International Journal of Informatics and Communication Technology (IJ-ICT)
Vol. 13, No. 3, December 2024, pp. 400~409
ISSN: 2252-8776, DOI: 10.11591/ijict.v13i3.pp400-409

Automated multi-document summarization using extractive-abstractive approaches

Maulin Nasari, Abba Suganda Girsang
Department of Computer Science, BINUS Graduate Program - Master of Computer Science, Bina Nusantara University
Jakarta, Indonesia

Article Info

Article history:

Received Feb 27, 2024
Revised Aug 14, 2024
Accepted Aug 27, 2024

Keywords:

BART
Extractive-abstractive
Multi-document summarization
TextRank

ABSTRACT

This study presents a multi-document text summarizing system that employs a hybrid approach, including both extractive and abstractive methods. The goal of document summarizing is to create a coherent and comprehensive summary that captures the essential information contained in the document. The difficulty in multi-document text summarization lies in the lengthy nature of the input material and the potential for redundant information. This study utilises a combination of methods to address this issue. This study uses the TextRank algorithm as an extractor for each document to condense the input sequence. This extractor is designed to retrieve crucial sentences from each document, which are then aggregated and utilised as input for the abstractor. This study uses bidirectional and auto-regressive transformers (BART) as an abstractor. This abstractor serves to condense the primary sentences in each document into a more cohesive summary. The evaluation of this text summarizing system was conducted using the ROUGE measure. The research yields ROUGE R1 and R2 scores of 41.95 and 14.81, respectively.

Corresponding Author:

Maulin Nasari
Department of Computer Science, BINUS Graduate Program - Master of Computer Science
Bina Nusantara University
Jakarta 11480, Indonesia
Email: maulin.nasari@binus.ac.id

1. INTRODUCTION

The automatic document summarization system shortens the length of the document(s) that are being input while preserving all of the information that is pertinent to the situation [1]–[3]. When it comes to summarization approaches, the classification of single-document or multi-document techniques is determined by the number of documents that are input. Additionally, multi-document summarization is a useful tool for consolidating information from a group of related documents to create a concise summary [4], [5]. In contrast, single-document summarization may only partially capture the main topic as it focuses on summarizing just one document. There are two methods that can be utilized in order to achieve this objective: extractive and abstractive. Extractive methods are used to generate summaries by picking the information from the original document(s) that are deemed to be the most important [6], [7]. An extractive method is appropriate for lengthy texts with a well-defined structure, whereas an abstractive method is more ideal for concise writings [8]. Meanwhile, the objective of abstractive methods is to produce new words and phrases in a manner that is analogous to the way in which humans develop summaries. In modern times, an encoder-decoder arrangement is frequently utilized for abstractive summarization [9], [10]. There are two components that make up the architecture, which is referred to as sequence-to-sequence (Seq2Seq) and consists of an encoder and a decoder. Encoder module is responsible for converting the text that is input into a vector representation that is compact. Following that, the output is sent into the decoder module, which is responsible for producing the ultimate abstractive summary [11]. This technique has been used mostly by researchers to summarise single documents. The reason for this is that when summarizing several documents, the summarizer needs to account for a wide range of dependencies, which results in increased computational complexity [12]. Content produced by abstractive approaches frequently has problems including low readability, data repetition, and significant semantic differences from the original source [12]. They are unable to accurately convey the meaning of the material [13]–[15]. Additionally, when input sequences become longer, the attention mechanism can potentially cause diversion or loss of focus [16]. While the extractive methods frequently encounter one-sidedness and limited coverage, hindering their ability to capture the complete semantics of the material [17]. According to the Liu *et al.* [18] and Habib *et al.* [19] study, combining extractive and abstractive approaches can result in higher-quality summaries. They suggest a two-stage hybrid strategy to enhance document summarizing by combining the benefits of abstractive and extractive methods.

