing performance in semantic similarity computations. The study also introduced a KBBI-based dataset consisting of synonym pairs to serve as a benchmark for evaluating the tested approaches. For evaluation purposes, performance metrics such as accuracy, precision, recall, and F1 score were employed to provide a quantitative analysis of each method's effectiveness. This research aims to support the advancement of Indonesian NLP, particularly in the context of word similarity measurement for applications such as text analysis.

II. METHODOLOGY

This study comprised seven main stages: data collection, text preprocessing, term weighting, application of word embedding models, dimensionality reduction using LSA, cosine similarity evaluation, and performance assessment of similarity tests.

A. DATA COLLECTION

The dataset utilized in this study consisted of 300 synonym word pairs obtained from the official KBBI website, maintained by the Ministry of Education and Culture. These pairs served as the basis for evaluating semantic similarity using two main approaches: word embedding and the TF-IDF weighting technique. The choice of 300 pairs was driven by considerations of efficiency [22], aiming to provide sufficient representation for evaluating the Word2Vec model's capability in measuring semantic similarity, while simultaneously

maintaining tractable computational complexity in terms of processing time and resource consumption. The full dataset is presented in Table I.

B. TEXT PREPROCESSING

Text preprocessing involves a series of steps applied to textual data prior to similarity analysis, aiming to clean and standardize the data for subsequent processing stages. In this study, the preprocessing pipeline consisted of several procedures [23], including case folding, which converts all characters to lowercase to address inconsistencies caused by irregular capitalization [24]; stopword removal, which eliminates non-informative words from the text [25]; stemming, which reduces words to their base or root form [26]; and tokenization, which segments sentences into individual units or tokens [27]. These preprocessing steps ensure that the textual input is clean, normalized, and suitable for vectorization.

C. WORD WEIGHTING

TF-IDF is employed to assign importance scores to each term in the dataset. The TF-IDF algorithm quantifies a word's relevance by considering how frequently it appears in a specific document relative to its frequency across all documents [28]. In this study, the TfidfVectorizer library was used for implementation. The IDF formula is defined as in (1).

$$IDF = Log(\frac{n}{df_i}). \quad (1)$$

where n denotes the total number of sentences, an df_i represents the number of documents in which the term appears.

D. WORD EMBEDDING

Word embedding is a term that refers to language modeling and feature learning techniques in NLP. Each word in the vocabulary is represented with a vector that describes its meaning. The words are mapped into a vector of real numbers [29]. Word embedding used in this research were Word2Vec, fastText, GloVe, and S-BERT.

The Word2Vec developed in this study was trained using a dataset containing a collection of definitions of 300 words in KBBI and using the CBOW architecture, as illustrated in Figure 1. CBOW is a model used to generate word vector representations. From the CBOW architecture in Figure 1, it appears that the input layer contains context words taken from surrounding target word to be predicted, the hidden layer contains a vector of context words that are summed or averaged to form a vector representation, the hidden layer aggregates their vector representations—either by summation or averaging—and the output layer produces a prediction score by applying the context vector to a weight matrix.

The fastText model used was a pretrained model for the Indonesian language named cc.id.300.bin, trained on large-scale corpora such as Common Crawl and Wikipedia. It also used the CBOW architecture with a vector dimensionality of 300, a context window size of 5, 5-character n-grams, and 10 negative samples. The inclusion of character-level n-grams enables the model to capture subword information and morphological features, which enhances its ability to represent rare or previously unseen words in the training corpus.

The GloVe model implemented in this study was "glove_50dim_wiki.id.case.text," which was pretrained on Indonesian Wikipedia and generated word embeddings with 50 dimensions. GloVe captures semantic relationships between words based on global word co-occurrence statistics within a large corpus. The GloVe model's architecture is depicted in Figure 2.

TABLE I
DATASET OF SYNONYM WORD DEFINITION

Word	Definition
abrasi	*pengikisan batuan oleh air; es; atau angin yg mengandung dan mengangkut hancuran bahan; luka lecet atau jejas karena pengikisan kulit oleh benda kasar; pengikisan selaput lendir (misalnya dalam membersihkan rahim)*
pengikisan	*Proses cara perbuatan mengikis; situasi saling mengikis; erosi*
ayah	*Orang tua kandung laki-laki; bapak; panggilan kepada orang tua kandung laki-laki.*
...	...
...	...
pengabdian	*Proses; cara; perbuatan mengabdi atau mengabdikan*
dialog	*Percakapan (dalam sandiwara; cerita; dan sebagainya): karya tulis yang disajikan dalam bentuk percakapan antara dua tokoh atau lebih*
obrolan	*Percakapan ringan dan santai; omong kosong*

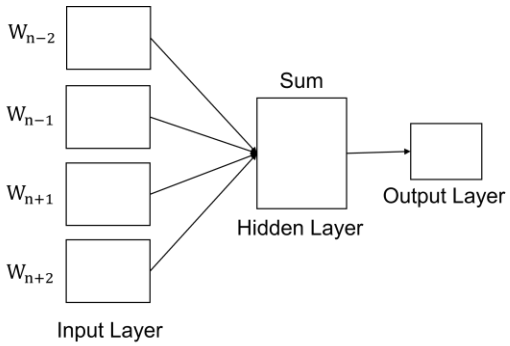

Figure 1. Architecture of continuous bag of words.

