Journal of
Information Systems Engineering
and Business Intelligence

Vol.11, No.2, June 2025
Available online at: http://e-journal.unair.ac.id/index.php/JISEBI

Incorporation of IndoBERT and Machine Learning Features to Improve the Performance of Indonesian Textual Entailment Recognition

Teuku Yusransyah Tandi [1)], Taufik Fuadi Abidin [2)*], Hammam Riza [3)]

[1)2)]*Department of Informatics, Universitas Syiah Kuala, Banda Aceh, Indonesia*
[1)] t.yusran@mhs.usk.ac.id, [2)] taufik.abidin@usk.ac.id

[3)]*National Research and Innovation Agency (BRIN), Jakarta, Indonesia*
[3)] hammam.riza@brin.go.id

Abstract

Background: Recognizing Textual Entailment (RTE) is a task in Natural Language Processing (NLP), used for question-answering, information retrieval, and fact-checking. The problem faced by Indonesian NLP is based on how to build an effective and computationally efficient RTE model. In line with the discussion, deep learning models such as IndoBERT-large-p1 can obtain high F1-score values but require large GPU memory and very long training times, making it difficult to apply in environments with limited computing resources. On the other hand, machine learning method requires less computing power and provide lower performance. The lack of good datasets in Indonesian is also a problem in RTE study.
Objective: This study aimed to develop Indonesian RTE model called Hybrid-IndoBERT-RTE, which can improve the F1-Score while significantly increasing computational efficiency.
Methods: This study used the Wiki Revisions Edits Textual Entailment (WRETE) dataset consisting of 450 data, 300 for training, 50 for validation, and 100 for testing, respectively. During the process, the output vector generated by IndoBERT-large-p1 was combined with feature-rich classifier that allowed the model to capture more important features to enrich the information obtained. The classification head consisted of 1 input, 3 hidden, and 1 output layer.
Results: Hybrid-IndoBERT-RTE had an F1-score of 85% and consumed 4.2 times less GPU VRAM. Its training time was up to 44.44 times more efficient than IndoBERT-large-p1, showing an increase in efficiency.
Conclusion: Hybrid-IndoBERT-RTE improved the F1-score and computational efficiency for Indonesian RTE task. These results showed that the proposed model had achieved the aims of the study. Future studies would be expected to focus on adding and increasing the variety of datasets.

Keywords: Textual Entailment, IndoBERT-large-p1, Feature-rich classifiers, Hybrid-IndoBERT-RTE, Deep learning, Model efficiency.

Article history: Received 6 December 2025, first decision 3 February 2025, accepted 9 June 2025, available online 22 July 2025

I. INTRODUCTION

Rapid technological advancements are enabling high-performance data processing capabilities, significantly accelerating the development and widespread application of artificial intelligence technologies, including Natural Language Processing (NLP) [1]. The technology is a popular artificial intelligence that helps understand language principles. Additionally, the development of NLP started in 1950 to connect artificial intelligence and linguistics. NLP uses various artificial intelligence methods to analyze text, allowing computers to understand natural language similar to humans [2]. However, the technology has challenges related to ambiguous sentences and semantic complexity that make its tasks very difficult. These challenges include Recognizing Textual Entailment (RTE), which is an NLP task that can classify the relationship between Hypothesis (H) and Text (T) as entailment or non-entailment [3]. When H and T are related, the sentence is considered an entailment. Consequently, when the sentence is not related, then the sentence is considered non-entailment [4]. For example:

T: Mary killed a spider.
H: The spider is dead.

* Corresponding author

ISSN 2443-2555 (online) 2598-6333 (print)
doi: http://dx.doi.org/10.20473/jisebi.11.2.173-186

From the mentioned sentences, the outcome can be concluded that H comes from T and the sentence is considered an entailment.

Background knowledge in performing RTE related to understanding the meaning of each word contained in the Text (T) and Hypothesis (H) is obtained from a large corpus, which is then used to train the model before being used in classification [5]. RTE study plays a crucial role in NLP as it represents semantic analysis that can be applied in various domains, such as Question Answering (QA), Machine Translation (MT), Information Extraction (IE), Information Retrieval (IR), Summarization (SUM), and others [6].

Various studies on RTE have been conducted in various domains, specifically for detecting misinformation and disinformation during Covid-19. This was performed through the FacTeR-Check study, which was an architecture designed to conduct fact-checking and propagation analysis semi-automatically [7]. The system uses XLM-RoBERTa to assess semantic similarity, perform Natural Language Inference (NLI), and extract information for fact-checking. The incorporation of NLI and FacTeR-Check shows that the RTE methods can identify false claims by analyzing the relationship between statements and verified facts. Similarly, the process is related to classifying spam emails that perform detection using a topic-based method and machine learning models. The methods include deep learning models such as BERT, which have a significant role in developing cybersecurity technology [8]. These studies explain how RTE can be applied in various domains, such as detecting hoax information and cybersecurity.

Despite RTE study for English growing rapidly, few studies are in Indonesian, including the availability of large-scale datasets and sophisticated as well as numerous pre-trained models. A major challenge concerning this discussion is the limited high-quality datasets that are essential for training good models. The Wiki Revisions Edits Textual Entailment (WRETE) dataset is available data for Indonesian RTE, consisting of 450 pairs of annotated texts. Different from RTE datasets built using crowd-sourced or question-answer corpora, WRETE is uniquely formed because it is obtained from the revision history of Indonesian Wikipedia, which is developed using a semi-supervised co-training algorithm. The method uses T and H texts for sentences before as well as after revision as the first point of view. In addition, the model uses the comments of the author after revision as the second point of view. This method is performed by using a small part of the data labeled as initial data. Subsequently, the co-training algorithm automatically labels the unlabeled data using two different classification models for each point of view.

WRETE dataset is unique because even minor changes to revised sentences can cause lexical adjustments that can significantly change the semantics of the sentence. In addition, the dependence on the revision history between Wikipedia editors can provide a lot of linguistic variation, making detecting how a sentence is an entailment or not-entailment more complex. This dataset is also small in size, which can be a challenge when studies want to train the model using a deep learning method. WRETE is a dataset used by IndoNLU, which is state-of-the-art for evaluating Indonesian NLP [9]. Further study is needed for Indonesian RTE due to the minimal dataset and the linguistic complexity of Indonesian language, such as exploring more syntactic or semantic features and applying similar methods to broader domains, including online news or social media.

Several studies have conducted Indonesian language NLP analyses related to RTE. The semi-supervised method using WRETE dataset and Long Short-Term Memory (LSTM) achieved an accuracy of 76% [10]. According to the IndoNLU leaderboard, F1-score for IndoBERT-large-p1 model is 84.14%. In 2020, Indonesian Natural Language Understanding (IndoNLU) designed an RTE model with WRETE dataset using a model based on Bidirectional Encoder Representations from Transformers (BERT) [11]. Following the discussion, BERT is an NLP known for its good performance and has become a standard in NLP models [12]. IndoNLU uses IndoBERT, which is a version of BERT that is specially trained using Indonesian language data [13]. According to the IndoNLU leaderboard, F1-score for IndoBERT-large-p1 model is 84.14% [9], [14], with a training time of 32 hours [14]. On the other hand, the mBERT model has a slightly better F1-score of 84.40% [9]. The transformer-based model performs well but requires high computing resources, causing challenges to implement in an environment with limited resources. There is also an alternative method called feature-rich classifier, which can classify textual entailment in Indonesian text using WRETE dataset [14]. The method conducts feature-engineering process where each sentence in the text is extracted from its linguistic features and then divided into several groups, such as distance-based, token-based, part-of-speech (POS) tags, and negation-based features. This model can significantly reduce the training time to 0.0109 seconds but has the disadvantage of a lower F1-score of 79.65%.