Transformers, developed by [17], use the Seq2Seq architecture and are capable of modeling generative tasks on their own. Transformers outperform long short-term memory (LSTM) networks in certain natural language processing (NLP) tasks by effectively handling longer dependencies. Transformers have had a positive influence on pre-trained language models including bidirectional encoder representations from transformers (BERT) [20], XLNet [21], bidirectional and auto-regressive transformers (BART) [22], and text-to-text transfer transformer (T5) [23], enhancing their capabilities. BERT and XLNet exclusively employ encoders, but BART and T5 incorporate both encoder and decoder components. BERT and XLNet is suitable for jobs involving categorization, whereas BART and T5 is suitable for tasks involving generation. Hence, the optimal choice for abstractive summarization would be either BART or T5 [11]. Prior studies on abstractive multi-document summarization were conducted by Beltagy *et al.* [24], Pasunuru *et al.* [25], and Xiao *et al.* [26]. The method developed in [24] utilizes longformer-encoder-decoder (LED) models for multi-document summarization. Pasunuru *et al.* [25] introduced an efficient approach for multi-document summarization by leveraging the BART pre-trained model. The PRIMERA model [26] is a pre-trained model specifically designed for the task of multi-document text summarization. PRIMERA use the sentence generation objective (GSG) to conceal significant sentences that will then be re-predicted in order to generate a summary. Meanwhile, the papers by Aote *et al.* [27], Mojrian and Mirroshandel [28], Sanchez-Gomez *et al.* [29], and Tomer and Kumar [30] propose a method for extractive multi-document summarization. Aote *et al.* [27] utilizes the binary particle swarm optimization (BPSO) technique along with a customized genetic algorithm. The method employed by Tomer and Kumar [30] is firefly-based text summarizing (FbTS). The paper by Sanchez-Gomez *et al.* [29] use the asynchronous parallel MOABC (AMOABC) technique. The papers by Mojrian and Mirroshandel [28] employs the quantum-inspired genetic algorithm (QIGA) technique. The research paper by Fabri *et al.* [31] presented two datasets, multi-XScience and multi-news, which are specifically designed for large-scale multi-document summarization. This dataset is suitable for training summarization models using an abstractive method. Multi-XScience comprises scientific information, whereas multi-news comprises news text. Muniraj *et al.* [32] introduced a single-document summarizing technique employing a hybrid approach. This study employs an extractive-abstractive technique to perform summarization. The employed model is HNTSumm. HNTSumm is a fusion of the TextRank method, which functions as an extractor, and a hybrid sequence-to-sequence encoder-decoder model, which serves as an abstractor. In addition, study by Ghadimi and Beigy [11] introduced a hybrid approach for multi-document summarization. This study utilizes the determinantal point process (DPP) method to generate an initial extraction summary. The DPP approach relies on the fundamental elements of quality and diversity. The deep submodular network (DSN) [33] is used to evaluate quality (relevance) and measure diversity using a BERT-based representation. By employing this method, the DPP is able to allocate a numerical value to every sentence inside the input texts. The sentences with the highest scores are chosen to generate a first extracted summary. Two abstractive summaries are produced by feeding the resulting summary into the pre-trained models, BART and T5. Lastly, the final summary is chosen by comparing the diversity of sentences in each summary; the summary with greater diversity is chosen. Based on the research approaches by Muniraj *et al.* [32] and Ghadimi and Beigy [11], [33], This research uses extractive and abstractive approaches for multi-document summarization.

This research involves using the TextRank algorithm as an extractor and BART as an abstractor. TextRank is an unsupervised graph-based learning system specifically created for extractive summarization in the field of NLP. This approach relies on Google's PageRank algorithm, which utilizes connections to prioritize web sites in search engine rankings [34]. The TextRank algorithm extracts significant text terms by constructing a network with sentences as nodes. Textual sentences must be transformed to vector format. The weight between two nodes is derived using a similarity metric like cosine or Jaccard similarity. In our research, we used the cosine similarity measure. This algorithm works by iteratively updating the weights of nodes in the graph until convergence is achieved. The node with the highest weight is then selected as the key phrase or sentence that best summarizes the document. Overall, the TextRank algorithm is a powerful tool for extractive summarization in NLP [32]. BART is a denoising autoencoder that returns a corrupted document to its original format [22]. The model is a sequence-to-sequence architecture with a bidirectional encoder designed for corrupted text and a left-to-right autoregressive decoder. The BART-large-cnn model is a transformer encoder-decoder (Seq2Seq) that has been pre-trained in English and fine-tuned using the CNN Daily Mail dataset. The model integrates a bidirectional encoder similar to BERT with an autoregressive decoder similar to GPT, which is particularly beneficial for text production tasks such as summarization. The model is pre-trained by perturbing text using a random noising function and subsequently learns to reconstruct the original text. This enables the model to develop robust representations of the input sequences.

