IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 15, No. 2, April 2026, pp. 1919~1930
ISSN: 2252-8938, DOI: 10.11591/ijai.v15.i2.pp1919-1930

Summarization of IndoSum dataset using enhanced TextRank with weighted word embedding

Evi Yulianti, Piawai Said Umbara
Department of Computer Science, Faculty of Computer Science, Universitas Indonesia, Depok, Indonesia

Article Info

Article history:

Received Aug 15, 2024
Revised Feb 7, 2026
Accepted Mar 5, 2026

Keywords:

FastText
IndoBERT
IndoSum
Summarization
TextRank
Weighted word embedding
Word2Vec

ABSTRACT

This study evaluates the effectiveness of combining the TextRank method with word embedding on the Indonesian text summarization (IndoSum) dataset. Two experimental scenarios were applied: unweighted and weighted. The unweighted scenario incorporates word embedding, such as Word2Vec, FastText, and Indonesian bidirectional encoder representations from transformers (IndoBERT), into the TextRank framework. The weighted scenario further augments the term frequency-inverse document frequency (TF-IDF) weighting to the word embedding in the initial scenario. Our results on the effectiveness of enhanced TextRank using word embedding on IndoSum data are consistent with those reported in previous work on Liputan6 data. Both scenarios can significantly improve the effectiveness of TextRank summarization. Then, the weighted scenario showed performance improvement in most summarization systems compared to the unweighted scenario, with an average performance increase of 5.55% in recall-oriented understudy for gisting evaluation (ROUGE)-1 and 9.95% in ROUGE-2. This result confirms the robustness of the enhanced TextRank with weighted word embedding on the IndoSum data. Lastly, our study also highlights the importance of using domain-specific training data to optimize summarization performance.

Corresponding Author:

Evi Yulianti
Department of Computer Science, Faculty of Computer Science, Universitas Indonesia
Depok, West Java, Indonesia
Email: evi.y@cs.ui.ac.id

1. INTRODUCTION

The internet has become an integral part to Indonesian people in the past decade. According to the DataReportal statistics in 2023, the number of internet users in Indonesia reached 212.9 million in 2023, accounting for approximately 77.02% of the total population [1]. The internet has also shifted the way people seek information, from reading printed media to reading online media. On average, a person spends around eight to nine hours online daily, primarily accessing social media, and reading news [1].

Surveys over the past three years have consistently shown that online media remains the most preferred source of information [2]. Currently, thousands of active news platforms provide various information, ranging from politics, economics, entertainment, and technology. However, this increasing amount of information also creates the challenge of information overload, where someone often has difficulty finding relevant information amidst the large amount of text data available. Methods that can present the primary information from a text by identifying and extracting essential phrases or sentences are needed to address this issue. One commonly used method is text summarization.

Text summarization is a task of extracting meaningful information from one or multiple documents, ensuring that the information presented remains complete [3]. Based on the output form, text summarization

can be divided into abstractive and extractive [4]. Abstractive summarization generates paraphrases of sentences in the input text, while extractive summarization selects meaningful sentences directly from the input text without paraphrasing. Extractive summarization is often preferred because it is easier to implement and does not face the complex challenges of semantic representation and natural language generation [5].

One of the most popular algorithms in extractive summarization is TextRank [6], which adapts PageRank algorithm [7] to rank sentences within a document. It has been widely used in previous research on various tasks due to its proven effectiveness, such as on news article summarization [8]–[14], biomedical text summarization [15], legal document summarization [16], web document summarization [17], topic/keyword extraction [18], software documentation (release note production) [19], and information credibility assessment [20]. However, TextRank has limitations in capturing the semantic relationships between sentences because it primarily relies on word or phrase overlaps without considering the deeper context. To overcome these limitations, several studies have proposed incorporating word embeddings into TextRank [11], [12], [14], [18].

In general, this study aims to examine the effectiveness of TextRank method that is combined with weighted word embedding, proposed in [14], to summarize the Indonesian text summarization (IndoSum) dataset [21]. Word embedding [22], [23] is used to better estimate the sentence similarity within TextRank. Then, word weighting, such as term frequency-inverse document frequency (TF-IDF), is used to weigh the word embedding, emphasizing the importance of a word within a document relative to the entire corpus, focusing on more relevant words, and reducing the influence of ordinary words with little informative value [13], [17]. The use of IndoSum data in this study aims to evaluate the robustness of the method when it is applied to other datasets with different characteristics. Here, the IndoSum dataset has a larger number of documents and a longer average document length than Liputan6 dataset that was used in the original paper (the statistical comparison of these two datasets can be seen in section 3). With these two characteristics, the use of IndoSum in this study is expected to provide a more comprehensive understanding of the effectiveness of the enhanced TextRank method on Indonesian texts. In addition, we also explore the effect of utilizing different sources of data to train the word embedding models that enhance TextRank. While in [14], the Word2Vec and FastText models were only trained using Wikipedia data; in this work, we also used IndoSum [21] and Tempo [24] data to train those models. This exploration investigates to what extent the summarization system performance differs when the word embeddings are learned from the same domain corpus, i.e., the news corpus. To sum up, the main contributions of this paper are twofold. Firstly, we examine the performance of TextRank method combined with word embedding to summarize IndoSum data in an unweighted and weighted scenario. This study aims to confirm the robustness of the method when it is applied to other datasets with different characteristics. Secondly, we investigate the effect of using different data sources to train the word embedding models on the summarization system performance. This study analyzes the effect of learning word embeddings using the same domain corpus (as the documents to be summarized) on the summarization system performance.

This study provides valuable insights into future summarization research. It emphasizes the efficacy of TextRank method that is combined with weighted word embedding on the IndoSum dataset, and the effect of using the same domain corpus to train word embedding models on the summarization performance. This research also advances the development of more effective Indonesian natural language processing (NLP) applications, more specifically, a summarization system, with implications for improving user effectiveness and efficiency in accessing important information from documents.

The rest of the paper is then organized as follows. Section 2 presents the reviews of related literature. Section 3 describes the methods used in our study. Section 4 describes our results and analysis. Sections 5 and 6, respectively explain the discussion and conclusion of our work.

2. LITERATURE REVIEW

There are some previous studies that are closely related to this research [11]–[14], [17], [18]. These studies enhanced TextRank method using word embedding and/or TF-IDF weighting methods. The position of our work towards these previous studies is summarized in Table 1.