Previous studies using deep learning obtained high F1-score values but required very high computing resources compared to feature-rich method, which was very efficient in computing power but not better for F1-score. Based on the trade-off, both methods need a hybrid model that uses the advantages of each procedure, allowing an efficient model but has a good F1-score for Indonesian RTE. The hybrid method that combines deep and machine learning has

been proven to improve performance of the model [15]. However, most studies often focus on increasing the F1-score without discussing the efficiency when training the model.

This study aimes to propose IndoBERT-large-p1 model, modified by incorporating the method with machine learning features in feature-rich classification. The incorporation aims to improve model performance without neglecting efficiency when training the model, moderating the computing resources. Furthermore, Hybrid-IndoBERT-RTE model is expected to balance F1-score and training time to provide a strong and reliable solution for WRETE dataset.

II. METHODS

A. Datasets

This study used WRETE dataset, which had been applied by previous studies in [14] and [9]. WRETE was a dataset in the form of a .csv file consisting of 100 test, 300 training, and 50 validation data. In addition, the dataset was categorized as NotEntail and Entail_or_Paraphrase. Table 1 showed the distribution of the dataset against all labels.

TABLE 1
DISTRIBUTION OF DATASET LABELS

Dataset	Entail	Not Entail
Training Data	183 pairs	117 pairs
Testing Data	61 pairs	39 pairs
Validation Data	31 pairs	19 pairs

Each instance in the dataset consisted of two sentences, where sent_A represented the premise and sent_B signified the hypothesis. The model aimed to determine how the hypothesis included the premise, as Table 2 showed an example of a dataset entry.

TABLE 2
DATASET EXAMPLE

sent_A	sent_B	Category	Label
Mumps is a disease caused by a virus.	Mumps is a type of contagious disease.	pandemic	Entail_or_Paraphrase
In Indonesian, it was translated as: *Beguk merupakan satu penyakit yang disebabkan oleh virus*	In Indonesian, it was translated as*: Beguk adalah suatu jenis penyakit berjangkit.*	In Indonesian, it was translated as: *pandemi*	

In Table 2, the sent_A column contained the premise, while sent_B comprised the hypothesis, as both were in Indonesian. The label column showed how the second sentence included the first. The category column provided additional metadata about the type of sentence pairs. However, the column was not directly used in the classification model or evaluation.

B. Prepocessing

During this stage, the dataset was passed through preprocessing, where every important piece of information was extracted [16]. The process included (1) ensuring there were no double spaces in the sentence. (2) Changing sentences to lowercase letters to be consistent and uniform for the data in the dataset. (3) Removing all punctuation, such as , ?! @ # $%. (4) Changing all sentences into a basic form. (5) Removing stopwords contained in the document. This preprocessing method ensured the dataset was clean, consistent, and well-structured for further analysis and modeling [17]. Relating to the dicussion, the preprocessing process flow was shown in Fig. 1.

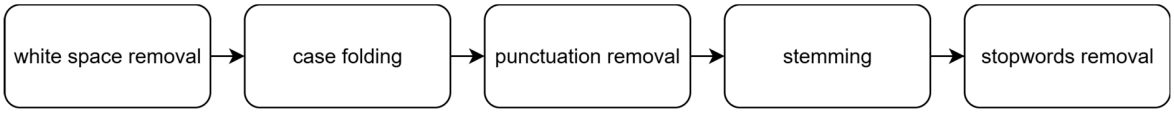


Fig. 1: Dataset Preprocessing

C. Feature Extraction

In feature-rich classifier [14], features extraction used various machine learning algorithms. These features were divided into five categories, namely distance-based, token-based, POS-tag, negation, and BLEU score features. Distance-based features, such as Word Mover's Distance (wmd), Levenshtein Distance (lev_dist), Jaccard Similarity

(jaccard_sim), and Euclidean Distance (euclidean_dist), were used to measure semantic differences or characters in sentences. Additionally, Wmd used pretrained word2vec with a score of 0, showing equality in semantics [18]. Lev_dist calculated the minimum number of edit operations required to change a sentence into another [19]. Following the discussion, Jaccard_sim measured the ratio of similar words to total unique words between two sentences. Euclidean distance calculated the TF-IDF between sentences, where a lower value showed a better match [20]. Token-based features consisted of raw token similarity (raw_similar_tok), preprocessed token similarity (similar_tok), and the difference between the number of tokens of the two sentences (raw_diff_tok and diff_tok) [14]. These features measured the difference in sentence length between T and H and also the degree of overlap between words. Features such as the number of tokens in T and H (num_tok_T and num_tok_H), same_unigram, same_bigram, and vec_gram were also included. Raw token similarity counted the tokens that matched the two raw sentences. Meanwhile, same_bigram counted the number of bigrams appearing between the two sentences. POS-tag features were used for syntactic pattern analysis, consisting of matching POS-tag percentage (match_postag_pct), identical POS-tag sequences (match_postag, same_postag), and exact POS–token matches (fullmatch_postag). These features were extracted using Indonesian POS-tagger derived from Conditional Random Fields trained with the dataset from [21], which measured the grammatical level of a text. The negation features identified how a negation token appeared between two sentences to help the model detect and distinguish meaning shifts caused by negation words [22]. These features counted the number of negation words in both T and H, as well as the differences. BLEU score features that comprised BLEU-1 (bleu1gram) and BLEU-2 (bleu2gram), were used to measure n-grams by considering the sequence of words. Different from token-based similarity, BLEU calculated in more detail because the model observed the similarity at the phrase level and combined it with the sequential structure. BLEU-3 (bleu3gram) and BLEU-4 (bleu4gram) were also calculated to ensure better calculations as well as to evaluate more comprehensive syntax [23]. In line with the discussion, all these features were combined before classification. The combining process was based on experimental evaluation by assessing the contribution of the features to the F1-score of the model. Table 4 showed selections from several examples of extracted features during the process.

D. Modelling

The next step was to incorporate IndoBERT-large-p1 with machine learning features obtained from features extraction using feature-rich classifier. This incorporation then produced a modified architecture called Hybrid-IndoBERT-RTE, as shown in Fig. 2.

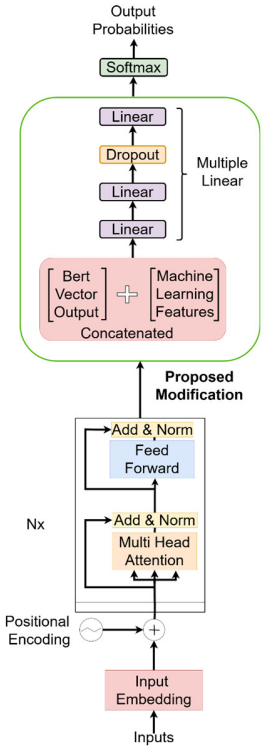


Fig. 2 Proposed Modification (adapted from [11] with modifications)

The text data in the dataset experienced vector extraction using IndoBERT-large-p1 first, which was obtained from pooler_output. These vectors were combined with machine learning features during the process. Subsequently, both vectors were combined using the concatenation method to produce a combined vector with a larger dimension, as shown in Fig. 3.

