Accredited Ranking SINTA 2
Decree of the Director General of Higher Education, Research and Technology, No. 158/E/KPT/2021
Validity period from Volume 5 Number 2 of 2021 to Volume 10 Number 1 of 2026

Published online on: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 7 No. 6 (2023) 1464 - 1475 ISSN Media Electronic: 2580-0760

Abstractive and Extractive Approaches for Summarizing Multi-document Travel Reviews

Narandha Arya Ranggianto[1], Diana Purwitasari[2]*, Chastine Fatichah[3], Rizka Wakhidatus Sholikah[4]
[1,2,3,4]Informatics Engineering Departement, Institut Teknologi Sepuluh Nopember, Indonesia
[1]narandharanggi@gmail.com, [2]diana@if.its.ac.id, [3]chastine@if.its.ac.id, [4]wakhidatus@its.ac.id

Abstract

Travel reviews offer insights into users' experiences at places they have visited, including hotels, restaurants, and tourist attractions. Reviews are a type of multi-document, where one place has several reviews from different users. Automatic summarization can help users get the main information in multi-document. Automatic summarization consists of abstractive and extractive approaches. The abstractive approach has the advantage of producing coherent and concise sentences, while the extractive approach has the advantage of producing an informative summary. However, there are weaknesses in the abstractive approach which results in inaccurate and less information. On the other hand, the extractive approach produces longer sentences compared to the abstractive approach. Based on the characteristics of both approaches, we combine abstractive and extractive methods to produce more concise and informative summary than can be achieved using either approach alone. To assess the effectiveness of abstractive and extractive, we use ROUGE based on lexical overlaps and BERTScore based on contextual embeddings which it be compared with a partial approach (abstractive only or extractive only). The experimental results demonstrate that the combination of abstractive and extractive approaches, namely BERT-EXT, leads to improved performance. The ROUGE-1 (unigram), ROUGE-2 (bigram), ROUGE-L (longest subsequence), and BERTScore values are 29.48%, 5.76%, 33.59%, and 54.38%, respectively. Combining abstractive and extractive are higher performance than partial approach.

Keywords: abstractive; extractive; summarization; bert; gpt2; clustering; sentiment; keyword

1. Introduction

Online review sites significantly affect the tourism industry, as travellers can review various travel products. Reviews can provide information regarding places and services before the user decides to travel, such as hotels, restaurants, and tourist attractions [1]. For example, the travel review website, TripAdvisor is the largest platform for sharing consumer reviews about their experience with accommodation. Reviews consist of multi-document making it difficult to read them. Moreover, users must deal with various topics and redundant information from a set of reviews [2]. It will take the reader time to get the main information.

The solution is to get the main information quickly using automatic summarization. In general, automatic summarization uses an abstractive approach that has the advantages of getting the main information because it produces better coherence and concise summary [3]. The deep learning method was used in an abstractive approach because the transformer model has an encoder and decoder architecture [4]. The encoder architecture helps in understanding important information while the decoder builds more concise sentences. However, the transformer model runs into problems when facing summarization in multi-document reviews.

In multi-document reviews consist of many reviews which causes the input token to get longer. This will be a weakness of the transformer model because it has limited input tokens so the tokens will be discarded if they exceed the maximum limit to reduce large computations [5]. The model requires large data to understand the variety of words because the review consists of informal data that spelling mistakes, informal expression, and grammatical mistakes. It leads to incorrect summary information and lack of information generated [6]. Previous studies used the BERT model which still produced errors in the summary results that many of the resulting sentences were incomplete, phrases were inaccurate, and there were repeated words [7]. Other study also stated that the extractive approach with the TextRank model produced superior performance with a ROUGE-1 value of 16.89% and ROUGE-L of 12.15% compared to the abstractive BART model of 15.70% for

Accepted: 13-06-2023 | Received in revised: 10-11-2023 | Published: 30-12-2023

ROUGE-1 and 10.88% for ROUGE-L [8]. On the other hand, there is an extractive approach that can provide more appropriate information because it does not require a complex process to build paraphrase sentences, such as abstractive approach [9]. It selects important sentences based on calculating features in the text. This makes it easier for an extractive approach to avoid lack information and redundant text in summary [10]. However, this approach has the drawback of generating a larger number of sentences. A previous study reported that the effectiveness of TextRank increased as the number of sentences in the document grew larger. This resulted in a reduction of only 50% in the number of sentences from the original document [11]. Based on the characteristics of both approaches, we combine abstractive to produce a concise summary and extractive to produce a more informative summary.

Previous studies only used abstractive approach with pre-trained models such as BERT and GPT2. In transformer, there are pre-trained models such as BERT is a bidirectional-encoder being used to perform summarization automatically. Even though its nature is an encoder, BERT can be used as a decoder because BERT has a Sentence Prediction training concept to generate text [7], [12]. GPT2 is an autoregressive-based model used for sentence construction commonly used as a decoder [13]. Meanwhile, an extractive approach can also be carried out using BERT which is called BERT Extractive, but it is still vulnerable to understanding context because the model is trained more on news data than review data [14]. Besides BERT, extractive approaches that pay attention to semantic values can be carried out based on feature combinations to get candidate summary from higher scores. Previous studies used a combination of features based on centroid and statistical approaches [15]. This combination has the advantages of semantic information compared to just using the statistical feature, which is called TextTeaser [16]. The sematic values based on centroid approach is carried out to obtain the value of relevance and novelty value. The statistical value based on position text is an important feature in a sentence. This combination of features provides good performance compared to using TF-IDF, LexRank, and TextRank [15]. However, it still has weaknesses in the segmentation and limited features for review text.

Segmentation was carried out only based on dot punctuation (.) to divide one sentence and others. Reviews have informal sentences and short text so that the segmentation results will produce unimportant topics. Reviews also consist of high sparse data, where sentences have no significance in a document. In addition, there are spam sentences that can make sentences biased and prone to incorrect judgments [17]. Usually in travel reviews there are topics that always appear in each review so these become important topics discussed [18]. The use of the clustering method will help group important topics, ensuring that sentences not related to the main topic are deleted [19]. Furthermore, combination features in previous study only used the centroid and position to focus on news text [15]. This reduces the information needed to get the words of the sentence because review text is more likely to contain opinions and objective evaluations from the users.

The feature commonly used in review texts is sentiment. Sentiment will provide information related to a positive or negative assessment of a text so that this assessment is considered an important value in summarization [20]. Previous study used sentiment based on VADER, where this method can process a sentence negation so as to provide a more specific sentiment value produced good performance compared to Textrank and Lexrank [21], [22]. However, the use of the sentiment feature is only able to consider sentences based on their positive or negative which will eliminate the important sentence. For example, in the sentence “what an extraordinary experience and moment that cannot be wasted”. The sentence has a positive sentiment value, but the context in the sentence becomes ambiguous because it does not have the element of the intended object.