Barman *et al.* [11] employed the global vectors (GloVe) embedding model, trained on 2014 Wikipedia corpus, to summarize English news articles from the BBC news summary. They used the GloVe embedding to produce the sentence vectors to be input into TextRank summarizer. In contrast to Barman *et al.* [11], we use Word2Vec, FastText, and IndoBERT embeddings to be combined with TextRank method for summarizing Indonesian text. Rani and Lobiyal [12] used Word2Vec embedding trained on the Google News corpus in their proposed semantic summarizer using linguistic and statistical features. This work is different from theirs in that we use Word2Vec in combination with TextRank summarization algorithm.

Table 1. Research position of this work

Work	Task	Method	Word embedding	Weighting status of word embedding	Using TF-IDF?	Data	Corpus to train word embedding models	Lang
Barman *et al.* [11]	Summarization	TextRank	GloVe	Unweighted	No	BBC News	Wikipedia and Gigaword 5	EN
Rani and Lobiyal [12]	Summarization	Semantic summarizer using linguistic and statistical features	Word2Vec	Weighted	Yes	DUC 2007	Google News	ID
Zuo *et al.* [18]	Topic extraction	TextRank	Word2Vec	Unweighted	No	Literature abstracts	Wikipedia	EN
Guan *et al.* [17]	Summarization	TextRank → keyword extraction. TF-IDF → sentence ordering	-	-	Yes	Web documents	-	EN
Aware *et al.* [13]	Summarization	TextRank	-	-	Yes	BBC News	-	EN
Yulianti *et al.* [14]	Summarization	TextRank	Word2Vec, FastText, IndoBERT	Unweighted and Weighted	Yes	Liputan6	Wikipedia	ID
This work	Summarization	TextRank	Word2Vec, FastText, IndoBERT	Unweighted and Weighted	Yes	IndoSum	Wikipedia Tempo IndoSum	ID

Zuo *et al.* [18] used Word2Vec to enhance the TextRank method to incorporate the semantic relationship between keywords. The main difference between our work and Zuo *et al*'s work is in the downstream task. In their work, TextRank was adopted for the topic extraction task, but in our work, it is applied for text summarization. In addition, TF-IDF weighting is also employed in our work to weigh the word embeddings. Guan *et al.* [17] also utilized TextRank method for keyword extraction, although the main task of their study is on text summarization. They used the TextRank method to extract important keywords from documents, and each sentence in a document was scored using the total TF-IDF scores of keywords in the sentence. The top-scoring sentences are then taken as a summary. While Guan *et al.* [17] used the TF-IDF method to weigh the keywords we use it to weigh the word embedding for text summarization.

Zaware *et al.* [13] applied TF-IDF to summarize news from the BBC news summary. In their work, the TF-IDF is used to compute each element of the sentence vectors (bag-of-words vectors using TF-IDF), while in this research, it is used to weigh the word embeddings, and the average weighted word embeddings will form the sentence vectors. Recently, Yulianti *et al.* [14] incorporated Word2Vec, FastText, and IndoBERT embedding into TextRank method, to summarize Indonesian text. While they used Liputan6 dataset [8] in this work, their method are evaluated using IndoSum dataset [21] that has significantly higher number of documents, and each contaning longer text (see Table 2 for the comparison of these two datasets). We further analyse the effect of training word embedding models using different data sources.

3. METHOD

3.1. Dataset

IndoSum data [21] is used in this study to evaluate the performance of the enhanced TextRank method with word embedding. This is different from [14] who utilized the Liputan6 dataset using the same method as this work. The underlying motivation for using IndoSum dataset in this study is because it is another benchmark for Indonesian text summarization, besides Liputan6 dataset that has different characteristics from Liputan6, such as a larger number of documents and a longer average document length than Liputan6. Table 2 compares the statistics between the IndoSum and Liputan6 datasets in more detail.

Table 2. Statistical comparison between Liputan6 and IndoSum datasets

Dataset	Number of documents	Average number of sentences per document	Average number of words per document	Average number of words per sentence
Liputan6	10,972	11.71	218.55	18.67
IndoSum	93,987	18.36	346.84	18.89

IndoSum consists of 93,870 news articles from a former online news aggregator site called Shortir.com. This dataset is publicly available under Apache license, Version 2.0 at https://github.com/kata-ai/IndoSum. Each article includes attributes such as title, category, article content, news source, source URL, extractive summary flag (boolean), and abstractive summary. We used the extractive summary ground truth

as a reference summary in our evaluation. IndoSum is organized into five cross-validation folds, each divided into three subsets: train, dev, and test sets. Each fold contains 18,774 news articles, with the train, dev, and test sets having approximately equal proportions (around 0.75:0.05:0.2). In this work, the test sets from each fold are combined for evaluation in our experiment. Table 3 describes the statistics of IndoSum dataset used in this work. From Table 3, it can be observed that the documents in the IndoSum dataset have around 18 sentences per document. The average number of words per document is approximately 347, with each sentence containing about 19 words. To ensure consistency and comparability with the ground truth summaries, the length of our summaries will follow the length of the ground truth summaries.

A few preprocessing steps are performed on the IndoSum data. They include case folding, removing tokens containing non-alphanumeric characters (except hyphen), and replacing number tokens with a special token @@NUM@@. This pre-processing is needed because we use the pretrained Word2Vec and/or FastText that was preprocessed using the same procedure.

We also utilize the Indonesian Wikipedia database dump from May 2nd, 2024 (idwiki-20240502-pages-articles.xml.bz2*)*, to train Word2Vec and FastText embedding models. The Indonesian Wikipedia dataset was chosen for its extensive size and public availability, which aligns with common practice in similar research. Document scraping is carried out using the Gensim library, which efficiently processes the large volume of Wikipedia articles. To examine the influence of using different data sources to train the word embedding models on summary performance, we also use the IndoSum data (in the train and val sets), consisting of 75,096 documents, and pretrained Word2Vec and FastText models using Tempo data, consisting of 400K docs, 160M word tokens, and 600K word types.

In addition to Word2Vec and FastText, we also leverage a pre-trained language model, BERT, specifically IndoBERT [25]. IndoBERT is trained on the Indo4B dataset, a large-scale Indonesian corpus that contains 3,581,301,476 words. This extensive dataset ensures that the BERT model captures a wide range of linguistic patterns and nuances present in the Indonesian language.