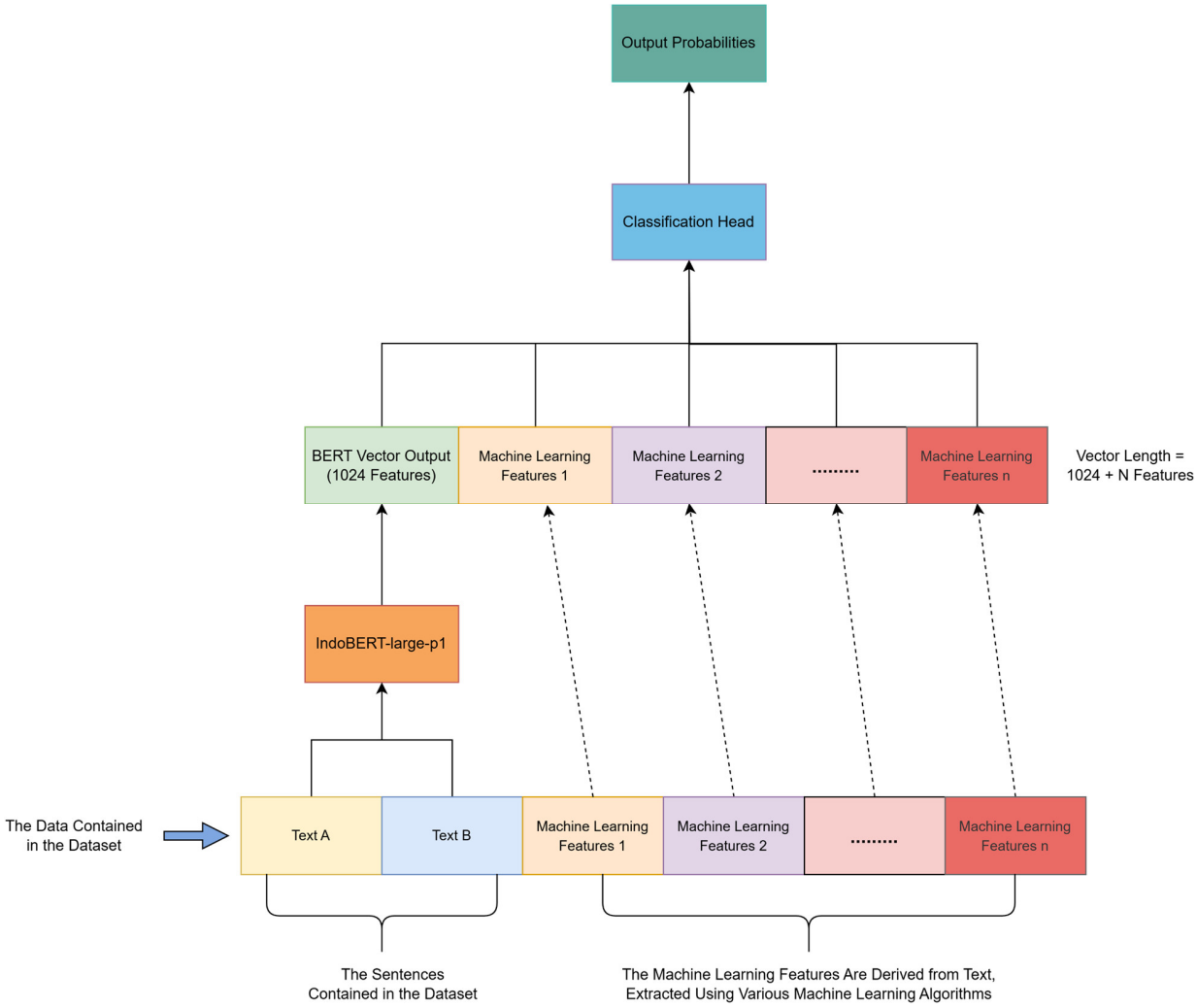


Fig. 3 Integration between BERT vector and machine learning features

For example, the vector obtained from IndoBERT-large-p1 with 1024 dimensions was combined with one from machine learning consisting of 4 features, leading to a combined vector of 1028 values. A sample of the process during the analysis was shown as follows.

Input Text: (text A : “it is very cold tonight”, text B: “it is raining outside”)

Vector extracted using IndoBERT-large-p1 : [0.25, -0.13,..., 0.05] (1024 values)

Additional machine learning features (e.g., wmd,lev_dist) : [0.8, 0.9,0.7,0.6] (4 values)

Final concatenated vector : [0.25, -0.13,…,0.05, 0.8, 0.9 0.7,0.6] (total 1028 values).

The combination of these vectors maintained the semantic richness value between IndoBERT-large-p1 and machine learning features that had been combined. This blend improved classification performance because linguistic perceptions between vectors had been joined. During the study, the model obtained more relevant information and captured all important patterns that might be unacquired when it only relied on the vectors from IndoBERT-large-p1 with a combination enriching the textual representation. Different from other transformer-based models that modified and represented features by adding additional layers or projections, this method maintained the original character of

the vectors from both features. This process was conducted to maintain the contextual meaning between IndoBERT-large-p1 and machine learning features without distortion.

The combination provided an easy and computationally efficient way of incorporation and avoided complexities such as transformation-based methods. In performing the combination, adjustments were required to the classification head because the original IndoBERT-large-p1 architecture (1024 inputs) had a different number of inputs than that in the Hybrid-IndoBERT-RTE architecture. Additionally, the classification head in the Hybrid-IndoBERT-RTE architecture was dynamic, depending on the combined dimensions of the vector.

The classification head consisted of one input layer that accepted the incorporated vector. These three hidden layers understood the patterns from the combined features, and each hidden layer comprised linear levels to capture complex relationships in the data. Furthermore, a dropout layer was added to prevent overfitting by randomly deactivating particular neurons during training, as shown in Fig. 4.

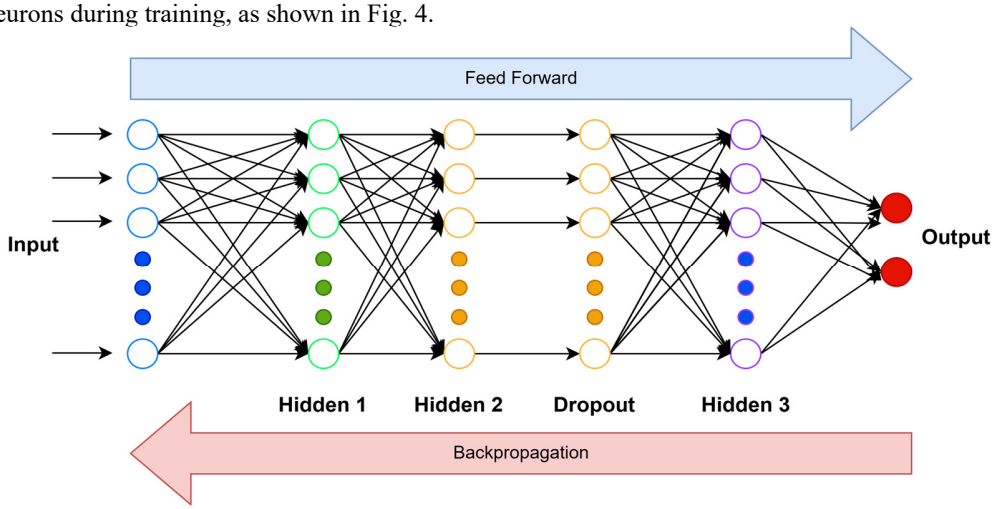


Fig. 4 Classification head for Hybrid-IndoBERT-RTE

The output layer produced the final prediction based on the data representation processed through the hidden and dropout layers. When training, the part trained during the process was the classification head only. At the same time, IndoBERT-large-p1 was frozen and became features extractor to convert words into vectors, making the training process efficient, as shown in Fig. 5.