Retrieval of object words in a sentence can be done using keyword extraction. Previous study used TF-IDF to extract keywords in the form of nouns in sentences where the word will show the importance value of a sentence. If the noun appears in a sentence and is unique, it produces a high value compared to other sentences [10]. In addition to TF-IDF, there is a YAKE keyword extraction method that has better performance. YAKE has the advantage of extracting keywords in its own documents without the need for external dictionaries such as IDF calculations and other corpus such as NER, POS Tagger, or stopwords [23]. Therefore, we not only use centroid and position features but also sentiment and keyword features.

Based on the previous explanation, we model the text summarization by combining two approaches, namely abstractive and extractive. This combination allows us to create both concise and informative summary, whereas previous studies used only one approach. Apart from that, we use clustering to overcome the segmentation problem that produces unimportant sentences in extractive approach. We also add sentiment and keyword features to produce a new combination of features, namely centroid, position, sentiment, and keyword. The sentiment value is obtained from the VADER model, while the keyword value uses the YAKE model. This helps provide additional information in the sentence, which enhances the quality of the summarization evaluation.

DOI: https://doi.org/10.29207/resti.v7i6.5170

2. Research Methods

The proposed method can be seen in Figure 1, which consists of 4 main stages: data collection and pre-processing, abstractive, extractive, and combining abstractive-extractive. Firstly, we collect data on travel reviews from TripAdvisor. Then, the annotators create summary references or ground truth for each review object. The review will continue in the pre-processing stage for review standardization. The abstractive stage involves fine-tuning pre-trained models to generate candidate summaries based on paraphrasing. The extractive stage is used to provide additional main information. It consists of clustering and feature combinations, including relevance score, novelty score, position score, and sentiment keyword score. The sentences from the abstractive and extractive stages will be merged to create the final summary.

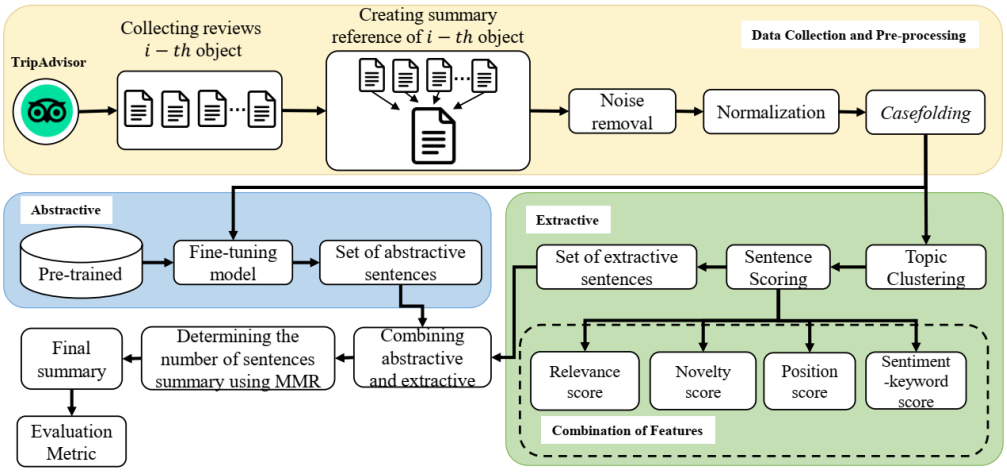

Figure 1. Proposed Method of Abstractive and Extractive Approaches

2.1 Data Collection and Pre-processing

In this study, we crawl reviews from the TripAdvisor website. Each object will consist of more than one review. From several reviews for the same object, a summary reference will be created by the annotator, which will be used for training and evaluation. There are three annotators with linguistic skills: graduates in Indonesian Literature or Indonesian Language Education Teaching.

After obtaining summary references for all objects from the annotators, the reviews are processed to eliminate informal words. Noise removal is conducted to eliminate unimportant characters. Other deletions are based on URLs, multiple punctuation marks or spaces, and non-ASCII characters. Some punctuation marks are removed except for periods and commas, as we use them for sentence tokenization. Then, the sentences are normalized by converting informal sentences based on a formal dictionary. If informal words appear in the review, they are replaced with formal words from the dictionary. Words that have more than 2 repeating characters at the end, for example, "veryyyy good," is changed to "very good. Then, we convert words to all lowercase letters in a process called case folding. This ensures that words do not have different meanings based on their capitalization. After preprocessing each document, we merge them into a single document for each object. This results in one document containing several reviews, which is used to calculate relevance, novelty, position, and sentiment-keyword scores.

2.2 Extractive Approach

The extractive approach involves a clustering process and sentence scoring. During the clustering process, sentences are grouped based on topics. Unimportant topics will be deleted to reduce noise sentences. Next, important sentences are assessed for their significance through a combination of features, namely relevance, novelty, position, and sentiment-keyword. The final combined value is utilized to select important sentences, with the highest values indicating their significance.

2.2.1 Topic Clustering

The obtained objects can be restaurants, hotels, or tourist attractions which have several reviews. Each object consists of several reviews which are combined to become one main review which will be processed in the summarization model which is denotated as D_i, where i is number of objects. For example, object i=1 is named "Novotel" in the hotel category. After the pre-processing stage, each document gets segmentation process, where the document will be separated based on dot punctuation (.) to become sentences. It is denotated $D_i = \{s_{t,i}\}$, where t is the index of sentence in the document D_i.

The results of segmentation in document D_i provide important topics or unimportant topics. An example of

DOI: https://doi.org/10.29207/resti.v7i6.5170

a sentence related to an important topic is "the facilities in the room are good". Meanwhile, an example of an unimportant sentence is "The last time I stayed here was May 12th". Therefore, we use clustering in each object D_i to keep important topics and delete unimportant topics.

We eliminate unimportant topics that consist of fewer than 3 member sentences in each cluster. We have document with important topics denoted as $D'_i = \{s_{n,i}\}$, where D'_i consists of sentences that have an important topic and n is the new index of a sentence from document D_i. In the clustering process, we transform the sentence vectors using BERT embeddings. BERT embeddings have a dimension of 768.

We use PCA to reduce vector dimensions to 2 vectors. Our empirical experiments show that using the original 768 dimensions leads to more clusters, which can be merged. Determination of the number of clusters is based on the optimal results obtained from the silhouette and elbow methods. The initial number of clusters is set within the range of 2 to 10. We choose to initialize a maximum of 10 clusters because the elbow point cannot be reliably detected with fewer than 10 clusters. Once we have identified the best number of clusters, the next step is to choose the most appropriate clustering method for our specific field. We modify the method selection process to make sure it works well for a wider range of applications. Furthermore, the selection of the clustering method uses unsupervised evaluation, namely the Silhouette Coefficient (SC), Calinski-Harabasz Index (CHI), and Davies Bouldin Index (DBI). The clustering methods being compared include k-means, k-medoids, and agglomerative hierarchical. The entire clustering process is illustrated using pseudocode in Table 1.