Table 3. Statistics of IndoSum datasets

Subset	Number of documents	Average number of sentences per document	Average number of words per document	Average number of words per sentence
Train	71,353	18.36	346.79	18.88
Development	3,743	18.33	348.24	19.00
Test	18,774	18.36	346.77	18.89
Total	93,870	18.36	346.84	18.89

3.2. The framework of summarization system

We use the enhanced TextRank method using weighted word embedding method [14], illustrated in Figure 1. For details of the method, please refer to the original paper. In summary, this method works as follows: initially, word embeddings that are obtained from Word2Vec, FastText, or IndoBERT models are weighted using TF-IDF scores that are computed using IndoSum data. Then, sentence embeddings are generated by averaging the word embeddings of words contained in the sentence. The pairwise cosine similarity between sentences is computed to build a similarity matrix to be input into the TextRank method, which will rank the sentences according to the relationship between other sentences in a document. The top N sentences are then selected as a summary.

3.2.1. Word embedding

This study explores three types of word embeddings: Word2Vec [22], FastText [23], and IndoBERT [25]. We train the Word2Vec and FastText models using Indonesian Wikipedia, and IndoSum dataset. Additionally, we also used pre-trained Word2Vec and FastText models learned from Tempo [24] data, and the Indonesian BERT pre-trained model from previous work: IndoBERT [25].

This work employs the skip-gram model for Word2Vec because it is more effective at capturing detailed semantic relationships. Using the Gensim library, the Word2Vec model is trained on the Indonesian Wikipedia and IndoSum dataset. Following a research by Yulianti *et al.* [14], default hyperparameter values are used to build the Word2Vec model, such as: a learning rate of 0.025, five training epochs, and a window size of five. We use exactly the same hyperparameter settings as [14] to ensure fair comparison.

A FastText model incorporates subword information to generate word embedding. Therefore, this approach can generate embeddings for rare or unseen words based on their subwords, addressing the out-of-vocabulary problem. The FastText model is implemented using Gensim with Skip-gram architecture and default hyperparameter settings. It is trained on the Indonesian Wikipedia and IndoSum datasets, following a similar approach to the Word2Vec model.

BERT generates embeddings by considering the context in which a word appears. This bidirectional approach enables BERT to capture complex word relationships and contextual nuances more effectively. In this work, BERT is only used to generate word embeddings, which are further incorporated into the TextRank algorithm. BERT has been pre-trained on a large Indonesian text corpus of around 4 billion words, known as IndoBERT, in two versions: IndoBERTBASE and IndoBERTLARGE. Here, the large version has a higher number of parameters than the base version.

3.2.2. Term frequency-inverse document frequency weighting

TF-IDF is a method used to assign weight to words in a text based on their frequency within a document and across a collection of documents. In our summarization method, TF-IDF is used to weigh the word embeddings (see Figure 1), by multiplying the word embedding with the TF-IDF of that word, before averaging to the sentence embedding. The average of all weighted word embeddings in a sentence is then taken as a sentence vector. In vanilla TextRank, the sentence vector is obtained directly using a simple bag-of-words method, which therefore do not capture the semantic information within sentences. In this work, word embedding enables the incorporation of semantic information, and the TF-IDF method further gives more weight to the word embedding of more important words. Therefore, the sentence vector will be contributed more by the more important words rather than the less important words. The formulas to compute TF and IDF scores are given by (1) and (2).

$$tf(t,d) = \frac{f_{(t,d)}}{\sum_{\hat{t}\in d} f_{(\hat{t},d)}} \quad (1)$$

$$idf(t) = \frac{|C|}{|d\in C: t\in d|} \quad (2)$$

Where $f_{(t,d)}$ is the actual TF of word t in document d, $\sum_{\hat{t}\in d} f_{(\hat{t},d)}$ is the total terms in document d, $tf(t,d)$ is the normalized TF of word t in document d, $|C|$ is the total number of documents in the collection, $|d \in C: t \in d|$ is the number of documents containing the word t, and $idf(t)$ is the IDF of word t.

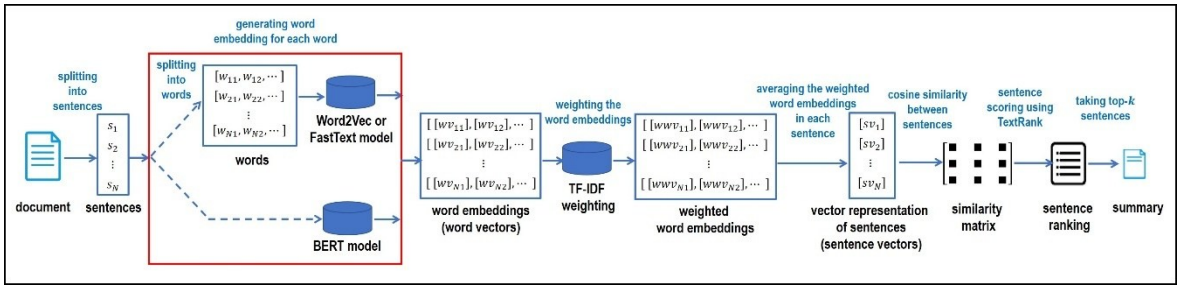

Figure 1. The framework of the enhanced TextRank summarization system [14]

3.2.3. TextRank

In the text summarization domain, TextRank is a graph-based ranking algorithm for sentence ranking. The relationships between sentences in a document are the main factors that contribute to the score assigned by TextRank to each sentence. In this study, the traditional bag-of-words vectors that represent sentence vectors in the original TextRank is replaced with the average of TF-IDF weighted word embeddings of words in the sentences. Here, word embeddings are obtained from Word2Vec, FastText, and IndoBERT models. This modification is expected to enhance the accuracy of the original TextRank summarization method because word embeddings consider semantic similarity between sentences, and the TF-IDF allows more important words to contribute higher in the sentence vectors.

After sentence vectors are formed, pairwise cosine similarity between sentences is computed to construct a similarity matrix. This matrix forms the basis for creating a weighted graph of sentences, where a vertex is represented as a sentence, and an edge is represented as a similarity score between two sentences. This graph will be processed by the TextRank method to score the sentences in a document, and the top N highest-scoring sentences are then selected as the summary. Here, N is adjusted to match the actual length of the ground truth summary to ensure fair recall-oriented understudy for gisting evaluation (ROUGE) score comparison. Note that this may not reflect the real-world summarization scenario that usually uses the same fixed-length summary for each document.