S-BERT in this study used a pretrained model named "distiluse base-multilingual-cased-v2," which has been trained for multiple languages, thus supporting up to 50 languages, such as Korean, Arabic, German, and Russian, including Indonesian. The model is designed to generate vector representations that can capture the semantic meaning of sentences in multiple languages. It produces vectors with a length of 512 dimensions. The S-BERT architecture is shown in Figure 3.

In the architecture of the S-BERT model in Figure 3, Sentence A and Sentence B are the two sentences used as input, BERT is the stage when the sentences are put into two identical BERT models and generate a representation vector for each token, mean pooling is the stage to get one sentence representation vector (u for Sentence A and v for Sentence B), and finally the similarity between the two sentences was measured by cosine similarity. The application of Word2Vec, fastText, GloVe, and S-BERT in this research was used to generate document vectors from two specific documents ("doc1" and "doc2"), which are the meaning of synonymous words and measure the cosine similarity between the two document vectors.

E. LATENT SEMANTIC ANALYSIS

LSA was employed to reduce the dimensionality of vector representations generated by word embedding models such as

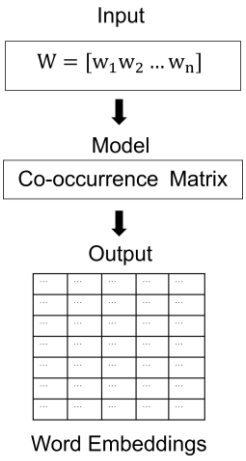

Figure 2. GloVe model's architecture.

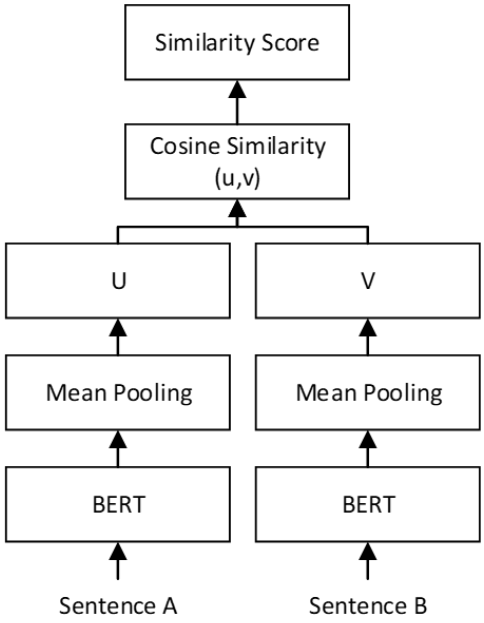

Figure 3. S-BERT model's architecture.

fastText, GloVe, and S-BERT, utilizing the TruncatedSVD function from the sklearn. This technique decomposes a high-dimensional data matrix into a lower-dimensional form while preserving essential information. In this study, the vectors produced by fastText, GloVe, and S-BERT were reduced to 25 dimensions to enhance the efficiency of semantic similarity computations. Word2Vec vectors were excluded from this dimensionality reduction process, as the model was already trained using a suitably optimized dimensional size.

Dimensionality reduction via LSA addresses data sparsity by filtering more relevant information, resulting in more compact and semantically coherent word representations. Moreover, LSA enables the identification of latent relationships among words, thereby allowing infrequently occurring words to remain semantically associated with more common terms.

F. SIMILARITY MEASUREMENT

Cosine similarity was adopted to measure the degree of similarity between document vectors. This method assesses the angle between two vectors in a multidimensional space, based on the cosine of the angle formed by the dot product of the vectors. Since the cosine of 0 degrees is 1, and the cosine of larger angles is less than 1, a similarity score closer to 1 indicates a higher degree of similarity between the two vectors [30]. In this study, cosine similarity was used to evaluate the semantic similarity of synonymous word pairs derived from both word embedding and TF-IDF representations.

The formula for cosine similarity is as follows:

$$\cos(\theta) = \frac{A.B}{\|A\|\,\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2 \times \sum_{i=1}^{n}(B_i)^2}}. \quad (2)$$

In (2), A and B represent the vectors being compared in the similarity test. $A.B$ denotes the dot product between vectors A and B, $|A|$ and $|B|$ refer to the length of the vectors, and $|A||B|$ refers to the cross product between $|A|$ and $|B|$.

G. EVALUATION

The similarity scores obtained via cosine similarity were evaluated using four standard performance metrics: accuracy, precision, recall, and F1 score, based on 60 samples. These consisted of 30 synonym word pairs and 30 non-synonym word pairs.

In this study, a similarity threshold of 0.7 was applied to classify word pairs as synonymous or not. The similarity values obtained were converted into binary labels: if the similarity score was greater than or equal to 0.7, the word pair was labeled as synonymous (label 1); otherwise, it was classified as non-synonymous (label 0).

To compare predicted results with the reference data (ground truth), a confusion matrix was utilized. The ground truth comprised 30 instances labeled as 1 (synonymous pairs) and 30 labeled as 0 (non-synonymous pairs). The model's ability to classify synonym pairs was evaluated by comparing its predictions against the ground truth using the confusion matrix, enabling the calculation of accuracy, recall, precision, and F1 score values.

In Figure 4, there are several terms in the confusion matrix. True negative (TN) occurs when the model predicts data in the negative class and the data are actually in the negative class, true positive (TP) occurs when the model predicts data in the positive class and the data are actually in the positive class, false negative (FN) occurs when the model predicts data in the negative class, but the data are actually in the positive class, while false positive (FP) occurs when the model predicts data in the positive class, but the data are actually in the negative class.