Table 1. Pseudocode of Topic Clustering

Algorithm 1: Pseudocode of topic clustering

```
Input: List of sentences in each D_i
Output: List of sentence important topic
    sents: List of sentences in each D_i
    # List to store elbow and silhouette score
    elbowScore ← empty list
    silhouetteScore ← empty list
    # List of sentences for important topic
    newDocument ← empty list
    for i ← 1 in sents do
    # Embedding sentence process
    sentVector[i] ← bertEmbedding (sent[i])
    end for
    # Reducing 2 dimensions using PCA
    for i ← 1 in sents do
        sentVector[i] ← pca(sentVector[i], 2)
    end for
    # Store elbow and silhouette score
    for i ← 2 to 10 do
        clusterResult ← clusterMethod(sentVector, i)
        elbow ← elbowMethod(clusterResult)
        silhouette ← silhouetteMethod(clusterResult)
    end for
    # Get the optimal number of clusters
    k_optimal1 ← clusterOptimal(elbowScore)
    k_optimal2 ← clusterOptimal (silhouetteScore)
    # Clustering process using optimal cluster number
    if k_optimal1 > k_optimal2 do
        topicCluster ← clusterMethod(sentVector,
        k_optimal1)
    else
        topicCluster ← clusterMethod(sentVector,
        k_optimal2)
    end if
    # Store topics and sentences in an array
    for topic in clusters do
        if count(clusters[topic]) >= 3 do
            for sent in sents:
                if sent in topic:
                    newDocument.add(sent)
                end if
            end for
        end if
    end for
    return newDocument
```

2.2.2 Relevance Score

Based on the document $D'_i = \{s_{n,i}\}$, we calculate each sentence $s_{n,i}$ to get feature combination such as relevance score, novelty score, position score, and sentiment-keyword score. For the first score is relevance feature which is carried out by finding similarity between sentence and centroid's document D'_i. We employ BERT sentence embeddings with 768 dimensions to calculate both scores, denoted as $\overrightarrow{s_{n,i}}$. The relevance score is derived from the proximity between the sentence vector and the centroid vector, which has been computed using Formula 1. Highest score in relevance showing the sentence closer to the centroid.

$$relevance(s_{n,i}) = cosine\ similarity(\overrightarrow{s_{n,i}}, D'_{i\_centroid} \quad (1)$$

2.2.3 Novelty Score

The centroid is also used to calculate novelty score which serves to acquire new information and minimize redundancy in the summary. The novelty score is shown in Formula 2. Its value is derived from the relevance score of a sentence compared to other sentences in D'_i. The sentence $\overrightarrow{s_{n,i}}$ will be calculated for similarity with $\overrightarrow{s_{l,i}}$ using cosine similarity, where $1 \leq l \leq |D'_i|,\ l \neq n$. Afterward, we compare the highest obtained similarity, $sim(\overrightarrow{s_{n,i}}, \overrightarrow{s_{l,i}})$, with a threshold. There are a few conditions for calculating the novelty score. The first condition is that if the highest similarity of $sim(\overrightarrow{s_{n,i}}, \overrightarrow{s_{l,i}})$ is lower than threshold, the sentence $s_{n,i}$ is given a high novelty score of 1. The second condition states that if the maximum similarity of $sim(\overrightarrow{s_{n,i}}, \overrightarrow{s_{l,i}})$ is greater than threshold, the sentence will be proceeded to the calculation of the relevance score. The sentence $s_{n,i}$ will be compared with index of sentence from highest value of $sim(\overrightarrow{s_{n,i}}, \overrightarrow{s_{l,i}})$ which is denoted as $s_{r,i}$. If the

DOI: https://doi.org/10.29207/resti.v7i6.5170

relevance score of $s_{n,i}$ is greater than $s_{r,i}$, then $s_{r,i}$ has high novelty value with value of 1. The third condition, if it does not meet previous condition then the novelty score is 1-max$\left(sim\left(\overrightarrow{s_{n,i}}, \overrightarrow{s_{l,i}}\right)\right)$. Novelty score is calculated in Formula 2 2, where r is argmax that means to get the index of sentence with highest value of $sim\left(\overrightarrow{s_{n,i}}, \overrightarrow{s_{l,i}}\right)$ and τ_n is novelty threshold. The threshold value used is 0.95.

$$novelty(s_{n,i}) = \begin{cases} 1, max(sim(\overrightarrow{s_{n,i}}, \overrightarrow{s_{l,i}})) < \tau_n, \\ 1 \le l \le |D'_i|, l \ne n \\ \\ 1, max(sim(\overrightarrow{s_{n,i}}, \overrightarrow{s_{l,i}})) > \tau_n \text{ and} \\ relevance\ (s_{n,i}) > relevance\ (s_{r,i}), r = \\ argmax(sim(\overrightarrow{s_{n,i}}, \overrightarrow{s_{l,i}})) \\ \\ 1\text{-max}\left(\ sim\left(\overrightarrow{s_{n,i}}, \overrightarrow{s_{l,i}}\right)\right), otherwise \end{cases} \quad (2)$$

2.2.4 Position Score

The sentence position value of a document has the concept that the first position is the most important sentence. The value will decrease if the sentence is far from the first position. In this section, we use information position of sentence from D'_i. The Formula 3 is used to get position score in this study, where $p(s_{n,i}, D_i)$ is position of sentence $s_{n,i}$ in D_i, D_i is the initial document before the clustering process, and $|D_i|$ is the number of sentences in the initial document. The range of position values is 0.5 to 1.

$$position(s_{n,i}) = \max(0.5, \exp\left(\frac{-p(s_{n,i}, D_i)}{\sqrt[3]{|D_i|}}\right)) \quad (3)$$

2.2.5 Sentiment-keyword Score

We will calculate $sentkey(s_{n,i})$ to determine the sentiment-keyword value using Formula 4. The sentiment weight is derived from the VADER method, with a value range of -1 to 1. However, in this study, we only uses positive values so as not to produce smaller value for negative sentiment sentence, so an absolute process is carried out. The VADER weight is divided by the number of words in the sentence to prioritize assigning a higher score to short sentences than to long ones. The sentiment calculation depends on the number of words denoted by $|s_{n,i}|$.

The value of keyword is calculated based on extracting keyword using YAKE of sentences in the document D'_i. YAKE gives weight of keywords with a range of 0 to 1. The YAKE model extracts several keywords in each sentence $s_{n,i}$, denoted as keywords $k_{c,i}$ and corresponding keyword weights $v_{c,i}$. Here, c represents the index of the keywords and i represents the index of the review object. Keyword $k_{c,i}$ is a collection of N-gram keywords for document D_i. Each sentence $s_{n,i}$ is examined using N-grams, where the N-grams in question are sets of words with lengths N=1, N=2, and N=3, denoted as N-grams($s_{n,i}$). If N-grams($s_{n,i}$) matches the N-gram keyword $k_{c,i}$, then that sentence is assigned a weight of $v_{c,i}$. Formula 4 is the calculation of the sentiment-keyword value which is the sum of the weights between sentiment and keywords.

$$sentkey(s_{n,i}) = \left(\frac{\text{abs}\left(VADER(s_{n,i})\right)}{|s_t|}\right) + \sum_{V(s_{n,i})=\{v_{c,i}|(k_{c,i},v_{c,i})\in YAKE(D_i)\}} V(s_{n,i}) \quad (4)$$