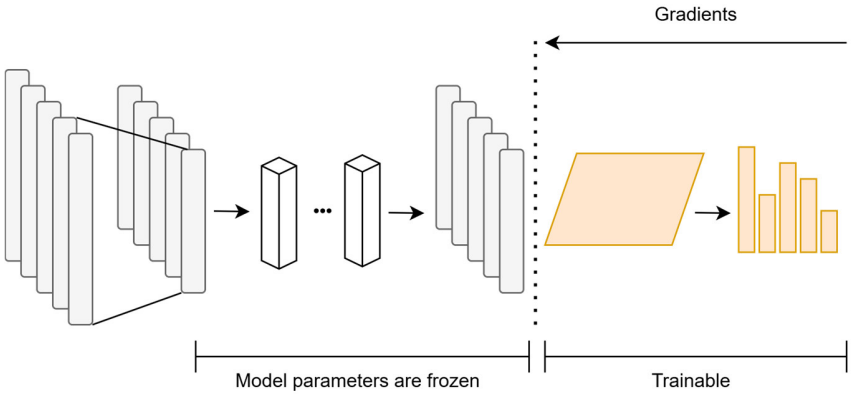


Fig. 5 Illustration of features extraction using IndoBERT-large-p1 and the modified classification head (adapted from [24])

After incorporating the frozen BERT output as well as machine learning features, the model was trained, and the hyperparameter tuning process was conducted on the classification head. Some of these hyperparameter adjustments included (1) selecting the best optimizer to adjust the weight values in the classification head, such as Adam or AdamW. During this analysis, the best optimizer was the one that produced the best results. (2) Validation data was used to monitor performance of the classification head during the training process to avoid overfitting. (3) The batch size was adjusted to the available memory capacity. During the analysis, the size affected the stability and speed of

training. (4) The optimal number of epochs was observed from learning curve. (5) The correct learning rate was selected to speed up the training process without convergence problems.

An optimization process was performed on machine learning features and the classification head, in addition to tuning hyperparameters. This process included (1) Features selection of available machine learning features. This process was important because using relevant features improved model performance and efficiency [25]. (2) The classification head was also adjusted for accepting combined vector input. The addition or reduction of neurons or layers was adjusted to the number of features selected from features selection results. This process was conducted because the appropriate number of layers improved model performance [26]. Excessive layers led to overfitting, while inadequate layers caused underfitting. The dropout on each hidden layer was also arranged to form the best configuration.

E. Evaluation Procedure

The evaluation in this study followed the same procedure as the previous studies to ensure a fair comparison. Similar to previous studies, this analysis used the F1-score as the primary performance metric and the testing dataset to validate the model.

III. RESULTS

A. Preprocessing Datasets

Preprocessing was conducted before the model training process was performed. Table 3 showed examples of data that had not been preprocessed during the analysis. After being processed, both tables ("sent_A" and "sent_B") contained the original version of the sentence, and ("preprocessed_A" and "preprocessed_B") comprised the processed sentences.

The preprocessing steps that were applied followed the method explained previously to ensure the consistency of the data in the dataset before features extraction was conducted. After preprocessing, the next step was features extraction stage using feature-rich classifier method. Features extracted included Word Mover's Distance (Wmd), Jaccard Similarity (jaccard_sim), and several others, such as raw token similarity, POS-tag matching, Levenshtein distance, and BLEU score. Moreover, the extraction results were shown in Table 4, where only a small part of the data was presented for clarity.

TABLE 3
DATA IN THE DATASET BEFORE AND AFTER PREPROCESSING

sent_A	sent_B	preprocessed_A	preprocessed_B
In 1964, after holding various positions in the Egyptian government, he was selected by President Gamal Abdel Nasser to serve as Vice President. In Indonesian, it was translated as: *Pada 1964 , setelah memegang berbagai jabatan dalam pemerintahan Mesir , ia dipilih oleh Presiden Gamal Abdel Nasser untuk menjabat sebagai Wakil Presiden .*	In 1964, after holding various positions in the Egyptian government, he was selected by President Gamal Abdel Nasser to serve as a cleaning service. In Indonesian, it was translated as: *Pada 1964 , setelah memegang berbagai jabatan dalam pemerintahan Mesir , ia dipilih oleh Presiden Gamal Abdel Nasser untuk menjabat sebagai cleaning service .*	1964 held various positions govern mesir selected President Gamal Abdel Nasser held Vice President In Indonesian, it was translated as: *1964 pegang bagai jabat perintah mesir pilih presiden Gamal Abdel Nasser jabat bagai wakil presiden.*	1964 held various positions govern mesir selected President Gamal Abdel Nasser held cleaning service In Indonesian, it was translated as: *1964 pegang bagai jabat perintah mesir pilih presiden Gamal Abdel Nasser jabat bagai cleaning service.*

TABLE 4
EXAMPLE OF FEATURES EXTRACTION RESULTS

sent_A	sent_B	Wmd	jaccard_sim	Label
1964 held various positions govern mesir select President Gamal Abdel Nasser held Vice President In Indonesian, it was translated as: *1964 pegang bagai jabat perintah mesir pilih presiden gamal abdel nasser jabat bagai wakil presiden*	1964 held various positions govern mesir select President Gamal Abdel Nasser held a cleaning service In Indonesian, it was translated as: *1964 pegang bagai jabat perintah mesir pilih presiden gamal abdel nasser jabat bagai cleaning service*	3,8910176850642800	0,7857142857142860	NotEntail

B. Result

Features extraction process produced several machine learning features such as "wmd," "raw_similar_tok," "bleu1gram," and "same_bigram." Each of these features was tested individually to see its relevance. The process of selecting machine learning features was performed by empirical evaluation, where performances of the features were tested one after another for its contribution to the classification performance. This method allowed the study to identify the most relevant machine learning features that were combined with deep learning representations. The results of the empirical evaluation of each machine learning features were shown in Table 5, as features including WMD and BLEU-1 contributed significantly to the classification performance. In addition, token-based features such as raw_similar_tok and same_bigram were also very effective in directly detecting lexical similarity.