The function to apply the TextRank method is presented in (3).

$$S(V_i) = (1-d) + d \times \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{V_k \in Out(V_j)} w_{jk}} \times S(V_j) \quad (3)$$

Where V_i is the i-th vertex, $S(V_i)$ is the score for the i-th vertex (the sentence score), $In(V_i)$ is the set of vertices that go into V_i, $Out(V_j)$ is the set of vertices that go out from V_j, d is damping factor, w_{ji} is the weight of the edge between sentences j and i, and w_{jk} is the weight of the edge between sentences j and k.

The Python library NetworkX was used to apply the TextRank method. We set the damping factor to 0.85, because it has been shown to perform well in the original TextRank method [9], as well as some previous work on summarization [10], [11], [14], [15], [19], [26]. We do not experiment with varying numbers of damping factors and decide to use the same value as [14], i.e., 0.85, to ensure fair comparison.

3.3. Evaluation metric

The results are evaluated using the ROUGE metric [10], a widely used metric for assessing extractive text summarization performance. To measure our ROUGE scores, we choose the F1-score, which balances precision and recall, thus offering a comprehensive assessment of summarization quality. The use of ROUGE-F-1 allows for the balance between informativeness and conciseness in the measurement. In this study, we only use ROUGE score and do not consider other summarization metrics, such as BERTScore [27] or metric for evaluation of translation with explicit ordering (METEOR) [28], because we want to make fair comparison with the results obtained in [14] that used Liputan6 dataset. Note that the goal of this study is to evaluate the robustness of the method proposed in [14] when it is evaluated on IndoSum dataset. We utilize the Python library rouge_score to compute the ROUGE scores of our generated summaries. Before the ROUGE scores are computed, case normalization is applied to convert all letters in the ground truth summaries and system summaries into lowercase.

The statistical significance test between the ROUGE scores of our summaries against those of baseline summaries is conducted using paired *t*-test. The confidence level used in this study is 0.05. The sample size is 18,774 documents, as it is the total number of documents in the test split of IndoSum dataset that are summarized by our methods. Paired *t*-test is chosen because, given pairs of scores, i.e., scores of our methods and each of the baseline methods, we want to decide if the difference between paired measurements for a population is zero or not. If the difference is not zero, then it means there is a statistically significant difference between the results of our summarization method and the baseline methods.

The basic assumptions for paired *t*-test [29] are the data is continuous (i.e., the ROUGE scores are continuous variables as they have ratio scale), each of the paired measurements are obtained from the same subject (each pair of ROUGE scores are computed from the summaries generated from the same document). The differences between the paired observations are approximately normally distributed (i.e., the results of the Shapiro-Wilk test shows that the difference of ROUGE scores between our method and baseline methods are normally distributed, since the p-value <0.05). There are no outliers across either of the group (i.e., there is no ROUGE-score values with a z-score greater than 3 or less than -3).

4. RESULTS AND ANALYSIS

4.1. The enhanced TextRank using word embedding

Table 4 explains the results of enhanced TextRank methods using word embedding from Word2Vec, FastText, and IndoBERT to summarize documents in IndoSum dataset (test subset). To examine whether our results are consistent with those obtained in previous work using Liputan6 dataset, we also display in the bottom part of Table 4 the scores reported in [14]. To be comparable, we only present in the table the results of summarization systems that use Word2Vec and FastText trained using Wikipedia. The results of those using Word2Vec and FastText trained using IndoSum and Tempo data are presented later (see Table 5).

The results indicate that all systems utilizing word embeddings to summarize documents in IndoSum dataset outperform the original TextRank baseline. This highlights the effectiveness of integrating word embeddings into the TextRank algorithm, enhancing its summarization capabilities. This is consistent with the results from previous work using Liputan6 dataset [14].

The system that gains the best results is the enhanced TextRank using IndoBERTLARGE embedding, which is also in line with the results of previous work [14]. The IndoBERTLARGE system performs slightly better than the IndoBERTBASE system, achieving ROUGE-1 and ROUGE-2 scores of 0.460 and 0.345 compared to 0.435 and 0.313, respectively. This marginal improvement aligns with previous findings, likely due to IndoBERTLARGE's greater capacity to capture contextual nuances given its larger number of layers and parameters. Note that the embedding dimensionality and the number of hidden layers, as well as attention heads in IndoBERTLARGE, are higher than IndoBERTBASE. Consequently, the number of parameters learned in IndoBERTLARGE is also higher than IndoBERTBASE. The embedding size of

IndoBERTLARGE is 1,024, while that of IndoBERTBASE is 768; The number of hidden layers in IndoBERTLARGE is 24, while that in IndoBERTBASE is 12; The number of attention heads in IndoBERTLARGE is 16, while that in IndoBERTBASE is 12; The total parameters of IndoBERTLARGE is 335.2M, while that of IndoBERTBASE is 124.5M.

Overall, contextualized word embeddings, IndoBERT, significantly boost the performance of the TextRank algorithm. It can capture more semantic meanings and linguistic features than static word embedding models (Word2Vec and FastText). This improvement can be attributed to the superior semantic understanding provided by these embeddings, which results in more accurate summaries.

Based on running times of methods, it appears from the table that TextRank+Word2Vec is the most efficient because it can run very fast, requiring only 5 minutes to summarize around 19K documents. The TextRank+FastText takes longer time to finish the same job because in the FastText model, the embedding of each word is obtained by summing up the embeddings of its subwords. Therefore, its computation takes longer. In addition, Wikipedia collection also contains a huge number of words, and this makes the vocabulary size for FastText much bigger than Word2Vec, resulting in much higher execution time for the TextRank+FastText model. While in the unweighted scenario the IndoBERT-based models are the most effective (based on ROUGE scores), it turns out to be the least efficient as it takes the longest time to run. It takes about ten hours to summarize around 19K documents. This can be explained because when generating the embedding of words in the sentence using IndoBERT, the input sentence needs to be passed forward through the neural network architecture that consists of several feed-forward and self attention layers. This process is clearly expensive, especially if it is conducted for a high number of long documents.