2.2.6 Final Score

Once the relevance score, novelty score, position score, and sentiment-keyword score are obtained, their total value will be calculated through a linear combination of these four values. Each value will be multiplied by its corresponding constant used as the weight: α, β, γ, and δ. It's important to note that $\alpha + \beta + \gamma + \delta = 1$. Weighted α for relevance features, β for novelty feature, γ for position feature, and δ for sentiment-keyword feature. Mathematically, the calculation of the total sentence value is represented by Formula 5. The sentences selected for extraction, referred to as $SE_i = \{s_{e,i}\}$, are chosen based on the values of $finalScore(s_{n,i})$. The sentence is ordered by higher final score. We select the sentences with word limitation, where total of extractive sentence is lower than average summary reference in the document. Extractive sentences will be continued in the process of merging sentences to abstractive approach. The process of extractive approach is shown using the pseudocode in Table 2.

$$finalScore(s_{n,i}) = \alpha \times relevance(s_{n,i}) + \beta \times novelty(s_{n,i}) + \gamma \times position(s_{n,i}) + \delta \times sentkey(s_{n,i}) \quad (5)$$

Table 2. Pseudocode of Extractive Approach

Algorithm 2: Selecting sentence using extractive approach
Input: document D_i, document D'_i
Output: List of extractive sentences
sents: List of sentences in each document D'_i
List to store sentence scores
finalScore ← empty list
List to store sentences of all topics
tmpSent ← empty list
List to store extractive sentences
extractiveSent ← empty list
Sentence embedding process

DOI: https://doi.org/10.29207/resti.v7i6.5170

Algorithm 2: Selecting sentence using extractive approach

```
sentVector ← embeddingSentence(sents)
# Calculating centroid
centroid ← mean(sentVector)
# Calculating relevance score
for i ← 1 to count(sents) do
    relevance[i] ← relevanceScore(sentVector[i], centroid)
end for
# Calculating novelty score
for i ← 1 to count(sents) do
    novelty[i] ← noveltyScore(sentVector[i],
    relevance, τ_n)
end for
for i ← 1 to count(sents) do
    # Get position information in each document
    position[i] ← positionScore(sent[i], |D_i|)
    # Number of sentences in sent[i]
    countWord[i] ← count(sent[i])
    # Calculating sentiment score
    sentiment[i] ← sentimentScore(sent[i], countWord)
    # List of n-gram keywords
    listKeyword ← YAKE(S_{n,i})[0]
    # List of keyword weight for n-gram
    weightKeyword ← YAKE(S_{n,i})[1]
    for keyword in listKeyword do
        if keyword in sent[i] do
            keywordScore[i] ← keywordScore[i] +
            weightKeyword[keyword]
        end if
    end for
    sentkey[i] ← sentiment[i] + keywordScore[i]
    # Calculating final score
        relevance[i] ← relevanceScore(sentVector[i],
        centroid)
    end for
    # Calculate novelty score
    finalScore[i] ← α * relevance[i] + β * novelty[i] + γ *
    position[i] + δ * sentkey[i]
end for
# Selecting sentence of extractive approach
extractiveSent ← selectingSentence(sents, finalScore,
sentenceLimit)
return extractiveSent
```

2.3 Abstractive Approach

We use 2 pretrained models namely BERT and GPT2. The fine-tuning process is carried out using the collected dataset. In the BERT and GPT2 models, special tokens [SEP] and [CLS] will be added. The [SEP] token is used to indicate the beginning of a sentence, while [CLS] is the end of a sentence. After tokenization based on each model, we build the encoder-decoder architecture.

In encoder-decoder architecture, standard GPT2 can only work on the decoder model, so it will combine with BERT as the encoder. While standard BERT can be used as encoder or decoder.

After the fine-tuning process has been carried out, the model will generate a sentence of summary with beam search to get better and varied sentences. The abstractive sentences generated by either BERT or GPT2. After tokenization based on each model, we construct an encoder-decoder architecture. In this architecture, the standard GPT-2 is employed exclusively for the decoder model and is coupled with BERT as the encoder. Meanwhile, the standard BERT can be used as either the encoder or the decoder. Following the completion of the fine-tuning process, the model will utilize beam search to generate summary sentences that are improved and diverse. The abstractive sentences generated by either BERT or GPT-2 are referred to as $SA_i = \{s_{a,i}\}$.

2.4 Combining Abstractive and Extractive

The result of abstractive and extractive approaches will be segmented each sentence to get final summary. To maintain sentence coherence, abstractive results are prioritized over extractive. Therefore, sentences in extractive will be calculated dissimilarity to sentences in abstractive, so the resulting dissimilarity value is as much as the number of list sentences in the abstractive. To get one value for the extractive sentence list, we calculate the average of dissimilarity values. The value of dissimilarity is shown in Formula 6. $SE_i = \{s_{e,i}\}$ is a set of extractive sentences, where e is the index of extractive sentence and i is the index of review object. $SA_i = \{s_{a,i}\}$ is a set of abstractive sentences, where a is the index of abstractive sentence and i is the index of the review object. We calculate the dissimilarity score between each sentence of $s_{e,i}$ and all sentences in SA_i. After all sentences in SE_i have been assigned an average dissimilarity score using Formula 5, any sentence in SE_i with a score less than the threshold τ_d will be removed, where τ_d is dissimilarity threshold. After the deletion process, the set SE_i becomes a new set, denoted as SE'_i. SA_i will be combined with SE'_i denotated as $S_i = SA_i \cup SE'_i$, where S_i is set of combined extractive and abstractive sentences.

$$dissimilarity(s_{e,i}) = avg_{s_{a,i} \in SA_i}(1 - sim(s_{e,i}, s_{a,i})) \quad (6)$$

2.5 Determining Number of Sentence Summary

The next process is to perform sentence selection using the concept of reducing redundant sentences through Maximal Marginal Relevance (MMR).

The MMR process is used to maintain the summary length, ensuring that the combination of abstractive and extractive remains concise based on diverse sentence in selection of summary candidates. In Formula 7, MMR is obtained from the subtraction of $\lambda sim_1(s_n, Q)$ and $(1-\lambda) max_{s_m \in S} sim_2(s_n, s_m)$. Notation of λ is user-tunable diversity, s_n is sentences that have not been extracted, Sum is sentences that have been extracted into summary. MMR will store sentences that have a high difference value compared to other sentences in one document S_i.

The MMR continues to run when the desired number of summary sentences has not been reached. The value of λ is initialized at 1.0 and continues to decrease by

DOI: https://doi.org/10.29207/resti.v7i6.5170

0.1 for each iteration as many as the desired number of summary sentences. The process of combining abstractive and extractive approaches are shown using the pseudocode in Table 3.