TABLE 5
EXAMPLE OF MACHINE LEARNING FEATURES TESTING

Machine Learning Features	Training Time (seconds)	F1-Score
wmd, raw_similar_tok, bleu1gram, bleu2gram	172,87	72%
wmd, raw_similar_tok, jaccard_sim, match_postag_pct	171,42	53%
wmd, raw_similar_tok, jaccardsim, matchpostagpct, matchpostag	171,70	74%
wmd, raw_similar_tok, jaccard_sim, match_postag_pct, match_postag, same_postag, lev_dist, same_bigram, bleu1gram, bleu2gram	172,95	76%
wmd, raw_similar_tok, bleu1gram, same_bigram	175,45	**85%**
wmd, raw_similar_tok	173,19	71%
jaccard_sim, raw_similar_tok	173,00	66%
jaccard_sim, lev_dist	172,00	68%
match_postag_pct, lev_dist	173,80	42%
match_postag_pct	172,57	69%
Wmd	171,49	76%
bleu1gram	172,75	70%
same_bigram	173,27	73%
bleu1gram, lev_dist, same_bigram	**169,92**	63%
wmd, raw_similar_tok, bleu1gram	171,65	67%

The training process was conducted by carefully searching for hyperparameters. The best hyperparameters for this study were as follows: batch size = 32, maximum sequence length (max_seq_len) = 512, number of epochs = 14, learning rate = 0.01000001, and Adam optimizer, as training took 175.45 seconds. These hyperparameters were determined after a series of manual tests where various combinations were attempted repeatedly. Table 6 showed that the selection process was performed by adjusting the parameters gradually to obtain the best.

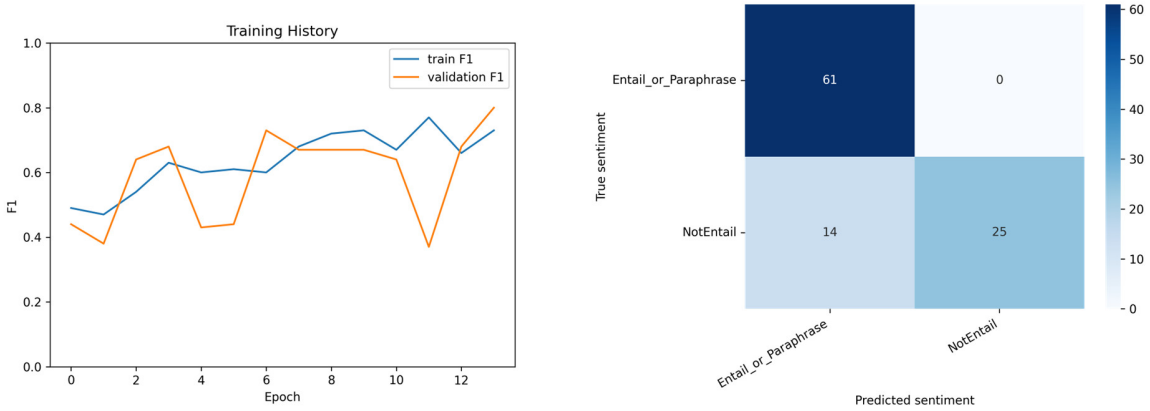
TABLE 6
EXAMPLE OF HYPERPARAMETER TUNING

Hyper Parameter	Training Time (seconds)	F1-score
batch size = 32, *max_seq_len* = 512, *epoch* = 14, *learning rate* = 0.01000001 and *optimizer* = adam	175,45	**85%**
batch size = 32, *max_seq_len* = 512, *epoch* = 13, *learning rate* = 0.01000001 and *optimizer* = adam	160,94	64%
batch size = 32, *max_seq_len* = 512, *epoch* = 12, *learning rate* = 0.01000001 and *optimizer* = adam	146,96	35%
batch size = 32, *max_seq_len* = 512, *epoch* = 11, *learning rate* = 0.01000001 and *optimizer* = adam	133,13	70%
batch size = 32, *max_seq_len* = 512, *epoch* = 10, *learning rate* = 0.01000001 and *optimizer* = adam	**120,56**	74%
batch size = 32, *max_seq_len* = 512, *epoch* = 14, *learning rate* = 0.01000000 and *optimizer* = adam	175,76	64%
batch size = 32, *max_seq_len* = 512, *epoch* = 14, *learning rate* = 0.01000002 and *optimizer* = adam	175,11	66%
batch size = 32, *max_seq_len* = 512, *epoch* = 14, *learning rate* = 0.01000003 and *optimizer* = adam	169,89	76%
batch size = 32, *max_seq_len* = 512, *epoch* = 14, *learning rate* = 0.01000004 and *optimizer* = adam	172,99	73%
batch size = 32, *max_seq_len* = 512, *epoch* = 14, *learning rate* = 0.01000005 and *optimizer* = adam	171,97	69%
batch size = 32, *max_seq_len* = 512, *epoch* = 14, *learning rate* = 0.1 and *optimizer* = adam	170,53	70%
batch size = 32, *max_seq_len* = 512, *epoch* = 14, *learning rate* = 0.01 and *optimizer* = adam	172,19	64%
batch size = 32, *max_seq_len* = 512, *epoch* = 14, *learning rate* = 0.001 and *optimizer* = adam	172,51	75%
batch size = 32, *max_seq_len* = 512, *epoch* = 14, *learning rate* = 0.0001 and *optimizer* = adam	171,60	71%
batch size = 32, *max_seq_len* = 512, *epoch* = 14, *learning rate* = 0.00001 and *optimizer* = adam	173,40	79%

Major influential hyperparameters included the epoch and learning rate because both parameters significantly affected how the model learned. Increasing the epoch value increased learning time of the model, but the impact on its performance was not entirely linear. Following the discussion, a higher epoch value occasionally contributed well, but an excess caused poor results due to overfitting.

Learning rate had a crucial role in optimizing acquiring process during the study. A slight change in learning rate

value led to a significant difference, proving that changes in this rate were very sensitive. Learning rates that were significantly higher speed up model training and allowed the model to pass its maximum point. Meanwhile, substantially low learning rates reduced the training process and prevented the model not reaching its maximum point. Based on this test, systematically and optimally testing hyperparameters was necessary to balance efficiency as well as optimal model performance.



(a) (b)

Fig. 6 (a) Graph of the effect of epochs on F1-score; (b) Confusion matrix for testing data

The training process using the Hybrid-IndoBERT-RTE architecture produced an F1-score graph for training and validation, as shown in Fig. 6a. This graph represented the effect of the number of epochs and the improvement of model performance with the F1-score metric during training, both in the training and validation processes. Based on this graph, the analysis observed how the model learned according to the training data provided, adjusting each weight value of each internal parameter to optimize performance. Validation curves assessed how a model learned from data that the model had used without causing overfitting or underfitting. After completing the training process, the model was evaluated with test data consisting of 100 samples. The results of this test signified an F1-score value of 85%, with a confusion matrix, as shown in Fig. 6b. This confusion matrix provided an overview of how the model classified the given test data, such as errors and correct predictions.