1) ACCURACY

Accuracy is the ratio of correct predictions (both positive and negative) to the total number of predictions. It measures the overall accuracy of the model in performing predictions, calculated using (3):

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN}. \quad (3)$$

2) PRECISION

Precision is the ratio of true positives to the total number of instances predicted as positive, calculated using (4):

$$precision = \frac{TP}{TP+FP}. \quad (4)$$

3) RECALL

Recall is the ratio of true positives to the total number of actual positive instances, calculated using (5):

$$recall = \frac{TP}{TP+FN}. \quad (5)$$

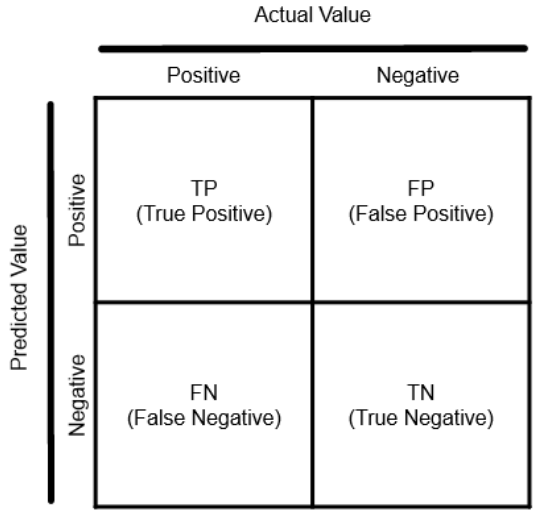

Figure 4. Confusion matrix.

4) F1 SCORE

The F1 score represents a value that reflects the combination of recall and precision. The formula to compute the F1 score is presented in (6).

$$F1\ score\ = 2\ \times \frac{precision \times recall}{precision+recall}. \quad (6)$$

III. RESULTS AND DISCUSSION

This study commenced by performing text preprocessing on pairs of synonymous word meanings using Jupyter Notebook. The first experiment employed the TfidfVectorizer library, while the second utilized Word2Vec in conjunction with cosine similarity. The results of both experiments were compared to determine which method yielded the highest similarity score.

Prior to calculating the TF-IDF values and cosine similarity, a series of preprocessing steps was conducted. This process began with importing and initializing the necessary libraries and datasets. The dataset used for the similarity test consisted of 30 pairs of synonymous word meanings, including: "*ayah*" and "*bapak,*" "*ekonomis*" and "*hemat,*" "*datuk*" and "*kakek,*" "*kakak*" and "*abang,*" "*tabiat*" and "*watak,*" "*pelestarian*" and "*konservasi,*" "*giat*" and "*rajin,*" "*rumah*" and "*hunian,*" "*seniman*" and "*artis,*" "*guru*" and "*pengajar,*" "*pelajar*" and "*siswa,*" "*instruksi*" and "*arahan,*" "*bakat*" and "*talenta,*" "*penelitian*" and "*riset,*" "*ahli*" and "*pakar,*" "*berdikari*" and "*mandiri,*" "*berhasil*" and "*sukses,*" "*topan*" and "*siklon,*" "*swatantra*" and "*otonomi,*" "*flora*" and "*tumbuhan,*" "*kebisaan*" and "*kepandaian,*" "*vandalisme*" and "*destruksi,*" "*advokat*" and "*pengacara,*" "*wabah*" and "*epidemi,*" "*mampu*" and "*bisa,*" "*laris*" and "*laku,*" "*uzur*" and "*halangan,*" "*dedikasi*" and "*pengabdian,*" "*keahlian*" and "*kepandaian,*" "*dokter*" and "*tabib.*" These synonymous word pairs were selected based on their frequent usage in daily communication, which made them familiar to the general public and suitable for evaluating semantic similarity. The definitions of the synonyms were sourced from the KBBI. An example is presented below:

ayah = *orang tua kandung laki-laki; bapak; panggilan kepada orang tua kandung laki-laki*

bapak = *orang tua laki-laki; ayah; orang laki-laki yang dalam pertalian kekeluargaan boleh dianggap sama dengan ayah (seperti saudara laki-laki ibu atau saudara laki-laki bapak); orang yang dipandang sebagai orang tua atau orang yang dihormati (seperti guru kepala kampung); panggilan kepada orang laki-laki yang lebih tua dari yang memanggil; orang yang menjadi pelindung (pemimpin, perintis jalan, dan sebagainya yang banyak penganutnya); pejabat*

The text preprocessing consisted of the following stages. First, case folding was applied to standardize all text to lowercase characters, as shown below:

ayah = *orang tua kandung laki-laki; bapak; panggilan kepada orang tua kandung laki-laki*

bapak = *orang tua laki-laki; ayah; orang laki-laki yang dalam pertalian kekeluargaan boleh dianggap sama dengan ayah (seperti saudara laki-laki ibu atau saudara laki-laki bapak); orang yang dipandang sebagai orang tua atau orang yang dihormati (seperti guru kepala kampung); panggilan kepada orang laki-laki yang lebih tua dari yang memanggil; orang yang menjadi pelindung (pemimpin perintis jalan dan sebagainya yang banyak penganutnya); pejabat*