Table 3. Pseudocode of Combining Abstractive and Extractive

Algorithm 3: Combining Abstractive and extractive

Input: List of extractive sentences SE_i, list of abstractive sentences SA_i
Output: List of finalSummary

```
    sentAbs: List of abstractive sentences
    sentExt: List of extractive sentences
    # List to store abstractive-extractive sentence
    finalSummary ← empty list
    # Sentence embedding process
    sentVector_1 ← embeddingSentence(sentAbs)
    sentVector_2 ← embeddingSentence(sentExt)
    # Array to store sentence extractive with higher
    dissimilarity score
    filteredSentence ← empty list
    for i ← 1 to count(sentExt) do
        # List to store dissimilarity score of sentExt
        for j ← 1 to count(sentAbs) do
            dissimilarityScore[i] ←
            dissimilarity(sentVector_2[i], sentVector_1[j])
        end for
        # Sort filtered extractive sentence by higher
        dissimilarity score
        meanDissimilarity ← mean(dissimilarityScore)
        if meanDissimilarity > τ_d do
            filteredSentence.add(sentExt[i])
        end if
    end for
    # Merge sentences of abstractive and extractive
    for sent in filteredSentence do
        if length(sentAbs) < sentLimit do
            sentAbs.add(sent)
        end if
    end for
    # Sentence embedding process
    summVector ← embeddingSentence(sentAbs)
    # Calculate centroid for ordering sentence using
    relevance score
    centroid ← mean(summaryVector)
    for i ← 1 to count() do
        relevance[i] ← relevanceScore(summVector[i],
        centroid)
    end for
    # Select and sort final summary by relevance score
    finalSummary ← MMR(mergedSentence)
    return finalSummary
End function
```

$$MMR = argmax_{s_n \in S_i}[\lambda \times sim_1(s_n, S_i) - (1-\lambda) \times max_{s_m \in Sum}\, sim_2(s_n, s_m)] \quad (7)$$

2.5 Evaluation Metric

For evaluation extractive and abstractive, we use the Recall-Oriented Understudy for Gisting Evaluation (ROUGE), particularly ROUGE-N (ROUGE-1 and ROUGE-2) and ROUGE-L. ROUGE-N is calculated from the occurrence of the word n-gram from the summary prediction and summary reference. In Formula 8, RS are the reference summaries, j is the n-gram size, and $Count_{match}(gram_n)$ is the maximum number of n-grams co-occuring in a candidate summary and set of reference summaries. While ROUGE-L is based on the Longest Common Subsequence overlapping between prediction and reference. The ROUGE scores have a range of values from 0 to 1. However, these values will be converted into percentages to facilitate interpretation, making them range from 0% to 100%. Besides based on the calculation of word occurrences, we add one evaluation value using semantic value proximity using BERTScore that is commonly used for text generation. BERTScore can be used coherence between sentences. The method calculates summary prediction with summary reference based on cosine similarity using BERT sentence embedding.

$$ROUGE - N = \frac{\sum_{S \in RS} \sum_{gram_j \in S} Count_{match}(gram_j)}{\sum_{S \in RS} \sum_{gram_j \in S} Count(gram_j)} \quad (8)$$

3. Results and Discussions

In this section, we present a comparative of the results obtained by the proposed method of extractive (clustering and features combination) as well as abstractive and extractive combination. Therefore, we conducted several experiments, namely 1) investigating the optimal epoch for constructing pre-trained model sentences; 2) investigating the optimal clustering method for topic deletion in the review; 3) investigate the threshold value for abstractive-extractive combinations; 4) assessing the performance of the proposed method of extractive and abstractive-extractive with state-of-the-art.

3.1 Datasets

In this study, we use beautifulsoup4 library based on python for crawling data from TripAdvisor website.

Table 4. Data Statistics

	Training	Validation	Testing
# of documents	240	30	30
Avg # of sents/obj	6.51	6.05	6.49
Avg # of word/obj	89.06	85.27	92.69
Avg # of sents/summ	5.0	4.70	4.83
Avg # of word/summ	55.39	57.13	56.43

Dataset contains three categories: hotel, restaurant, and attraction. We collected reviews from each object based on those categories. Each object has 5 reviews taken from January 2018 to August 2022. Total object that is collected of 300. After the data obtained, an annotator created summary reference for those objects, so one object has one summary. For preprocessing, we used NLTK to tokenization for removing all noise sentences such as URLs, special characters, etc. After preprocessing stage, we split the dataset into 80% training, 10% validation, and 10% testing. Training and validation datasets are used for fine tuning the pre-trained model of BERT and GPT2. While dataset testing is used to evaluate the combination of abstractive and extractive. The complete statistics of the data are shown in Table 4.

DOI: https://doi.org/10.29207/resti.v7i6.5170

We use two additional datasets for evaluation. The public DUC2004 Task2 is created by NIST for multi-news dataset. DUC2004 Task2 consists of 50 clusters, where each set consists of 10. For each cluster, the annotator written four summaries. The second dataset is multi-document review from Amazon, which consists of 60 products and 8 reviews. Three reference summaries were written by an annotator for each product.

3.2 Experimental Setup for Abstractive Approach

The first process in abstractive approach was did fine-tune model for BERT and GPT2. We use the Huggingface library to build the encoder-decoder architecture commonly used for summarization. The model BERT has 12 encoder layers, 768 hidden sizes, and 110M parameters. The BERT model was obtained from the 'indobenchmark/indobert-base-p1' model that has been trained with Indo4B dataset. The second model, namely GPT2, uses the GPT2 model from 'cahya/gpt2-small-indonesian-522M' trained with Indonesian Wikipedia. GPT2 has 12 decoder layers, 768 hidden sizes, and 122M parameters.

We did the fine-tuning process for both models. The splitting data is based on Table 5. All models use an input sequence length of 512 for the encoder, which is the basic limit of both models. Then, the length of input sequence for decoder uses the average word length of review which is 80. If the document is less than the length of the sequence, the [PAD] token will be added so that the number of input tokens for encoder and decoder is the same.

Table 5. Evaluation of Epoch Number

Method	Epoch	R1	R2	RL
BERT	64	**28.23**	**4.12**	**32.49**
	128	26.47	3.22	30.26
GPT2	64	24.28	3.04	28.71
	128	**24.87**	**3.07**	**29.14**

In this section, the experiments are conducted to find optimal epoch for both models. We use epoch numbers 64 and 128. Other parameters used for fine-tuning are batch_size = 16 and learning_rate = 1e-4. All evaluations use F1 of ROUGE-1 (R1), ROUGE-2 (R2), and ROUGE-L (RL). The results of epoch comparison for both models can be seen in Table 5. BERT model shows that the optimal results are at epoch 64, while GPT2 is at epoch 128. The high epoch in BERT model experiences overfitting resulting in a lower rouge value. This is because the BERT model has been fine-tuned from various tasks such as text classification or sequence labeling which makes it at risk of overfitting.

3.3 Experimental Setup for Extractive Approach

Firstly, the experiments are conducted to process is clustering stage that uses sentence embedding from BERT from 'indobenchmark/indobert-base-p1' that has a hidden size of 768. We clustered the sentences for each object to remove unimportant topics. The first experiment carried out aims to get the best clustering method. We used several method comparisons to get the best clustering method, namely k-medoids, k-means, and agglomerative. The evaluation of clustering using unsupervised measurements: Silhouette Coefficient (SC), Davies Bouldin Index (DBI), and Calinski-Harabasz Index (CHI).