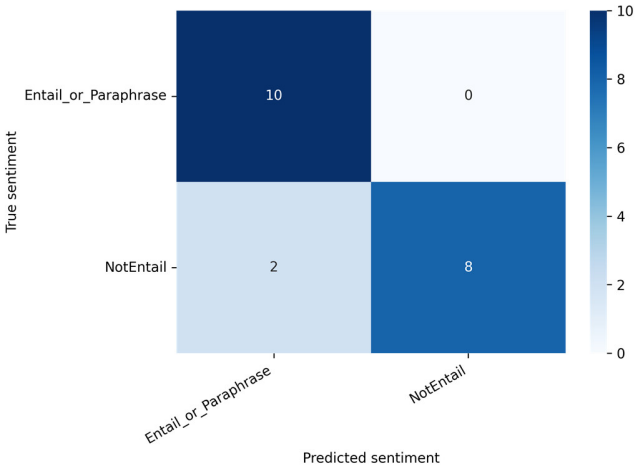


Fig. 7 Confusion matrix for 20 random data samples.

Testing was performed using 20 random datasets in the test data, 10 for each label. This additional evaluation was conducted to observe the consistency of the model in handling various types of input. The results of this random testing showed an F1-score of 90%, signifying that the model performed well in some cases. The confusion matrix in Fig. 7 showed performance of the model on the randomly tested data.

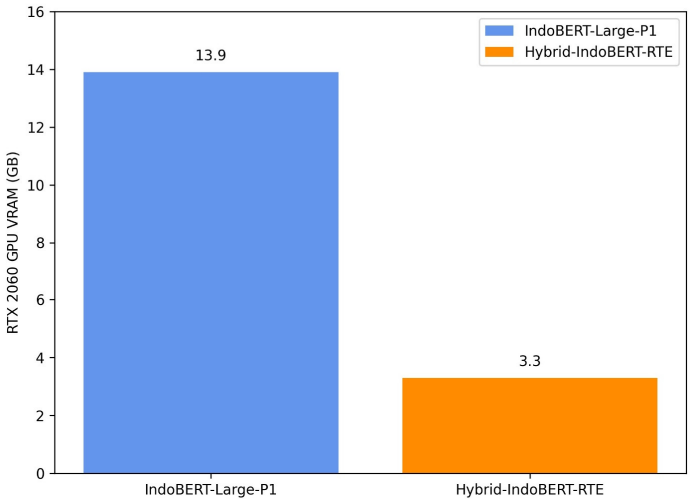


Fig. 8 GPU VRAM usage

The use of GPU memory resources showed a good increase in efficiency. Fig. 8 showed that Hybrid-IndoBERT-RTE significantly reduced GPU memory consumption. During the study, Hybrid-IndoBERT-RTE required 3.3 GB, while IndoBERT-large-p1 required 13.9 GB. Fig. 9 showed a comparison of the model training time used during the study. Hybrid-IndoBERT-RTE was 47.15 times faster, reducing training time from 8675.17 seconds to 183.93 seconds when trained with 1500 sentence pairs. These results provided a clear representation of the efficiency and performance of the model.

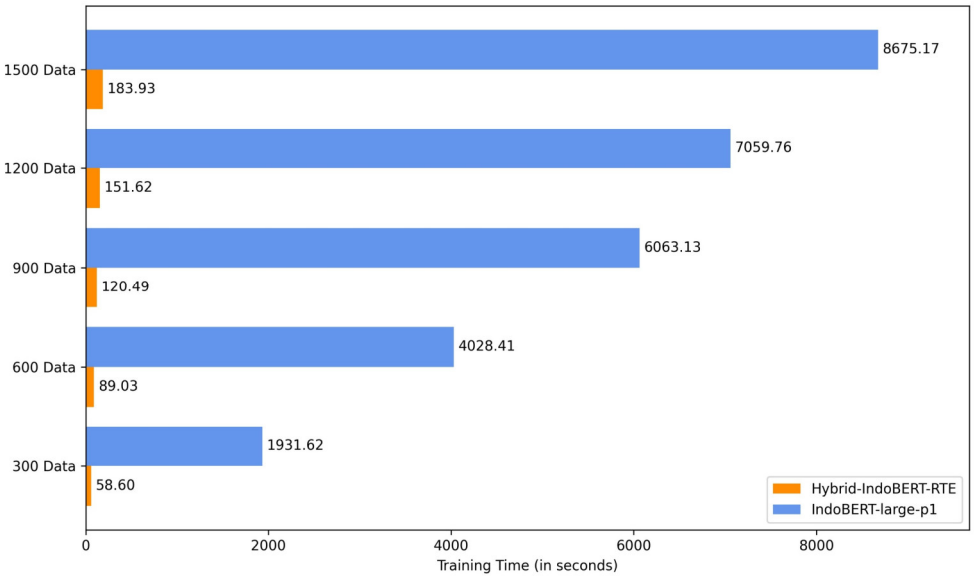


Fig. 9 Comparison of model training speed based on data size (in seconds)

The pre-trained IndoBERT-large-p1 model and Hybrid-IndoBERT-RTE model were compared. In the pre-trained IndoBERT-large-p1 model, there was an error in predicting the results, as shown in Table 7. During the analysis, both Text A and Text B were originally in Indonesian. The Hybrid-IndoBERT-RTE model showed accurate prediction results with better softmax probability values. This achievement signified that the Hybrid-IndoBERT-RTE model had learned and understood data patterns appropriately, leading to good predictions. In addition, the increasing softmax probability value showed that the Hybrid-IndoBERT-RTE model had more confidence in its predictions concerning the classification between entailment and non-entailment. This result proved that the model had learned and understood the pattern well, providing accurate prediction values.

TABLE 7
COMPARISON BETWEEN PRE-TRAINED AND TRAINED MODELS

Text A	Text B	Model	Softmax	Label
Electrons only occupy 0.06% total mass of the atom. In Indonesian, it was translated as: *Elektron hanya menduduki 0,06 % massa total atom.*	Electrons only take up 0.06% total mass of the atom. In Indonesian, it was translated as: *Elektron hanya mengambil 0,06 % massa total atom.*	Pre-trained	59.347%	NotEntail
		Trained	80.197%	Entail_or_Paraphrase
Now, no one knows exactly when history began. In Indonesian, it was translated as: *Sekarang , tidak ada yang tahu pasti kapan sejarah dimulai.*	History began at the beginning of human creation. In Indonesian, it was translated as: *Sejarah dimulai pada awal penciptaan manusia.*	Pre-trained	52.314%	Entail_or_Paraphrase
		Trained	83.608%	NotEntail
Yusran works at kominfo In Indonesian, it was translated as: *Yusran kerja di kominfo.*	Yusran is not yet employed. In Indonesian, it was translated as: *Yusran belum bekerja*.	Pre-trained	59.593%	Entail_or_Paraphrase
		Trained	84.431%	NotEntail
Yusran likes to play with mobile phones. In Indonesian, it was translated as: *Yusran suka bermain handphone.*	Yusran doesn't like to play with gadgets. In Indonesian, it was translated as: *Yusran tidak suka bermain dengan gawai.*	Pre-trained	65.262%	Entail_or_Paraphrase
		Trained	81.548%	NotEntail
Gadgets can damage eyes. In Indonesian, it was translated as: *Gawai dapat merusak mata.*	Gadgets can improve eye health. In Indonesian, it was translated as: *Gawai dapat menyehatkan mata.*	Pre-trained	66.469%	Entail_or_Paraphrase
		Trained	64.383%	NotEntail

IV. DISCUSSION

Based on the results of this study, the Hybrid-IndoBERT-RTE architecture produced a model with high performance in predicting RTE for Indonesian language using WRETE dataset. Table 8 compared this study and previous works [9] [14], which also used WRETE dataset and evaluated the models using the F1-score metric. To ensure a fair comparison, IndoBERT-large-p1 performance reported in this study referred to the source used by [14], and the best result listed on the IndoNLU leaderboard [9]. In addition to having an excellent F1-score, the Hybrid-IndoBERT-RTE architecture was also more efficient than IndoBERT-large-p1 architecture used by IndoNLU.