Second, stopword removal was performed to eliminate nondescriptive or uninformative words, resulting in the following cleaned text:

ayah = *orang tua kandung laki-laki; bapak; panggilan orang tua kandung laki-laki*

bapak = *orang tua laki-laki; ayah; orang laki-laki pertalian kekeluargaan dianggap ayah (seperti saudara laki-laki saudara laki-laki bapak); orang dipandang orang tua orang dihormati (seperti guru kepala kampung); panggilan orang laki-laki tua memanggil; orang pelindung (pemimpin perintis jalan penganutnya); pejabat*

Third, stemming was applied to remove affixes and convert all words into their root forms, producing the following results.

ayah = *orang tua kandung laki bapak panggil orang tua kandung laki*

bapak = *orang tua laki ayah orang laki tali keluarga anggap ayah seperti saudara laki saudara laki bapak orang pandang orang tua orang hormat seperti guru kepala kampung panggil orang laki tua panggil orang lindung pimpin rintis jalan anut jabat*

Upon completion of the preprocessing stage, the first experiment was conducted using TF-IDF and cosine similarity to assess the semantic similarity between each pair of synonymous meanings. The results of this experiment are presented in Table II.

The first pair of synonymous words yielded a cosine similarity score of 0.59 using the TF-IDF implementation. Subsequently, a second experiment was conducted to evaluate the semantic similarity between two synonymous words using the Word2Vec model. The Word2Vec model was built with several parameters, as following. The preprocessed_data referred to data containing the meanings of synonymous words that had undergone prior text preprocessing, which were then

TABLE II
RESULTS OF TF-IDF

Word	TF-IDF Vector	
	Ayah	***Bapak***
anggap	0.00	0.11
anut	0.00	0.11
ayah	0.00	0.23
bapak	0.21	0.08
guru	0.00	0.11
kandung	0.60	0.00
hormat	0.00	0.11
jabat	0.00	0.11
jalan	0.00	0.11
kampung	0.00	0.11
lindung	0.00	0.11
kepala	0.00	0.11
pimpin	0.00	0.11
rintis	0.00	0.11
pandang	0.00	0.11
keluarga	0.00	0.11
laki	0.42	0.42
orang	0.42	0.59
panggil	0.21	0.16
saudara	0.00	0.23
seperti	0.00	0.23
tali	0.00	0.11
tua	0.42	0.25

used to train the model. The vector size indicated the number of dimensions in the word vectors generated by the model; in this study, various vector sizes were used, including 10, 25, 50, 75, and 100. The window size represented the number of preceding and succeeding words considered by the model when training word vectors; in this study, a window size of 4 was used, meaning that four words before and after the target word were considered in predicting the target or training the word vector. The parameter min count = 1 indicated that the minimum number of occurrences for a word to be included in the model was one. The parameter workers = 4 specified the number of threads used for training. The skip-gram (SG) parameter determined whether the Word2Vec model would use the SG or CBOW architecture; this study used CBOW, so the SG value was set to 0, as setting SG to 1 would activate the SG model.

CBOW was chosen for its computational efficiency, particularly when processing large and diverse datasets. In Word2Vec, CBOW was used to predict the target word based on the surrounding context, allowing it to be trained more quickly than the SG model. This approach was beneficial for the word similarity task, especially given that the dataset in this study included 300 entries of synonymous word meanings. The experiments were conducted multiple times with different vector sizes to identify the highest similarity score for each size. Several resulting similarity values are shown in Table III.

Subsequent experiments were conducted using several other word embedding models, including fastText, GloVe, and S-BERT, in combination with LSA to reduce the vector dimensionality to 25 dimensions. These reduced vectors were then used in cosine similarity testing. The results of the similarity tests are presented in Table IV.

This experimental scheme was further applied to 29 additional synonym word pairs, based on the definitions of each

TABLE III
WORD2VEC TEST RESULTS WITH COSINE SIMILARITY OF SYNONYMS OF THE WORDS *AYAH/BAPAK*

Vector Size	Similarity Test Results
10	0.94
25	0.85
50	0.88
75	0.90
100	0.86

TABLE IV
TEST RESULTS OF WORD EMBEDDING WITH LSA AND COSINE SIMILARITY ON THE WORDS *AYAH* AND *BAPAK*

Word	Similarity Test Results		
	FastText	***GloVe***	***S-BERT***
ayah and *bapak*	0.97	0.97	0.69

word as provided by the KBBI, as shown in Table V. The average similarity test results using Word2Vec and cosine similarity on 30 synonym word pairs, with varying vector sizes, are summarized in Table VI.

The experiments were conducted on 30 pairs of synonymous words, yielding different average similarity scores for each vector size. A vector dimension of 10 yielded an average similarity score of 0.484; dimension 25 yielded 0.504; dimension 50 yielded 0.505; dimension 75 yielded 0.478; and dimension 100 yielded 0.495. The highest average score was achieved with a vector size of 50, with a value of 0.505. For comparison purposes, Table VII presents the average similarity scores obtained from TF-IDF weighting and word embedding methods for the same 30 synonym word pairs.