Table 6 shows comparison of three methods. From the table, k-means and agglomerative produced the highest scores compared to k-medoids. SC score measures the confidence level in grouping a cluster by looking at the distance of objects in one cluster and objects in the nearest neighboring clusters. CHI score is the ratio between the sum of square values between clusters (SSB) and the sum of square within-cluster (SSW) values multiplied by the normalization factor. The k-means gets the highest SC and CHI scores, respectively, 0.52 and 35.57. While DBI is average similarity of each cluster with cluster most similar to it with range 0-1. A smaller value indicates that each cluster is data that is different from other clusters. The lowest DBI value is obtained by agglomerative, which is 0.58. Based on the number of evaluations obtained, k-means got two better scores, namely SC and CHI, compared to agglomerative and k-medoids.

Table 6. Evaluation of Clustering Methods

Method	SC	DBI	CHI
k-medoids	0.50	0.67	32.36
k-means	**0.52**	0.61	**35.57**
agglomerative	0.51	**0.58**	33.59

Then, we computed combination of four features: relevance, novelty, position, and sentiment-keyword in Formula 5 to get final score, so each feature is weighted of $\alpha, \beta, \gamma,$ and δ. We use the concept of weight α, β has a higher value than γ. At α and β use a weight valued at 0 to 1 with a step constant of 0.1. While γ, δ is 0.05 to 0.2 with a constant step of 0.5. From testing data, we performed hyperparameters with total combination is 33. We obtained values of the hyperparameters are 0.5, 0.3, 0.05, and 0.15 for $\alpha, \beta, \gamma, and\ \delta$, respectively. Based on Table 4 which shows that the average document summary is 5 sentences, we use that number to take 5 sentences of the extractive approach from the higher final score.

3.4 Experimental Setup for Combining Abstractive and Extractive

The sentence results from the abstractive approach using either the BERT or GPT2 models will be combined with the sentence results from the extractive approach. Sentence merging is done by calculating the dissimilarity of abstractive and extractive sentences to get concise and informative summary.

DOI: https://doi.org/10.29207/resti.v7i6.5170

We conducted an experiment to determine the optimal threshold for combining abstractive and extractive sentences. For each model, the optimal dissimilarity value was calculated with range 0.1 to 0.5. Figure 2 shows that each model has its own dissimilarity value. BERT model got optimal results at 0.40 with value ROUGE-1 (R1) of 29.37% and GPT2 model got optimal results at 0.35 with value of ROUGE-1 (R1) 25.50% and ROUGE-L (RL) of 30.05%. Model that has a greater rouge value, the threshold value will be higher. This is because the sentence will range towards redundancy when the model can build sentences better.

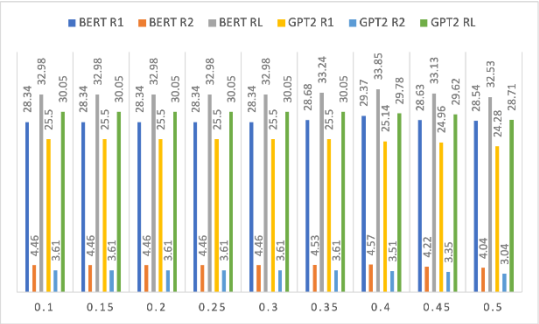

Figure 2. Threshold Dissimilarity

3.5 Comparative Evaluation

To evaluate the performance of extractive approach, we compare our model of clustering and 4 features with baseline using 3 features: relevance, novelty, and position. BERT model used for English is 'bert-base-uncased'. We used the dataset by baseline method, namely DUC2004. As an additional evaluation we used Amazon data that has the same text characteristics as TripAdvisor about product reviews. Apart from the baseline method, we did a comparison with unsupervised methods such as TextRank, BERT Extractive, and TextTeaser.

In this section we use the BERTScore evaluation matrix to assess sentence prediction and sentence reference based on their semantic proximity. Table 6 shows that our model is still superior for all datasets. It is surprising that our model is also superior for news data. The sentiment-keyword feature can focus on important topics that arise from one's opinion because they are unique keyword and explain additional information about the news content. The values of ROUGE-1 (R1), ROUGE-2 (R2), ROUGE-L (RL), and BERTScore (BS) are 27.04%, 5.52%, 31.75%, and 54.06%.

Then comparisons were made based on other unsupervised methods. Table 7 shows performance comparison of our model with the TextRank, BERT Extractive, and TextTeaser. Our model produces better performance compared to the three methods. The baseline of 3 features has a lower ROUGE-L value than TextTeaser, while our model has a superior performance for the overall evaluation. Furthermore, the lowest value is obtained by BERT Extractive.

The combination of abstractive and extractive compared its performance with extractive methods from baseline to other unsupervised methods. In addition, baseline abstractive approaches such as BERT and GPT2 were also compared to find out whether abstractive and extractive combinations outperformed the abstractive approach. Table 7 shows that combining abstractive and extractive is better than using only extractive or abstractive. The best combining abstractive and extractive using the BERT-EXT model with ROUGE-1 (R1) of 29.48%, ROUGE-2 (R2) of 5.76%, ROUGE-L (RL) of 33.59%, and BERTScore (BS) of 54.38%. Meanwhile, the combination in the GPT2-EXT model provides increased performance of the baseline of GPT2 but is still unable to outperform the extractive approach.

3.6 Discussions

This research combines abstractive and extractive approaches. The extractive approach is based on combining clustering, relevance feature, novelty feature, position feature, and sentiment-keyword feature.

The proposed method of extractive approach provides advantages over the baseline, which can be seen from the comparison of data from DUC2004 and Amazon in Table 7.

Table 7. Evaluation Result for Partial and Combining Approach

Method	R1 (%)	R2 (%)	RL (%)	BS (%)	Data
Extractive					
3 features (baseline)	28.85	6.07	29.43	66.47	DUC2004
Clustering + 4 features (proposed method)	**29.10**	**6.77**	**29.98**	**66.62**	DUC2004
3 features (baseline)	28.28	5.40	32.06	63.18	Amazon
Clustering + 4 features (proposed method)	**29.15**	**5.41**	**32.49**	**63.29**	Amazon
TextRank	24,15	3,40	27,61	48.27	TripAdvisor
BERT Extractive	24,17	4,61	29,14	50.34	TripAdvisor
TextTeaser	26.57	5.67	30.68	50.85	TripAdvisor
3 features (baseline)	25.99	5.84	30.56	51.85	TripAdvisor
Clustering + 4 features (proposed method)	**27.04**	**5.52**	**31.75**	**54.06**	TripAdvisor

DOI: https://doi.org/10.29207/resti.v7i6.5170

Method	R1 (%)	R2 (%)	RL (%)	BS (%)	Data
Abstractive					
BERT (baseline)	28.23	4.12	32.49	52.73	TripAdvisor
GPT2 (baseline)	24.87	3.07	29.14	50.20	TripAdvisor
Abstractive-extractive					
BERT-EXT (proposed method)	**29.48**	**5.76**	**33.59**	**54.38**	TripAdvisor
GPT2-EXT (proposed method)	27.29	4.57	31.72	52.75	TripAdvisor