It is interesting to note that the original TextRank method turns out to be quite time consuming. It takes almost ten times longer than the TextRank+Word2Vec model. This can be explained as follows. The original TextRank method uses bag-of-words vector to represent a sentence vector, and we use TF-IDF representation to represent each word element in the vector. The vector dimension is the total unique words in the document. Therefore, the dimension of vectors could be more than a thousand, depends on the length of the document. The lengthier the document, the higher the number of total unique words in that document, and so the higher the dimension of sentence vectors. When a document has a high dimension of sentence vectors, it takes longer times to generate the vectors. Note that every time we summarize a new document, the sentence vectors for each document need to be computed from scratch. This is different from the TextRank+Word2Vec model that just needs to look up the word embedding from the pretrained Word2Vec model, and take the average to produce sentence vectors. In addition, when the dimension of sentence vectors is high, it also takes longer times to build the sentence similarity matrix and to run the TextRank method. This causes the original TextRank method to run much slower than the TextRank+Word2Vec method.

To examine the effect of using a variation of data to train word embedding models, the Word2Vec and FastText embeddings are also trained using IndoSum data (train subset) and Tempo. These embeddings are then used to summarize IndoSum data (test subset). Because of the high complexity and resources needed to pre-train IndoBERT, training IndoBERT models using a variation of data is omitted in this investigation. The results are displayed in Table 5. Note that in this table, we cannot put the scores in [14] since they did not use a different dataset other than Wikipedia to train the Word2Vec and FastText models.

Table 4. The results using TextRank combined with word embedding

Work	Dataset	Summarization system	Data to train word embedding models	ROUGE-1	ROUGE-2	Running time
Our work	IndoSum	TextRank	-	0.382	0.247	59m 28 s
		TextRank+FastText	Wikipedia	0.417*	0.288*	9m 45 s
		TextRank+Word2Vec	Wikipedia	0.424*	0.296*	**5m 24 s**
		TextRank+IndoBERT$_{BASE}$	Indo4B (pre-trained)	0.435*	0.313*	9 h 51 m 39 s
		TextRank+IndoBERT$_{LARGE}$	Indo4B (pre-trained)	**0.460***	**0.345***	10 h 4 m 17 s
Yulianti *et al*. [14]	Liputan6	TextRank	-	0.348	0.234	-
		TextRan+FastText	Wikipedia	0.378*	0.264*	-
		TextRank+Word2Vec	Wikipedia	0.382*	0.269*	-
		TextRank+IndoBERT$_{BASE}$	Indo4B (pre-trained)	0.393*	0.284*	-
		TextRank+IndoBERT$_{LARGE}$	Indo4B (pre-trained)	**0.394***	**0.285***	-

Symbol * denote significant differences against TextRank, according to the paired *t*-test using confidence level 0.05, ($p < 0.05$).

Table 5. The results using variation of data to train the embedding models

Dataset	Summarization system	Data to train word embedding models	ROUGE-1	ROUGE-2	Running time
IndoSum	TextRank+FastText	Tempo	0.419*	0.294*	5m 41s
	TextRank+Word2Vec	Tempo	0.431*	0.311*	5m 37s
	TextRank+Word2Vec	IndoSum	0.457*	0.338*	5m 18s
	TextRank+FastText	IndoSum	0.466*	0.350*	5m 18s

Symbol * denote significant differences against TextRank, according to the paired *t*-test ($p < 0.05$).

Comparing the results from Tables 4 and 5, we can see that the systems with Word2Vec and FastText models trained on the IndoSum data gain higher accuracy than those trained on Wikipedia data. More specifically, the combination of TextRank with the FastText model trained on the IndoSum dataset achieves the highest performance. This demonstrates that using training data from the same domain as the test data can improve system accuracy. These findings highlight the importance of selecting appropriate training data and models to achieve optimal summarization results. The systems leveraging Word2Vec and FastText models trained on Tempo dataset also tends to be more accurate than those using Wikipedia. It could be because, similar to IndoSum, Tempo is also a collection of news articles. Therefore, it may share a lot of similar vocabulary with IndoSum. This increased performance underscores the importance of using domain-specific training data to enhance the quality of word embeddings and, consequently, the summarization results.

To better understand whether the Tempo and IndoSum (train) data are actually more similar to the IndoSum (test) compared to Wikipedia because they have the same domain data, we compute term overlap between each of these data with IndoSum (test) data. This overlap ratio is simply formulated as (4).

$$\text{Overlap Ratio} = \frac{|V_A \cap V_B|}{|V_B|} \quad (4)$$

Where V_A is set of unique terms in dataset A; V_B is set of unique terms in dataset B; and $| V_A \cap V_B |$ is the number of similar words in datasets A and B.

From Table 6 shows that the overlap ratio between Wikipedia and IndoSum (test) is obviously the lowest. Tempo and IndoSum (train) data are shown to contain a higher number of overlapping words with IndoSum (test) data because they have the same domain data, i.e., news articles. This causes the Word2Vec and FastText learned from these two datasets are more accurate compared to Wikipedia when they are tested to the IndoSum (test) data. This explains our findings described in Table 5.

4.2. The enhanced TextRank using weighted word embedding

Table 7 presents the results of enhanced TextRank methods using weighted word embedding to summarize documents in IndoSum dataset (test subset). Again, we also display the results in [14] to examine whether our results using IndoSum data are consistent with their results using Liputan6 data. It appears from the table that all systems utilizing weighted word embeddings to summarize documents in IndoSum dataset outperform the original TextRank baseline. Compared to previous results without TF-IDF weighting (displayed earlier in Table 4), there is a substantial improvement in ROUGE-1 and ROUGE-2 scores for the systems using static embedding (Word2Vec and FastText). Notably, the combination of TextRank and Word2Vec demonstrates the highest performance increase (with improvements of 13.21% for ROUGE-1 and 25.34% for ROUGE-2). This finding also agrees with that reported in [14] using Liputan6 data.