TABLE 8
EXPERIMENTS RESULT

Experiments	F1-Score
Features Rich [14]	79.65%
IndoBERT-large-p1 [9]	84.14%
mBERT [9]	84.40%
Hybrid-IndoBERT-RTE (Proposed)	**85%**

During training, IndoBERT-large-p1 architecture required full backpropagation and forward pass from both the model and the classification head, which led to very high GPU usage as well as longer training time. As a comparison, a benchmark was performed using a batch size = 16 and epoch = 5 with the same dataset. The training dataset of 300 pairs of sentences was replicated up to 1500 pairs of data sentences. This method simulated that training was performed with a large dataset during the analysis. Fig. 8 showed that there was a significant difference in the use of GPU resources. IndoBERT-large-p1 required 13.9 GB of VRAM, while Hybrid-IndoBERT-RTE only needed 3.3 GB of VRAM. The method showed that resource consumption was reduced by training only the classifier head. This method improved computational efficiency and showed the potential use of BERT as features extractor, making the model more accessible to studies with limited computing resources. The impact of high GPU usage on training speed was shown in Fig. 9. When the data was about 300 pairs of sentences, the Hybrid-IndoBERT-RTE model had a training time 32.96 times faster, which was 58.6 seconds compared to IndoBERT-large-p1 model with 1931.62 seconds.

When the amount of data increased to 600 pairs of sentences, Hybrid-IndoBERT-RTE model had a training time 45.24 times faster, which was 89.03 seconds compared to IndoBERT-large-p1 with 4028.41 seconds. Having a data

amount of 900 pairs of sentences, the model had a training time 50.32 times faster, which was 120.49 seconds compared to IndoBERT-large-p1 with 6063.13 seconds. Moreover, when the dataset size reached 1200 pairs of sentences, the Hybrid-IndoBERT-RTE model trained 46.55 times faster, completing in 151.62 seconds compared to 7059.76 seconds for IndoBERT-large-p1. Having 1500 pairs, the Hybrid model maintained its speed average, training in 183.93 seconds, 47.16 times faster than IndoBERT-large-p1, which took 8675.17 seconds. Based on the data mentioned earlier, the Hybrid-IndoBERT-RTE model completed the training time 44.44 times faster and with more efficient use of GPU VRAM resources 4.2 times compared to IndoBERT-large-p1 model in the IndoNLU study. More efficient use of resources and training time provided significant economic benefits because reducing energy consumption moderated costs. In addition, reducing energy consumption minimized carbon emissions caused by high electricity consumption [27]. The carbon footprint of a model was greatly influenced by the time required to train the model. In line with this discussion, the increasing use of computing resources, such as the number of cores on the CPU, also formed a larger carbon footprint [28].

Having more effective training time, studies could perform faster iterations to make the model development and optimization more efficient. The iteration opened up more access to AI technology for individuals and organizations, including small companies as well as the world of education. Therefore, increasing the efficiency of these resources also supported the development and innovation of artificial intelligence to be faster.

Despite the proposed model achieving a better F1-score than the previous model, there were still errors in its classification in some cases, signifying that the model had limitations in understanding the textual entailment relationship. Table 9 showed some model performance errors in classifying the textual entailment relationship of Indonesian sentences.

TABLE 9
EXAMPLES OF MISCLASSIFIED INSTANCES

Text A	Text B	Label	Prediction
Mary ke kantor menggunakan mobil. In English, it was translated as: Mary goes to the office by car.	*Mary menggunakan kendaraan roda empat.* In English, it was translated as: Mary uses a four-wheeled vehicle.	Entail_or_Paraphrase	NotEntail
Misalnya anak sekolah di tingkat SMA masih menggunakan bahasa Indonesia untuk ujian akhir. In English, it was translated as: For example, high school students still use Indonesian for their final exams.	*Misalnya anak sekolah di tingkat SMA masih menggunakan bahasa Melayu untuk ujian akhir.* In English, it was translated as: For example, high school students still use Malay for their final exams.	NotEntail	Entail_or_Paraphrase

The misclassification of the model showed that it had difficulty understanding subtle semantic differences. A possible cause was the limited size and diversity of the dataset. Small and homogeneous dataset could lead the model to overfit particular patterns during training, reducing its generalization performance.

Despite the limitations, the model could be classified better than previous models. The increase in F1-score did not fully reflect the overall ability of the model. Further evaluation including qualitative error analysis, would be needed to identify the linguistic challenges experienced. Features selection process in this study was still empirical, based on the increase in model performance obtained when selecting machine learning features. Despite the efficiency of this method, the model optimization process was limited. Future studies would be expected to explore a more systematic process of selecting machine learning features, such as ablation investigations or conducting features importance analysis to understand relevant features better. Selecting machine learning features with a strong theory would also be expected to increase the reliability of the model to be built.

Several factors could affect the validity of this study in the context of the topic. First, the dataset used in the study was limited, and it affected the ability of the model to perform Indonesian language textual entailment. Future studies should explore the dataset, allowing a larger and more diverse amount of data to improve the classification ability of the model against various types of data. Second, dependency because the trained model used was a pre-trained model, which could cause bias during learning. This bias was natural because the average available model was a pre-trained result. As a result, the model optimization should be conducted in more detail. Third, the ability and effectiveness of the model proposed in the study did not fully reflect Indonesian linguistic textual entailment because it was only tested using one dataset.

V. CONCLUSIONS

In conclusion, this study introduced a model called Hybrid-IndoBERT-RTE, a new method that incorporated IndoBERT-large-p1 with feature-rich classifier to improve performance of Indonesian RTE. The proposed model had an F1-score of 85% when tested with the WRETE dataset, outperforming previous studies, and made the computational load significantly more efficient. Using frozen BERT and only training the classification head, GPU memory usage was reduced by 4.2 times, and training time was also 44.44 times faster, making the model development process more efficient as well as easier. In addition, the analysis showed that using a hybrid method balanced performance and computational efficiency, allowing other studies to optimize models with limited resources in the future. The results showed the potential for using BERT as features extractor, which formed opportunities for further studies. As a result, future studies could explore larger datasets and perform features engineering to obtain new and more relevant machine learning features, including attempting other transformer-based architectures to improve Indonesian RTE model.

Author Contributions: *Teuku Yusransyah Tandi*: Conceptualization, Methodology, Software, Investigation, Data Curation, Writing - Original Draft. *Taufik Fuadi Abidin*: Conceptualization, Methodology, Writing - Review and Editing, Supervision. *Hammam Riza*: Writing - Review and Editing.

Funding: This research received no specific grant from any funding agency.

Conflicts of Interest: The authors declare no conflict of interest.

Data Availability: The data will be made available for sharing upon request.

Informed Consent: There were no human subjects.