Table 8. Example Summary of Abstractive and Extractive Approaches

Extractive (EXT) (125 words)

ada bagian yang **menjelaskan mengenai agama agama yang terdapat di indonesia** [38 kata tidak ditampilkan]. **dilengkapi toilet serta kios kios yang menjual makanan** [21 kata tidak ditampilkan] **alat alat peraga**. selain itu ada sejumlah taman museum **beragam tempat ibadah** teater dan wahana rekreasi menyenangkan lainnya. bukan cuma museum mengenai daerah tetapi juga **terdapat museum transportasi yang menjelaskan mengenai semua moda tranportasi yang ada di indonesia** ini, musem air tawar, iptek dan banyak lagi lainnya. **(Indonesian Language)**

there is a section that **explains the religions in Indonesia**, [38 words are abbreviated from here]. **equipped with toilets and stalls selling food** [21 words are abbreviated from here] **contain props**. besides that, there are several parks, museums, **various places of worship,** theaters, and other fun attractions. not only museums about the area but **there is also a transportation museum which explains all modes of transportation in Indonesia,** freshwater, **science and technology,** and many others. **(English Language)**

BERT (39 words)

taman ini memberikan pengalaman yang menyenangkan untuk belajar dan bermain. **anak - anak dapat mencoba alat - alat simulasi iptek.** kegiatan lainnya yaitu menggambar dan melukis. **tiketnya masih termasuk murah**. kekurangannya hanya parkiran yang sempit dan tidak ada lift. **(Indonesian Language)**

this park provides a fun experience to learn and play. **children can try science and technology simulation tools.** other activities are drawing and painting. **tickets are cheap.** the only drawback is the narrow parking lot and no elevator. **(English Language)**

BERT-EXT (119 words)

bukan cuma museum mengenai daerah tetapi juga **terdapat museum transportasi yang menjelaskan mengenai semua moda tranportasi yang ada di indonesia ini musem air tawar iptek dan banyak lagi lainnya.** kegiatan lainnya yaitu menggambar dan melukis. **anak - anak dapat mencoba alat - alat simulasi iptek. tiketnya masih termasuk murah.** kekurangannya hanya parkiran yang sempit dan tidak ada lift. ada bagian yang **menjelaskan mengenai agama agama yang terdapat di indonesia ini lengkap dengan ruang ibadahnya** [33 kata tidak ditampilkan] **(Indonesian Language)**

not only museums about the area but **there is also a transportation museum which explains all modes of transportation in indonesia,** freshwater, **science and technology,** and many others. other activities are drawing and painting. **children can try science and technology simulation tools. tickets are cheap.** the only drawback is the narrow parking lot and no elevator. **there is a section that explains about the religions in indonesia,** [33 words are abbreviated from here] **(English Language)**

Reference summary (71 words)

taman ini menyuguhkan semua informasi yang berkaitan dengan indonesia berbentuk museum. **terdapat penjelasan tentang agama serta budaya yang ada di indonesia**. **tempat ini juga menyediakan beberapa museum lain seperti museum transportasi dan iptek. museum iptek memiliki banyak perlatan sains yang edukatif. anak-anak bisa mencoba alat-alat tersebut.** fasilitas yang menarik lainnya yaitu taman, teater, dan wahana rekreasi. **harga tiket juga sangat terjangkau.** tempat ini juga menyediakan makanan yang dijual di kios-kios sekitar. **(Indonesian Language)**

this park presents all information related to indonesia in the form of a museum. **there is an explanation of religion and culture in indonesia. this place also provides several other museums such as the transportation and science and technology museum. the science and technology museum has a lot of educative science equipment. children can try these tools.** other interesting facilities are parks, theaters, and recreational rides. **ticket prices are also very affordable.** this place also **provides food which is sold in the stalls around. (English Language)**

In addition, the proposed method has been compared with other unsupervised methods such as TextRank, TextTeaser, and BERT Extractive which shows that ROUGE and BERTScore values are still superior. While the abstractive approach used pre-trained models, BERT and GPT2.

Overall, the abstractive approach to the BERT model has a superior value compared to the extractive approach in Table 7. GPT2 cannot provide optimal results because it cannot capture a good word context, this is because the model only captures information only from the previous word token input. While BERT can understand the context of words from two directions based on the bidirectional architecture. Table 8 is an example of an abstractive-extractive approach for a good summary. We choose the best example based on the rouge value obtained, namely ROUGE-1 (R1) 35.29, ROUGE-2 (R2) 6.06, and ROUGE-L (RL) 33.05. The best summary example is a tourist attraction called Taman Mini Indonesia. In the BERT summary results, only information was obtained related to the 'use of museum props' (marked in blue) and 'cheap entry ticket prices' (marked in orange). Whereas the results of the merger between BERT-EXT provide more information, namely 'there is a transportation museum' (marked in green), 'use of museum props' (marked in blue), 'cheap ticket prices' (marked in orange), and 'an explanation of religions throughout Indonesia' (marked in purple). The BERT-

DOI: https://doi.org/10.29207/resti.v7i6.5170

EXT model is obtained from the BERT results which can only construct information about the 'use of museum props' (marked in blue) and 'cheap ticket prices' (marked in orange). Meanwhile, the addition of extractive information related to 'there is a transportation museum' (marked in green) and 'an explanation of religion' (marked in purple).

The weakness of the abstractive-extractive combination is the addition of inconsistent sentences where the sentence will contain a few or more words. From Table 8, the BERT-EXT model still has 33 words that do not give important meaning to a summary. This makes the summary only provide additional words without getting the main information. This is due to extractive techniques that take sentences directly from documents without any modifications to the sentences. Therefore, the next process can use aspect and opinion extraction, where the selecting sentences focus on those extractions. It can also reduce unnecessary words in a sentence so that the resulting summary will be shorter.

4. Conclusion

In this study, we use abstractive and extractive approaches for summarizing the travel reviews, whereas, in the extractive approach, we modify the model using clustering and a combination of 4 features. The features used are the relevance score, novelty score, position score, and sentiment keyword score. Our extractive model provides increased performance compared to the baseline method. A combination of abstractive and extractive approaches to reduce incorrect and lacking information so that summary performance increases for ROUGE and BERTScore. The BERT-EXT model provides optimal value compared to GPT2-EXT. This value is also higher than just using the baseline method for extractive and abstractive. Other state-of-the-art unsupervised methods such as TextRank, BERT Extractive, and TextTeaser still have lower performance compared to BERT-EXT.

In future work, we will use extracting aspects and opinions of sentences to shorten sentences, so it compresses sentences. It might make the sentence remove unnecessary or consistent words, making it easier to omit the same information. Using LDA to group topic-based probability for reducing similar topics and removing unimportant topics. We compare different models of sentence embeddings for clustering and scoring. The abstractive approach is compared with another pre-trained model to get optimal results.

References

[1] C.-F. Tsai, K. Chen, Y.-H. Hu, and W.-K. Chen, "Improving text summarization of online hotel reviews with review helpfulness and sentiment," *Tour. Manag.*, vol. 80, p. 104122, Oct. 2020, doi: 10.1016/j.tourman.2020.104122.