Table 6. The overlap ratio between training data (A) and testing data (B)

Between datasets	Overlap ratio
A: Wikipedia, B: IndoSum (test)	0.4237
A: Tempo, B: IndoSum (test)	0.6034
A: IndoSum (train), B: IndoSum (test)	0.7829

Note: training data (A) indicates the data used to train Word2Vec and FastText models

Table 7. The results using TextRank combined with weighted word embedding

Work	Dataset	Summarization system	Training data	ROUGE-1	ROUGE-2	Running time
This work	IndoSum	TextRank	-	0.382	0.247	59 m 28 s
		TextRank+IndoBERT$_{BASE}$ + TF-IDF	Indo4B (pre-trained)	0.390*	0.260*	9 h 27 m 17 s
		TextRank+IndoBERT$_{LARGE}$ + TF-IDF	Indo4B (pre-trained)	0.418*	0.295*	9 h 35 m 23 s
		TextRank+FastText+TF-IDF	Wikipedia	0.478*	0.368*	10 m 44 s
		TextRank+Word2Vec+TF-IDF	Wikipedia	**0.480***	**0.371***	**5 m 33 s**
Yulianti *et al.* [14]	Liputan6	TextRank	-	0.348	0.234	-
		TextRank+IndoBERT$_{BASE}$+ TF-IDF	Indo4B (pre-trained)	0.404*	0.297*	-
		TextRank+FastText+TF-IDF	Wikipedia	0.404*	0.298*	-
		TextRank+IndoBERT$_{LARGE}$+ TF-IDF	Indo4B (pre-trained)	0.404*	0.298*	-
		TextRank+Word2Vec+TF-IDF	Wikipedia	**0.408***	**0.304***	**-**

Symbol * denote significant differences against TextRank, according to the paired *t*-test ($p < 0.05$).

Next, for the systems using contextual embedding (IndoBERT), the increased performance of weighted word embedding approach over unweighted word embedding approach is not found. This is similar to the results in [14]. which could not find statistically significant improvement for the systems using IndoBERT in Liputan6 data, although they still showed a slight increase in their ROUGE scores. Using IndoSum data, the system using IndoBERTBASE shows a ROUGE-1 score decrease of 0.045 (10.34%) and a ROUGE-2 score decrease of 0.053 (16.93%). Similarly, the system using IndoBERTLARGE shows a ROUGE-1 score decrease of 0.042 (9.13%) and a ROUGE-2 score decrease of 0.050 (14.49%). This indicates that TF-IDF weighting is not effective for contextual embeddings, because their inherent contextual understanding already captures essential semantic nuances and the importance of words according to the context. Therefore, an additional weighting that assumes word independence, such as TF-IDF, instead makes the semantic representation of words less accurate. This result indicates that the enhanced TextRank using unweighted IndoBERT embedding (without TF-IDF weighting) is already near the performance ceiling for this dataset.

Table 8 displays the results using variation of data to train word embedding models for the summarization systems with weighted word embedding approach. Here, the Word2Vec and FastText embeddings are trained using IndoSum data (train subset) and Tempo. These embeddings are then used to summarize IndoSum data (test subset). The variations of IndoBERT training data are omitted because its pre-training process requires high complexity and resources. The combination of TextRank with the FastText and Word2Vec embeddings trained on the IndoSum dataset consistently shows substantial improvement compared to those trained on Wikipedia. This emphasizes the importance of using domain-specific training data in enhancing the quality of word embeddings and summarization results. The use of Tempo data, however, appears to give inconsistent result, since it slightly increases the performance of system using FastText, but it slightly decreases the performance of system using Word2Vec.

Table 8. The results using a variation of data to train the embedding models

Dataset	Summarization system	Training data	ROUGE-1	ROUGE-2	Running time
IndoSum	TextRank+Word2Vec+TF-IDF	Tempo	0.477*	0.368*	5m 49s
	TextRank+FastText+TF-IDF	Tempo	0.480*	0.372*	5m 36s
	TextRank+Word2Vec+TF-IDF	IndoSum	0.494*	0.388*	5m 25s
	TextRank+FastText+TF-IDF	IndoSum	0.498*	0.393*	5m 47s

Symbol * denote significant differences against TextRank, according to the paired *t*-test ($p < 0.05$).

5. DISCUSSION AND FUTURE WORK

To provide a clearer picture of the performance of the evaluated methods, we conducted a comparative analysis on the summaries generated by various systems against the ground truth summary. Table 9 shows an example of generated summaries for a document with id 34 in the test subset of IndoSum dataset using the TextRank method combined with various word embedding models. In the scenario using unweighted word embeddings, the summary generated by the TextRank+FastText (Wikipedia) system does not contain any sentences from the ground truth.

The TextRank+Word2Vec (Wikipedia) system performs slightly better, including one sentence from the ground truth summary. The TextRank+IndoBERTBASE and TextRank+IndoBERTLARGE systems also only include one sentence from the ground truth summary, but they are more effective because containing a higher number of term overlaps with the ground truth summary, resulting in higher ROUGE scores. This showcases their superior ability to understand the context and semantics of the text. When the FastText and Word2Vec are pretrained using IndoSum dataset, the performance of the models increased, resulting in both the ROUGE-1 and ROUGE-2 scores are 1 (see TextRank+FastText(IndoSum) and TextRank+Word2Vec(IndoSum)). This reinforces our finding that using the same domain dataset as the test data can improve summary accuracy.

When the TF-IDF weighting is incorporated into the models, the performance of the systems based on static word embeddings also increased, resulting both of the ROUGE-1 and ROUGE-2 scores are 1 (see TextRank+FastText(Wikipedia)+TF-IDF and TextRank+Word2Vec(Wikipedia)+TF-IDF). This highlights the benefits of the weighting method for static embedding. However, the results of enhanced TextRank using IndoBERT model are not improved by the weighting method. This is because the bidirectional feature of the IndoBERT models already captures the semantic meaning of words. Therefore, the additional weighting instead makes the semantic representation of words less accurate. To overcome this limitation, in the future, more advanced contextual embeddings from a multilingual language model that includes Bahasa Indonesia in the pretraining data can be investigated.

Overall, the findings obtained in this study confirm the results reported in [14] that combining TextRank and word embedding is effective at improving the accuracy of a summarization system. Further

adding TF-IDF weighting can enhance the static embedding-based systems, but it is less effective or even detrimental for contextual embeddings like IndoBERT. In addition, our results also discover new findings regarding the influence of using the same data domain for training word embedding models on the accuracy of the generated summaries. Our results emphasize the importance of using domain-specific training data in enhancing the quality of word embeddings and summarization results.

The positive ROUGE score differences between our methods and the baseline methods indicate that our summaries have a higher number of term overlaps with the reference summaries, compared to the baseline summaries. While these results reflect the quality of summaries based on offline metric, it does not necessarily reflect the users' judgment. This is another limitation of this work. Therefore, further user study can also be performed in the future to assess the quality of summaries directly based on users' perspectives. This additional investigation on human evaluation of summary quality can confirm the results obtained in this work based on automatic evaluation.