Based on the results obtained, it can be analyzed that the similarity evaluations using TF-IDF and various word embedding models yielded different scores. The similarity scores obtained from word embeddings were generally higher than those from TF-IDF. This difference arises from the distinct mechanisms by which these methods represent words and capture semantic meaning. Word embedding models such as Word2Vec, GloVe, and fastText learn word representations based on the context in which words appear within sentences. These models capture semantic relationships by embedding words in a vector space where semantically similar or synonymous words are represented by closely positioned vectors. In contrast, S-BERT is designed to generate sentence-level representations, providing a broader contextual understanding of meaning. On the other hand, TF-IDF relies solely on term frequency and inverse document frequency, without considering the semantic context in which words occur. As a result, TF-IDF often produces lower similarity scores compared to word embedding models, especially for synonym recognition. The average similarity results for the 30 pairs of synonymous words using word embedding techniques and TF-IDF are illustrated in Figure 5.

Based on the similarity results obtained using word embeddings and the TF-IDF weighting technique as shown in Figure 5, it can be concluded that fastText yielded the highest and most semantically accurate similarity scores. It consistently produced higher and more stable average similarity values across all tested synonymous word pairs compared to the vector representations generated by TF-IDF and other word embedding approaches. The average similarity score obtained using fastText was 0.901, while the score obtained using GloVe

TABLE V
DEFINITION OF 30 PAIRS OF SYNONYMOUS WORDS

Word	Definition
ekonomis	*bersifat hati-hati dalam pengeluaran uang; penggunaan barang; bahasa; waktu; tidak boros; hemat*
hemat	*berhati-hati dalam membelanjakan uang dan sebagainya; tidak boros; cermat*
...	...
...	...
dokter	*lulusan pendidikan kedokteran yang ahli dalam hal penyakit dan pengobatan*
tabib	*orang yang pekerjaannya mengobati orang sakit secara tradisional; seperti dukun; dokter*

TABLE VI
AVERAGE RESULTS OF WORD2VEC TESTING WITH COSINE SIMILARITY

Vector Size	Average Results of Similarity Test
10	0.484
25	0.504
50	0.505
75	0.478
100	0.495

TABLE VII
AVERAGE TEST RESULTS OF TF-IDF AND WORD EMBEDDING WITH LSA AND COSINE SIMILARITY ON 30 PAIRS OF WORD MEANINGS

Word	Average of Similarity Test Result			
	FastText	*GloVe*	*S-BERT*	*TF-IDF*
30 pairs of synonymous words (according to those in Table V)	0.901	0.866	0.697	0.302

was 0.866, indicating a difference of 0.035. The average score for S-BERT was 0.697, with a difference of 0.204; for Word2Vec, the highest average score was 0.505, with a difference of 0.396; and for TF-IDF, the average score was 0.302, with a difference of 0.599 compared to fastText.

However, these similarity scores alone are not sufficient to validate the effectiveness of the word embedding methods employed. Therefore, a further evaluation was conducted using four performance metrics: accuracy, recall, precision, and F1 score. This evaluation was carried out by comparing similarity scores across 60-word pairs, consisting of 30 synonymous and 30 non-synonymous pairs. The list of synonymous word pairs is presented in Table V, while the non-synonymous pairs were constructed randomly, such as "*ayah*" and *"abang," "kakak"* and *"bapak," "datuk"* and *"hemat," "ekonomis"* and *"kakek," "tabiat"* and *"konservasi," "pelestarian"* and *"watak," "giat"* and *"hunian," "rumah"* and *"rajin," "guru"* and *"artis," "seniman"* and *"pengajar," "pelajar"* and *"arahan," "instruksi"* and *"siswa," "bakat"* and *"riset," "penelitian"* and *"talenta," "ahli"* and *"mandiri," "berdikari"* and *"pakar," "berhasil"* and *"siklon," "topan"* and *"sukses," "swatantra"* and *"tumbuhan," "flora"* and *"otonomi," "kebisaan"* and *"destruksi," "vandalisme"* and *"kepandaian," "advokat"* and *"epidemi," "wabah" and "pengacara," "mampu" and "laku," "laris"* and *"bisa," "uzur"* and *"pengabdian," "dedikasi"* and *"halangan," "keahlian"* and *"tabib,"* and *"dokter"* and *"kepandaian."*

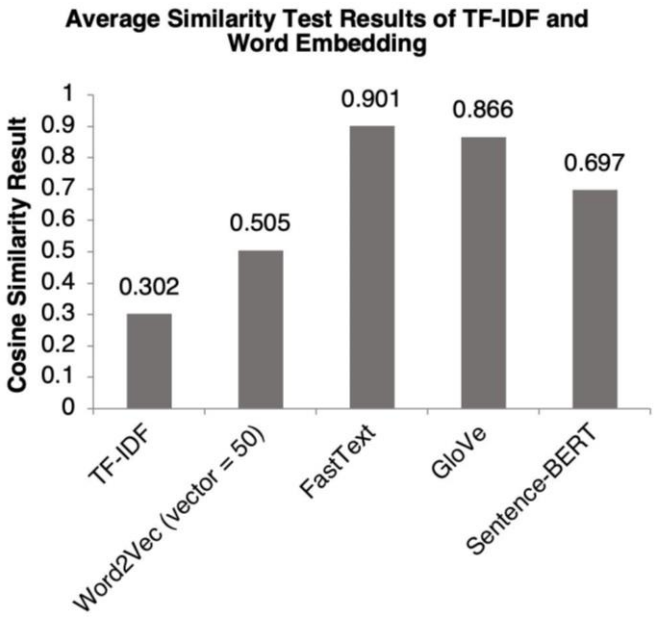

Figure 5. Graph of cosine similarity results with TF-IDF and word embedding in testing the semantic similarity of synonymous words.