Institutional Review Board Statement: Not applicable.

Animal Subjects: There were no animal subjects.

ORCID:
Teuku Yusransyah Tandi: https://orcid.org/0009-0004-4145-5017
Taufik Fuadi Abidin: https://orcid.org/0000-0002-3859-6706
Hammam Riza: https://orcid.org/0000-0002-5449-6828

REFERENCES

[1] S. Qiu, Q. Liu, S. Zhou, and W. Huang, "Adversarial attack and defense technologies in natural language processing: A survey," Jul. 01, 2022, *Elsevier B.V.* doi: 10.1016/j.neucom.2022.04.020.

[2] P. M. Nadkarni, L. Ohno-Machado, and W. W. Chapman, "Natural language processing: An introduction," Sep. 2011. doi: 10.1136/amiajnl-2011-000464.

[3] V. Sitzmann, M. Marek, and L. Keselman, "Multimodal Natural Language Inference Final Report," in *Stanford CS224U: Natural Language Understanding*, 2016.

[4] N. Reshmi S and Shreelekshmi, "Textual Entailment based on Semantic SimilarityUsing WordNet," in *2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, IEEE, 2019. doi: 10.1109/ICICICT46008.2019.8993180.

[5] A. Paramasivam and S. J. Nirmala, "A survey on textual entailment based question answering," Nov. 01, 2022, *King Saud bin Abdulaziz University*. doi: 10.1016/j.jksuci.2021.11.017.

[6] S. R. Joseph, H. Hlomani, K. Letsholo, F. Kaniwa, and K. Sedimo, "Natural Language Processing: A Review," 2016. [Online]. Available: http://www.euroasiapub.org

[7] A. Martín, J. Huertas-Tato, Á. Huertas-García, G. Villar-Rodríguez, and D. Camacho, "FacTeR-Check: Semi-automated fact-checking through semantic similarity and natural language inference," *Knowl Based Syst*, vol. 251, Sep. 2022, doi: 10.1016/j.knosys.2022.109265.

[8] F. Jáñez-Martino, R. Alaiz-Rodríguez, V. González-Castro, E. Fidalgo, and E. Alegre, "Classifying spam emails using agglomerative hierarchical clustering and a topic-based approach," *Appl Soft Comput*, vol. 139, p. 110226, May 2023, doi: 10.1016/j.asoc.2023.110226.

[9] 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.

[10] R. Mahendra, "Semi-supervised Textual Entailment on Indonesian Wikipedia Data Recommender Systems for e-Commerce View project Consumer Health Question Answering System for Indonesian Language View project," 2018, doi: 10.13140/RG.2.2.18820.27521.

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Association for Computational Linguistics, Oct. 2018. doi: 10.18653/v1/D15-1075.

[12] P. Ganesh *et al.*, "Compressing Large-Scale Transformer-Based Models: A Case Study on BERT," in *Transactions of the Association for Computational Linguistics, Volume 9*, 2020. doi: 10.1162/tacl_a_00413.

[13] F. Koto, A. Rahimi, J. H. Lau, and T. Baldwin, "IndoLEM and IndoBERT: A Benchmark Dataset and Pre-trained Language Model for Indonesian NLP," in *Proceedings of the 28th International Conference on Computational Linguistics*, Nov. 2020. doi: 10.18653/v1/2020.coling-main.66.

[14] R. A. Hidayat, I. N. Khasanah, W. C. Putri, and R. Mahendra, "Feature-Rich Classifiers for Recognizing Textual Entailment in Indonesian," in *Procedia CIRP*, Elsevier B.V., 2021, pp. 148–155. doi: 10.1016/j.procs.2021.05.094.

[15] T. S. Qaid, H. Mazaar, M. Y. H. Al-Shamri, M. S. Alqahtani, A. A. Raweh, and W. Alakwaa, "Hybrid Deep-Learning and Machine-Learning Models for Predicting COVID-19," *Comput Intell Neurosci*, vol. 2021, 2021, doi: 10.1155/2021/9996737.

[16] S. Fahmi, L. Purnamawati, G. F. Shidik, M. Muljono, and A. Z. Fanani, "Sentiment analysis of student review in learning management system based on sastrawi stemmer and SVM-PSO," in *Proceedings - 2020 International Seminar on Application for Technology of Information and Communication: IT Challenges for Sustainability, Scalability, and Security in the Age of Digital Disruption, iSemantic 2020*, Institute of Electrical and Electronics Engineers Inc., Sep. 2020, pp. 643–648. doi: 10.1109/iSemantic50169.2020.9234291.

[17] P. M. Prihatini, "Implementasi Ekstraksi Fitur Pada Pengolahan Dokumen Berbahasa Indonesia," *Matrix : Jurnal Manajemen Teknologi dan Informatika*, vol. 6, no. 3, 2017.

[18] M. J. Kusner, Y. Sun, N. I. Kolkin, and K. Q. Weinberger, "From Word Embeddings To Document Distances," *Proceedings of the 32nd International Conference on Machine Learning*, Jul. 2015, [Online]. Available: https://proceedings.mlr.press/v37/kusnerb15.html

[19] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet physics doklady*, vol. 10, no. 8, pp. 707–710, 1966.

[20] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, "A large annotated corpus for learning natural language inference," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2015, pp. 17–21. doi: 10.18653/v1/D15-1075.

[21] Arawinda Dinakaramani, Fam Rashel, Andry Luthfi, and Ruli Manurung, "Designing an Indonesian part of speech tagset and manually tagged Indonesian corpus," *2014 International Conference on Asian Language Processing (IALP)*, 2014, doi: 10.1109/IALP.2014.6973519.

[22] J. Reitan, J. Faret, B. Gambäck, and L. Bungum, "Negation Scope Detection for Twitter Sentiment Analysis," 2015. [Online]. Available: https://www.twitter.com

[23] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: a Method for Automatic Evaluation of Machine Translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, E. C. D. L. Pierre Isabelle, Ed., 2002, pp. 311–318.

[24] Lewis. Tunstall, L. von. Werra, and Thomas. Wolf, *Natural language processing with transformers : building language applications with Hugging Face*. O'Reilly Media, 2022.

[25] J. P. Vásconez, L. I. Barona López, Á. L. Valdivieso Caraguay, and M. E. Benalcázar, "A comparison of EMG-based hand gesture recognition systems based on supervised and reinforcement learning," *Eng Appl Artif Intell*, vol. 123, Aug. 2023, doi: 10.1016/j.engappai.2023.106327.

[26] M. Uzair and N. Jamil, "Effects of Hidden Layers on the Efficiency of Neural networks," in *Proceedings - 2020 23rd IEEE International Multi-Topic Conference, INMIC 2020*, Institute of Electrical and Electronics Engineers Inc., Nov. 2020. doi: 10.1109/INMIC50486.2020.9318195.

[27] E. Strubell, A. Ganesh, and A. McCallum, "Energy and Policy Considerations for Deep Learning in NLP," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. doi: 10.18653/v1/P19-1355.

[28] L. Lannelongue, J. Grealey, and M. Inouye, "Green Algorithms: Quantifying the Carbon Footprint of Computation," *Advanced Science*, vol. 8, no. 12, Jun. 2021, doi: 10.1002/advs.202100707.