[2] A. Reyes-Menendez, J. R. Saura, and J. G. Martinez-Navalon, "The Impact of e-WOM on Hotels Management Reputation: Exploring TripAdvisor Review Credibility With the ELM Model," *IEEE Access*, vol. 7, pp. 68868–68877, 2019, doi: 10.1109/ACCESS.2019.2919030.

[3] C. Ma, W. E. Zhang, M. Guo, H. Wang, and Q. Z. Sheng, "Multi-document Summarization via Deep Learning Techniques: A Survey," *ACM Comput. Surv.*, Apr. 2022, doi: 10.1145/3529754.

[4] D. Suleiman and A. Awajan, "Deep Learning Based Abstractive Text Summarization: Approaches, Datasets, Evaluation Measures, and Challenges," *Math. Probl. Eng.*, vol. 2020, 2020, doi: 10.1155/2020/9365340.

[5] A. Bajaj *et al.*, "Long Document Summarization in a Low Resource Setting using Pretrained Language Models," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, 2021, pp. 71–80. doi: 10.18653/v1/2021.acl-srw.7.

[6] Z. Liang, J. Du, and C. Li, "Abstractive social media text summarization using selective reinforced Seq2Seq attention model," *Neurocomputing*, vol. 410, pp. 432–440, Oct. 2020, doi: 10.1016/j.neucom.2020.04.137.

[7] R. Wijayanti, M. L. Khodra, and D. H. Widyantoro, "Indonesian Abstractive Summarization using Pre-Trained Model," in *3rd 2021 East Indonesia Conference on Computer and Information Technology, EIConCIT 2021*, Apr. 2021, pp. 79–84. doi: 10.1109/EIConCIT50028.2021.9431880.

[8] N. Giarelis, C. Mastrokostas, and N. Karacapilidis, "Abstractive vs. Extractive Summarization: An Experimental Review," *Appl. Sci.*, vol. 13, no. 13, p. 7620, Jun. 2023, doi: 10.3390/app13137620.

[9] W. S. El-Kassas, C. R. Salama, A. A. Rafea, and H. K. Mohamed, "Automatic text summarization: A comprehensive survey," *Expert Syst. Appl.*, vol. 165, p. 113679, 2021, doi: 10.1016/j.eswa.2020.113679.

[10] D. Patel, S. Shah, and H. Chhinkaniwala, "Fuzzy logic based multi-document summarization with improved sentence scoring and redundancy removal technique," *Expert Syst. Appl.*, vol. 134, pp. 167–177, Nov. 2019, doi: 10.1016/j.eswa.2019.05.045.

[11] M. R. Ramadhan, S. N. Endah, and A. B. J. Mantau, "Implementation of Textrank Algorithm in Product Review Summarization," Nov. 2020. doi: 10.1109/ICICoS51170.2020.9299005.

[12] Muhammad Ikram Kaer Sinapoy, Yuliant Sibaroni, and Sri Suryani Prasetyowati, "Comparison of LSTM and IndoBERT Method in Identifying Hoax on Twitter," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 7, no. 3, pp. 657–662, Jun. 2023, doi: 10.29207/resti.v7i3.4830.

[13] H. A. Wibowo *et al.*, "Semi-Supervised Low-Resource Style Transfer of Indonesian Informal to Formal Language with Iterative Forward-Translation," in *2020 International Conference on Asian Language Processing (IALP)*, Dec. 2020, pp. 310–315. doi: 10.1109/IALP51396.2020.9310459.

[14] S. Lamsiyah, A. El Mahdaouy, S. E. A. Ouatik, and B. Espinasse, "Unsupervised extractive multi-document summarization method based on transfer learning from BERT multi-task fine-tuning," *J. Inf. Sci.*, vol. 49, no. 1, pp. 164–182, Feb. 2023, doi: 10.1177/0165551521990616.

[15] S. Lamsiyah, A. El Mahdaouy, B. Espinasse, and S. El Alaoui Ouatik, "An unsupervised method for extractive multi-document summarization based on centroid approach and sentence embeddings," *Expert Syst. Appl.*, vol. 167, no. September 2020, p. 114152, 2021, doi 10.1016/j.eswa.2020.114152.

[16] D. Gunawan, A. Pasaribu, R. F. Rahmat, and R. Budiarto, "Automatic Text Summarization for Indonesian Language Using TextTeaser," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 190, p. 012048, Apr. 2017, doi: 10.1088/1757-

DOI: https://doi.org/10.29207/resti.v7i6.5170

899X/190/1/012048.
[17] A. Zaqiyah, D. Purwitasari, and C. Fatichah, "Text Generation with Content and Structure-Based Preprocessing in Imbalanced Data of Product Review," *Int. J. Intell. Eng. Syst.*, vol. 14, no. 1, pp. 516–527, Feb. 2021, doi: 10.22266/ijies2021.0228.48.
[18] S. Cahyaningtyas, D. Hatta Fudholi, and A. Fathan Hidayatullah, "Deep Learning for Aspect-Based Sentiment Analysis on Indonesian Hotels Reviews," *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, Aug. 2021, doi: 10.22219/kinetik.v6i3.1300.
[19] R. M. Alguliyev, R. M. Aliguliyev, N. R. Isazade, A. Abdi, and N. Idris, "COSUM: Text summarization based on clustering and optimization," *Expert Syst.*, vol. 36, no. 1, p. e12340, Feb. 2019, doi: 10.1111/exsy.12340.
[20] W. Jiang, J. Chen, X. Ding, J. Wu, J. He, and G. Wang, "Review Summary Generation in Online Systems: Frameworks for Supervised and Unsupervised Scenarios," *ACM Trans. Web*, vol. 15, no. 3, pp. 1–33, May 2021, doi: 10.1145/3448015.
[21] J. Lovinger, I. Valova, and C. Clough, "Gist: general integrated summarization of text and reviews," *Soft Comput.*, vol. 23, no. 5, pp. 1589–1601, Mar. 2019, doi: 10.1007/s00500-017-2882-2.
[22] N. Hafidz and D. Yanti Liliana, "Klasifikasi Sentimen pada Twitter Terhadap WHO Terkait Covid-19 Menggunakan SVM, N-Gram, PSO," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 5, no. 2, pp. 213–219, Apr. 2021, doi: 10.29207/resti.v5i2.2960.
[23] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt, "YAKE! Keyword extraction from single documents using multiple local features," *Inf. Sci. (Ny).*, vol. 509, pp. 257–289, Jan. 2020, doi: 10.1016/j.ins.2019.09.013.
[24] Kata-Ai, "Kata-ai/indosum: A benchmark dataset for Indonesian text summarization." Accessed: Aug. 7, 2023, [Online]. Available: https://github.com/kata-ai/indosum
[25] IndigoResearch. "IndigoResearch/text teaser: Official version of TextTeaser." Accessed: Aug. 7, 2023, [Online]. Available: https://github.com/IndigoResearch/textteaser
[26] dmmiller612. "DMMILLER612/bert-extractive-summarizer: Easy to use extractive text summarization with Bert." Accessed: Aug. 7, 2023, [Online]. Available: https://github.com/dmmiller612/bert-extractive-summarizer

DOI: https://doi.org/10.29207/resti.v7i6.5170