Table 9. Example of generated summaries

System	Summaries
Ground truth	*TURKI – Lebih dari 40 orang, sebagian besar mantan anggota unit pasukan khusus Turki, dijatuhi hukuman seumur hidup setelah dinyatakan terbukti bersalah mencoba membunuh Presiden Recep Tayyip Erdogan. Dalam persidangan di Mugla, kota di Turki barat daya hari Rabu (04/10), hakim menyatakan terdakwa menyerang hotel di resor wisata tepi laut di Marmaris, yang menjadi tempat penginapan Presiden Erdogan ketika berlangsung upaya kudeta yang gagal pada 15 Juli 2016.* (English translation: TURKEY – More than 40 individuals, most of them former members of Turkish special forces units, have been sentenced to life imprisonment after being found guilty of attempting to assassinate President Recep Tayyip Erdogan. During the trial in Mugla, a city in southwestern Turkey, on Wednesday (10/04), the judge stated that the defendants attacked a hotel in the seaside resort of Marmaris, where President Erdogan was staying during the failed coup attempt on July 15, 2016.)
TextRank+Word2Vec (Wikipedia)	(English translation: TURKEY – More than 40 individuals, most of them former members of Turkish special forces units, have been sentenced to life imprisonment after being found guilty of attempting to assassinate President Recep Tayyip Erdogan. According to the parliamentary committee investigating the incident, 34 special forces members arrived at the hotel at 3:20 a.m., while Erdogan landed in Istanbul at 3:40 a.m.) (ROUGE-1 =0.557, ROUGE-2 =0.433)
TextRank+FastText (Wikipedia)	(English translation: In an interview with CNN, President Erdogan said, "If I had stayed 10 or 15 minutes longer (in the hotel), I would have been killed or taken hostage." According to the parliamentary committee investigating the incident, 34 special forces members arrived at the hotel at 3:20 a.m., while Erdogan landed in Istanbul at 3:40 a.m.) (ROUGE-1 =0.303, ROUGE-2 =0.034)
TextRank+IndoBERT$_{BASE}$	(English translation: TURKEY – More than 40 individuals, most of them former members of Turkish special forces units, have been sentenced to life imprisonment after being found guilty of attempting to assassinate President Recep Tayyip Erdogan. President Erdogan himself left his resort in Marmaris before the attack, which killed two police officers, was launched.) (ROUGE-1 =0.595, ROUGE-2 =0.514)
TextRank+IndoBERT$_{LARGE}$	(English translation: During the trial in Mugla, a city in southwestern Turkey, on Wednesday (10/04), the judge stated that the defendants attacked a hotel in the seaside resort of Marmaris, where President Erdogan was staying during the failed coup attempt on July 15, 2016. Some analysts dismissed President Erdogan's claim that he had already left the hotel before the firing squad arrived at the hotel where Erdogan and his family were staying.) (ROUGE-1 =0.662, ROUGE-2 =0.625)
TextRank+Word2Vec (IndoSum)	Same as the ground truth summary (ROUGE-1 =1, ROUGE-2 =1)
TextRank+FastText (IndoSum)	Same as the ground truth summary (ROUGE-1 =1, ROUGE-2 =1)
TextRank+Word2Vec (Wikipedia)+TF-IDF	Same as the ground truth summary (ROUGE-1 =1, ROUGE-2 =1)
TextRank+FastText (Wikipedia)+TF-IDF	Same as the ground truth summary (ROUGE-1 =1, ROUGE-2 =1)
TextRank+IndoBERT$_{BASE}$+ TF-IDF	Same as the summary of TextRank+IndoBERT$_{BASE}$ above. (ROUGE-1 =0.595, ROUGE-2 =0.514)
TextRank+IndoBERT$_{LARGE}$+ TF-IDF	(English translation: During the trial in Mugla, a city in Southwestern Turkey, on Wednesday (10/04), the judge stated that the defendants attacked a hotel in the seaside resort of Marmaris, where President Erdogan was staying during the failed coup attempt on July 15, 2016. President Erdogan himself left his resort in Marmaris before the attack, which killed two police officers, was launched.) (ROUGE-1 =0.704, ROUGE-2 =0.650)
TextRank+Word2Vec (IndoSum)+TF-IDF	Same as the ground truth summary (ROUGE-1 =1, ROUGE-2 =1)
TextRank+FastText (IndoSum)+TF-IDF	Same as the ground truth summary (ROUGE-1 =1, ROUGE-2 =1)

6. CONCLUSION

This research evaluates the robustness of the enhanced TextRank method using word embeddings on the IndoSum dataset. In the first scenario of enhancing TextRank using unweighted word embeddings, our results are consistent with the findings in previous work using Liputan6 data. The contextual embeddings, like IndoBERT significantly improve the summarization capabilities of the original TextRank method and the enhanced TextRank method using static embeddings (Word2Vec and FastText). In the second scenario of enhancing TextRank using TF-IDF-weighted word embeddings, the results are also consistent with those from previous work, highlighting that TF-IDF weighting is effective for static embeddings, but it disrupts the performance of contextual embeddings. Overall, our results confirm the robustness of the enhanced TextRank with weighted word embedding for Indonesian text summarization. Lastly, this work also highlights the importance of using domain-specific training data to optimize summarization performance.

FUNDING INFORMATION

This research was funded by the Directorate of Research and Development, Universitas Indonesia, under Hibah PUTI Pascasarjana 2024 (Grant No. NKB-28/UN2.RST/HKP.05.00/2024).

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Evi Yulianti	✓	✓		✓	✓	✓			✓	✓		✓	✓	✓
Piawai Said Umbara			✓	✓	✓	✓	✓	✓	✓		✓			

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.