The accuracy of the cosine similarity method was assessed by examining the model's ability to distinguish truly synonymous pairs from non-synonymous ones. If the method assigned high similarity scores to unrelated word pairs, its performance was considered suboptimal. Conversely, if cosine similarity consistently produced high similarity scores for truly synonymous words, the method was deemed effective.

The results obtained indicate that cosine similarity using TF-IDF yielded an accuracy of 0.50, with a precision, recall, and F1 score of 0.00. In contrast, other approaches produced more favorable outcomes: Word2Vec achieved an accuracy of 0.60, precision of 0.88, recall of 0.23, and F1 score of 0.37; fastText attained an accuracy of 0.88, precision of 0.81, recall of 1.00, and F1 score of 0.90; GloVe obtained an accuracy of 0.80, precision of 0.74, recall of 0.93, and F1 score of 0.82; and S-BERT yielded an accuracy of 0.77, precision of 1.00, recall of 0.53, and F1 score of 0.70. The complete results are presented in Table VIII.

These varying results are attributable to differences in how each method represents words and the nature of the datasets employed in this study. For example, Word2Vec represents words as vectors based on the local context surrounding each word, using the CBOW architecture to generate word vector representations. However, this method does not consider a broader global context, as it focuses only on the immediate surroundings of the target word. Additionally, the dataset used consisted of only 300 synonymous word meanings, which limits Word2Vec's capacity to capture inter-word relationships on a larger scale. GloVe, in contrast, is designed to capture word co-occurrence information across an entire corpus by constructing a co-occurrence matrix that records how often words appear together within a specific context. By leveraging global co-occurrence statistics, GloVe is better equipped to capture semantic relationships across documents. Nonetheless, GloVe is limited in handling rare or out-of-vocabulary words, as it does not incorporate subword structures—similar to Word2Vec. TF-IDF, on the other hand, only provides a vector representation of a simple calculation by counting the occurrence or frequency of each word in a sentence or

TABLE VIII
EVALUATION RESULTS OF TF-IDF AND WORD EMBEDDING

Method	Evaluation Results			
	Accuracy	*Recall*	*Precision*	*F1 Score*
TF-IDF	0.50	0.00	0.00	0.00
Word2Vec	0.60	0.23	0.88	0.37
FastText	0.88	1.00	0.81	0.90
GloVe	0.80	0.93	0.74	0.82
S-BERT	0.77	0.53	1.00	0.70

document. This method lacks the ability to capture inter-word relationships or account for the semantic meaning of words within broader contexts, often resulting in less accurate similarity scores.

Unlike other word embedding techniques, S-BERT is designed to produce sentence- or text-level vector representations using a Siamese network architecture, which outputs fixed-size vectors. One limitation of this method is its relatively high computational cost compared to lighter models such as Word2Vec or TF-IDF. Furthermore, S-BERT is not optimized for capturing the semantics of individual words, as its primary focus is on encoding the meaning of entire sentences or longer texts. Additionally, S-BERT does not consider subword structures.

On the other hand, the fastText approach is an extension of Word2Vec, offering the advantage of incorporating subword information (n-grams) in generating word representations. This enables fastText to capture morphological features and effectively handle rare or out-of-vocabulary words, thereby enhancing the model's ability to represent semantic relationships between words. This advantage allows fastText to generate richer word representations by capturing more semantic and morphological information, which explains its ability to consistently produce the highest and most stable average similarity scores compared to other word embedding methods.

This study contributes to advancing the understanding of NLP techniques in the context of the Indonesian language, particularly in the application of TF-IDF, Word2Vec, fastText, GloVe, and S-BERT. The findings of this research may serve as a foundation for further studies aimed at improving NLP methods tailored to the unique characteristics of the Indonesian language. Moreover, this study has the potential to enhance the performance of various NLP applications, such as sentiment analysis, text classification, and speech recognition.

The study also underscores the importance of considering local linguistic and cultural contexts in the development of NLP technologies. By examining how different models function in the Indonesian language, these findings can assist NLP practitioners and developers in selecting the most appropriate word embedding method for specific projects—for instance, in cases requiring low-resource solutions or applications demanding deeper semantic representation.

Another contribution of this study lies in its potential to improve Indonesian language-based NLP applications, such as increasing the accuracy of machine translation systems and context-based text analysis. Furthermore, this research supports the development of more efficient and contextually relevant NLP techniques, particularly in the task of measuring semantic similarity between words.

IV. CONCLUSION

According to the findings of this study, the word embedding approach using fastText in combination with LSA has successfully enhanced the performance of cosine similarity in measuring the semantic similarity of words based on definitions from KBBI. This method proved to be more effective than TF-IDF and several other words embedding techniques, including Word2Vec, GloVe, and S-BERT. FastText achieved the highest performance in capturing semantic similarity, with an average cosine similarity score of 0.901. Moreover, the evaluation metrics indicate that fastText achieved an accuracy of 0.88, a recall of 1.00, a precision of 0.81, and an F1 score of 0.90. This superiority is primarily attributed to its ability to generate more contextually rich word representations compared to the other methods. For future research, it is recommended to expand the scope of the corpus and explore additional word embedding techniques in order to further improve the accuracy and effectiveness of semantic similarity measurement in the context of the Indonesian language.

CONFLICTS OF INTEREST

The authors declare that there is no conflict of interest.