REFERENCES

[1] S. Kemp, "Digital 2023: Indonesia," *Date Reportal*, 2023, Accessed: Jun. 19, 2024. [Online]. Available: https://datareportal.com/reports/digital-2023-indonesia
[2] N. Newman, R. Fletche, K. Eddy, C. T. Robertson, and R. K. Nielsen, "Digital news report 2023," *Reuter Institute*, 2023, Accessed: Jun. 22, 2024. [Online]. Available: https://reutersinstitute.politics.ox.ac.uk/digital-news-report/2023/indonesia
[3] S. R. Rahimi, A. T. Mozhdehi, and M. Abdolahi, "An overview on extractive text summarization," in *2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation, KBEI 2017*, 2017, pp. 54–62, doi: 10.1109/KBEI.2017.8324874.
[4] I. Mani, "Recent developments in text summarization," in *Proceedings of the tenth international conference on Information and knowledge management*, 2001, pp. 529–531, doi: 10.1145/502585.502677.
[5] M. Allahyari *et al.*, "Text summarization techniques: a brief survey," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 10, 2017, doi: 10.14569/ijacsa.2017.081052.
[6] R. Mihalcea and P. Tarau, "TextRank: bringing order into texts," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, EMNLP 2004 - A meeting of SIGDAT, a Special Interest Group of the ACL held in conjunction with ACL 2004*, 2004, pp. 404–411.
[7] L. Page and S. Brin, "The anatomy of a large-scale hypertextual web search engine," *Computer Networks*, vol. 30, no. 1–7, pp. 107–117, 1998, doi: 10.1016/s0169-7552(98)00110-x.
[8] F. Koto, J. H. Lau, and T. Baldwin, "Liputan6: a large-scale indonesian dataset for text summarization," *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, AACL-IJCNLP 2020*, 2020, pp. 598–608, doi: 10.18653/v1/2020.aacl-main.60.
[9] T. T. Hailu, J. Yu, and T. G. Fantaye, "A framework for word embedding based automatic text summarization and evaluation," *Information*, vol. 11, no. 2, 2020, doi: 10.3390/info11020078.
[10] A. Kazemi, V. P.-Rosas, and R. Mihalcea, "Biased TextRank: unsupervised graph-based content extraction," in *COLING 2020 - 28th International Conference on Computational Linguistics,* 2020, pp. 1642–1652, doi: 10.18653/v1/2020.coling-main.144.

[11] U. Barman, V. Barman, M. Rahman, and N. K. Choudhury, "Graph based extractive news articles summarization approach leveraging static word embeddings," in *2021 International Conference on Computational Performance Evaluation, ComPE 2021*, 2021, pp. 8–11, doi: 10.1109/ComPE53109.2021.9752056.
[12] R. Rani and D. K. Lobiyal, "A weighted word embedding based approach for extractive text summarization," *Expert Systems with Applications*, vol. 186, 2021, doi: 10.1016/j.eswa.2021.115867.
[13] S. Zaware, D. Patadiya, A. Gaikwad, S. Gulhane, and A. Thakare, "Text summarization using TF-IDF and TextRank algorithm," in *Proceedings of the 5th International Conference on Trends in Electronics and Informatics, ICOEI 2021*, 2021, pp. 1399–1407, doi: 10.1109/ICOEI51242.2021.9453071.
[14] E. Yulianti, N. Pangestu, and M. A. Jiwanggi, "Enhanced TextRank using weighted word embedding for text summarization," *International Journal of Electrical and Computer Engineering*, vol. 13, no. 5, pp. 5472–5482, 2023, doi: 10.11591/ijece.v13i5.pp5472-5482.
[15] S. Gupta, A. Sharaff, and N. K. Nagwani, "Biomedical text summarization: a graph-based ranking approach," in *Advances in Intelligent Systems and Computing*, 2022, pp. 147–156, doi: 10.1007/978-981-16-2008-9_14.
[16] D. Jain, M. D. Borah, and A. Biswas, "Summarization of lengthy legal documents via abstractive dataset building: an extract-then-assign approach," *Expert Systems with Applications*, vol. 237, 2024, doi: 10.1016/j.eswa.2023.121571.
[17] X. Guan, Y. Li, Q. Zeng, and C. Zhou, "An automatic text summary extraction method based on improved textrank and TF-IDF," *IOP Conference Series: Materials Science and Engineering*, vol. 563, no. 4, 2019, doi: 10.1088/1757-899X/563/4/042015.
[18] X. Zuo, S. Zhang, and J. Xia, "The enhancement of TextRank algorithm by using Word2Vec and its application on topic extraction," *Journal of Physics: Conference Series*, vol. 887, no. 1, 2017, doi: 10.1088/1742-6596/887/1/012028.
[19] S. S. Nath and B. Roy, "Towards automatically generating release notes using extractive summarization technique," in *Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE*, 2021, pp. 241–248, doi: 10.18293/SEKE2021-119.
[20] B. Bałcerzak, W. Jaworski, and A. Wierzbicki, "Application of textrank algorithm for credibility assessment," in *2014 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Workshops, WI-IAT 2014*, 2014, pp. 451–454, doi: 10.1109/WI-IAT.2014.70.
[21] K. Kurniawan and S. Louvan, "IndoSum: a new benchmark dataset for Indonesian text summarization," in *2018 International Conference on Asian Language Processing, IALP 2018*, 2018, pp. 215–220, doi: 10.1109/IALP.2018.8629109.
[22] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, 2013, pp. 1–12.
[23] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017, doi: 10.1162/tacl_a_00051.
[24] K. Kurniawan, "KaWAT: a word analogy task dataset for Indonesian," 2019, *arXiv:1906.09912*.
[25] B. Wilie *et al.*, "IndoNLU: benchmark and resources for evaluating Indonesian natural language understanding," in *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, 2020, pp. 843–857.
[26] S. Pan, Z. Li, and J. Dai, "An improved TextRank keywords extraction algorithm," in *ACM International Conference Proceeding Series*, 2019, pp. 1–7, doi: 10.1145/3321408.3326659.
[27] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "BERTscore: evaluating text generation with BERT," in *8th International Conference on Learning Representations, ICLR 2020*, 2020, pp. 1–43.
[28] A. Lavie and A. Agarwal, "METEOR: an automatic metric for MT evaluation with high levels of correlation with human judgments," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2007, pp. 228–231.
[29] E. C. Fein, J. Gilmour, T. Machin, and L. Hendry, *Statistics for research students: an open access resource with self-tests and illustrative examples*, University of Southern Queensland, 2021.

BIOGRAPHIES OF AUTHORS

Evi Yulianti is a lecturer and researcher at the Faculty of Computer Science, Universitas Indonesia. She received the B.Comp.Sc. degree from the Universitas Indonesia in 2010, the dual M.Comp.Sc. degree from Universitas Indonesia and Royal Melbourne Institute of Technology University in 2013, and the Ph.D. degree from Royal Melbourne Institute of Technology University in 2018. Her research interests include information retrieval and natural language processing. She can be contacted at email: evi.y@cs.ui.ac.id.

Piawai Said Umbara received his bachelor's degree in Computer Science from Universitas Indonesia in 2024. His research interests include machine learning and natural language processing. He can be contacted at email: piawai.said81@ui.ac.id.