AUTHORS' CONTRIBUTIONS

Conceptualization, Muhammad Rafli Aditya H., Muhammad Ilham, and Dewi Fatmarani Surianto; methodology, Muhammad Rafli Aditya H. and Muhammad Ilham; software, Muhammad Rafli Aditya H.; data validation, Muhammad Rafli Aditya H., Muhammad Ilham, and Dewi Fatmarani Surianto; formal analysis, Muhammad Rafli Aditya H. and Dewi Fatmarani Surianto; investigation, Muhammad Rafli Aditya H. and Dewi Fatmarani Surianto; resources, Muhammad Rafli Aditya H., Muhammad Ilham, and Dewi Fatmarani Surianto; data curation, Muhammad Rafli Aditya H.; writing-organizing the original draft, Muhammad Rafli Aditya H., Muhammad Ilham, and Dewi Fatmarani Surianto; writing-reviewing and editing, Muhammad Rafli Aditya H., Dewi Fatmarani Surianto, and Abdul Muis Mappalotteng; visualization, Muhammad Rafli Aditya H.; supervision, Dewi Fatmarani Surianto and Abdul Muis Mappalotteng; project administration, Muhammad Rafli Aditya H. and Dewi Fatmarani Surianto; funding acquisition, Muhammad Rafli Aditya H., Muhammad Ilham, Dewi Fatmarani Surianto, and Abdul Muis Mappalotteng.

REFERENCES

[1] Y. Caterina, M.A. Yaqin, and S. Zaman, "Pengukuran kemiripan makna kalimat dalam bahasa Indonesia menggunakan metode path," *Fountain Inform. J.*, vol. 6, no. 2, pp. 45–50, Nov. 2021, doi: 10.21111/fij.v6i2.4844.

[2] N.P. Paino, D.D.S. Hutagaol, and A.U. Sagala, "Analisis penanda hubungan sinonim dan hiponimi pada puisi 'Membaca Tanda-Tanda' karya Taufiq Ismail," *Pena Literasi, J. Pendidik. Bhs. Sastra Indones.*, vol. 4, no. 1, pp. 37–44, Apr. 2021, doi: 10.24853/pl.4.1.37-44.

[3] J. Wang and Y. Dong, "Measurement of text similarity: A survey," *Information*, vol. 11, no. 9, pp. 1–17, Sep. 2020, doi: 10.3390/info11090421.

[4] G.U. Abriani and M.A. Yaqin, "Implementasi metode semantic similarity untuk pengukuran kemiripan makna antar kalimat," *ILKOMNIKA, J. Comput. Sci. Appl. Inform.*, vol. 1, no. 2, pp. 47–57, Dec. 2019, doi: 10.28926/ilkomnika.v1i2.15.

[5] R.M. Arrasyid, D.E. Putera, and A.Y.P. Yusuf, "Analisis sentimen review pembelian produk di marketplace Shopee menggunakan pendekatan natural language processing," *J. Tekno Kompak*, vol. 18, no. 2, pp. 319–330, Aug. 2024, doi: 10.33365/jtk.v18i2.3813.

[6] S.A. Zulvian, K. Prihandani, and A.A. Ridha, "Perbandingan metode MSD dan cosine similarity pada sistem rekomendasi dengan pendekatan item-based collaborative filtering," *Intecoms, J. Inf. Technol. Comput. Sci.*, vol. 4, no. 2, pp. 340–347, Dec. 2021, doi: 10.31539/intecoms.v4i2.2781.

[7] Rismayani *et al.*, "Implementasi algoritma text mining dan cosine similarity untuk desain sistem aspirasi publik berbasis mobile," *Komputika, J. Sist. Komput.*, vol. 11, no. 2, pp. 169–176, Oct. 2022, doi: 10.34010/komputika.v11i2.6501.

[8] Y.A. Pradana, I. Cholissodin, and D. Kurnianingtyas, "Analisis sentimen pemindahan Ibu Kota Indonesia pada media sosial Twitter menggunakan metode LSTM dan Word2Vec," *JPTIIK (J. Pengemb. Teknol. Inf. Ilmu Komput.)*, vol. 7, no. 5, pp. 2389–2397, May 2023.

[9] A. Nurdin, B.A.S. Aji, A. Bustamin, and Z. Abidin, "Perbandingan kinerja word embedding Word2Vec, GloVe, dan fastText pada klasifikasi teks," *J. Tekno Kompak*, vol. 14, no. 2, pp. 74–79, Aug. 2020, doi: 10.33365/jtk.v14i2.796.

[10] R.P. Nawangsari, R. Kusumaningrum, and A. Wibowo, "Word2Vec for Indonesian sentiment analysis towards hotel reviews: An evaluation study," *Procedia Comput. Sci.*, vol. 157, pp. 360–366, Sep. 2019, doi: 10.1016/j.procs.2019.08.178.

[11] R.P. Hastuti, V. Riona, and M. Hardiyanti, "Content retrieval dengan fastText word embedding pada learning management system olimpiade," *J. Internet Softw. Eng.*, vol. 4, no. 1, pp. 18–22, May 2023, doi: 10.22146/jise.v4i1.6766.

[12] B. Juarto and A.S. Girsang, "Neural collaborative with sentence BERT for news recommender system," *JOIV, Int. J. Inform. Vis.*, vol. 5, no. 4, pp. 448–455, Dec. 2021, doi: 10.30630/joiv.5.4.678.

[13] L. Cagliero, P. Garza, and E. Baralis, "ELSA: A multilingual document summarization algorithm based on frequent itemsets and latent semantic analysis," *ACM Trans. Inf. Syst. (TOIS)*, vol. 37, no. 2, pp. 1–33, Apr. 2019, doi: 10.1145/3298987.

[14] M. Panji M and A.F. Huda, "Calculating the similarity of Indonesian sentences using latent semantic indexing based on KBBI," in *2022 Int. Conf. Inform. Multimed. Cyber Inf. Syst. (ICIMCIS)*, 2022, pp. 148–153, doi: 10.1109/ICIMCIS56303.2022.10017797.

[15] A. Sanjaya and S.D. Sasongko, "Uji kemiripan kalimat menggunakan fungsi terbilang pada pre-processing dan cosine similarity dalam bahasa Indonesia," *NERO (Netw. Eng. Res. Oper.)*, vol. 7, no. 2, pp. 95–104, Nov. 2022.

[16] A. Sanjaya et al., "Pengukuran kemiripan makna menggunakan cosine similarity dan basis data sinonim kata," *J. Teknol. Inf. Ilmu Komput.*, vol. 10, no. 4, pp. 747–752, Aug. 2023, doi: 10.25126/jtiik.2023106864.

[17] R.G. Ramli and Y. Sibaroni, "Klasifikasi topik Twitter menggunakan metode random forest dan fitur ekspansi Word2Vec," *e-Proc. Eng.*, vol. 9, no. 1, pp. 79–92, Feb. 2022.

[18] W. Widayat, "Analisis sentimen movie review menggunakan Word2Vec dan metode LSTM deep learning," *J. Media Inform. Budidarma*, vol. 5, no. 3, pp. 1018–1026, Jul. 2021, doi: 10.30865/mib.v5i3.3111.

[19] E. Suryati, Styawati, and A.A. Aldino, "Analisis sentimen transportasi online menggunakan ekstraksi fitur model Word2Vec text embedding dan algoritma support vector machine (SVM)," *J. Teknol. Sist. Inf.*, vol. 4, no. 1, pp. 96–106, Mar. 2023, doi: 10.33365/jtsi.v4i1.2445.

[20] G.W. Aldiansyah, P.P. Adikara, and R.C. Wihandika, "Rekomendasi lagu cross language berdasarkan lirik menggunakan Word2Vec," *JPTIIK (J. Pengemb. Teknol. Inf. Ilmu Komput.),* vol. 3, no. 8, pp. 8036–8041, Aug. 2019.

[21] R. Julistiana, "Kosa kata bahasa Indonesia yang unik dan menarik," *Abdima Dejurnal*, vol. 1, no. 1, pp. 106–112, Apr. 2024.

[22] X. Rong, "Word2Vec parameter learning explained," 2014, *arXiv: 1411.2738*.

[23] H. Arfandy and I.A. Musdar, "Rancang bangun sistem cerdas pemberian nilai otomatis untuk ujian esai menggunakan algoritma cosine similarity," *Inspir., J. Teknol. Inf. Komun.*, vol. 10, no. 2, pp. 123–136, Dec. 2020.

[24] A.E. Sari, S. Widowati, and K.M. Lhaksmana, "Klasifikasi ulasan pengguna aplikasi mandiri online di Google Play Store dengan menggunakan metode information gain dan naive Bayes classifier," *e-Proc. Eng.*, vol. 6, no. 2, pp. 9143–9157, Aug. 2019.

[25] R.S. Amardita, Adiwijaya, and M.D. Purbolaksono, "Analisis sentimen terhadap ulasan Paris Van Java Resort Lifestyle Place di Kota Bandung menggunakan algoritma KNN," *JURIKOM (J. Ris. Komput.)*, vol. 9, no. 1, pp. 62–68, Feb. 2022, doi: 10.30865/jurikom.v9i1.3793.

[26] S. Lumbansiantar, S. Dwiasnati, and N.S. Fatonah, "Penerapan metode cosine similarity dalam mendeteksi plagiarisme pada jurnal," *Format J. Ilm. Tek. Inform.*, vol. 12, no. 2, pp. 142–150, Jul. 2023, doi: 10.22441/format.2023.v12.i2.007.

[27] Apriani, H. Zakiyudin, and K. Marzuki, "Penerapan algoritma cosine similarity dan pembobotan TF-IDF system penerimaan mahasiswa baru pada kampus swasta," *J. Bumigora Inf. Technol. (BITe)*, vol. 3, no. 1, pp. 19–27, Jun. 2021, doi: 10.30812/bite.v3i1.1110.

[28] A.B.P. Negara, H. Muhardi, and I.M. Putri, "Analisis sentimen maskapai penerbangan menggunakan metode naive Bayes dan seleksi fitur information gain," *J. Teknol. Inf. Ilmu Komput.*, vol. 7, no. 3, pp. 599–606, Jun. 2020, doi: 10.25126/jtiik.202071947.

[29] I.K.B.A.W. Kencana and W. Maharani, "Klasifikasi opini pada fitur produk berbasis graph," *e-Proc. Eng.*, vol. 4, no. 2, pp. 3148–3155, Aug. 2017.

[30] M.D.R. Wahyudi, "Penerapan algoritma cosine similarity pada text mining terjemah Al-Qur'an berdasarkan keterkaitan topik," *Semesta Tek.*, vol. 22, no. 1, pp. 41–50, May 2019, doi: 10.18196/st.221235.