In this paper, we propose a multi-document summarization system which combines extractive and abstractive approaches. The system creates an extractive summary by combining several selected sentences or information extracted from each numerous input document, which is then used to construct the input of an abstractive summary. We use the BERT pre-trained language model to embed sentences in a context-aware approach. The graph is formed by the representations and their similarities. The graph is utilised to discard the length of input sequence for abstractive summarization in favor of their shorter, yet related, equivalents. Consequently, lengthier sentences are less likely to be present in the summary that is generated. Removing lengthier sequences also decreases computational time. In order to identify the significant sentences inside a document, we employ the TextRank algorithm, which gives each sentence a score. The selected sentences are those with the highest scores. The obtained summary is subsequently fed into the pre-trained models, BART, to generate abstractive summaries. Following this requirement, this research work has the following objectives: (i) conduct an experiment combining extractive and abstractive summarization methods to tackle input sequence length. (ii) Utilize recall-oriented understudy for gisting evaluation (ROUGE) as the assessment metrics. The next parts of this paper are organised in the following format. Section 2 provides an explanation of our methodology. Section 3 presents the results of the study, whereas section 4 provides the last remarks and conclusions of the study.

2. RESEARCH METHOD

This study employs the multi-news dataset. Multi-news contains news articles and manually created summaries sourced from newser.com. Each summary is meticulously crafted by editors and includes links to the referenced original articles. Our proposed strategy is illustrated in Figure 1. The pre-processing part involves converting the text format to lowercase, eliminating symbols, and deleting HTML tags. After that, each document inside a cluster is segmented into sentences and subsequently went through the sentence embedding process.

2.1. Extractive summarization

In the extractive summarization stage, the first process performed is sentence embedding. The initial step involves adding tokens [CLS], [SEP], and [PAD]. The [CLS] token is added at the beginning of each sentence. The token [CLS] is an essential component placed at the beginning of the input given to BERT, whether it is a single sentence or a pair of sentences. Miller's empirical research [35] shows that calculating the average value of the second-to-last hidden state in the BERT encoder network is more beneficial. The [SEP] token is inserted between sentences as a separator, and the [PAD] token is added at the end of each sentence for padding. The purpose of adding the [PAD] token is to make the length of each sentence uniform. The addition of [PAD] tokens is adjusted based on the longest sentence in a document. In this study, the token length is limited to a maximum of 128 tokens. Therefore, if a sentence exceeds 128 tokens, it is truncated to the maximum limit. Each token has an input ID. Specifically, the input IDs for the [CLS], [SEP], and [PAD] tokens are 101, 102, and 0, respectively, in sequence. These input IDs are then used as input to the BERT model to generate sentence vector representations containing semantic information. The output of the BERT model is a hidden state vector with a size of 768 for each sentence. Therefore, if there are n input tokens, the output of this sentence embedding process is of size $n \times 768$. The vectors representing each sentence have a length of 768 because the BERT model used in this study is $\text{BERT}_{\text{BASE}}$.

Once the vectors for each sentence are obtained, the next step is to represent these sentences in the form of a graph. This graph representation is constructed with each sentence as its node and the relationships between sentences as its edges. The extractive summarization process leverages the TextRank algorithm to perform extractive summarization on a set of documents. It constructs a graph where each sentence is a node, and edges represent the similarity between sentences. The similarity scores are computed based on cosine similarity. The cosine similarity formula can be seen in (1). The system calculates the cosine similarity between sentences using the scikit-learn library.

$$Cosine\ Similarity(S_1, S_2) = \frac{\vec{S}_1 \cdot \vec{S}_2}{|\vec{S}_1| \cdot |\vec{S}_2|} \tag{1}$$

where,

$$\vec{S}_1 = [S_{1,1}, S_{1,2}, \dots, S_{1,m}]\ (Vector\ representation\ of\ sentence\ S_1)$$

$$\vec{S}_2 = [S_{2,1}, S_{2,2}, \dots, S_{2,m}]\ (Vector\ representation\ of\ sentence\ S_2)$$

$$|\vec{S}_1| = \sqrt{S^2{}_{1,1} + S^2{}_{1,2} + \dots + S^2{}_{1,m}}\ (Euclidean\ norm\ of\ vector\ \vec{S}_1)$$

$$|\vec{S}_2| = \sqrt{S^2{}_{2,1} + S^2{}_{2,2} + \dots + S^2{}_{2,m}}\ (Euclidean\ norm\ of\ vector\ \vec{S}_2)$$

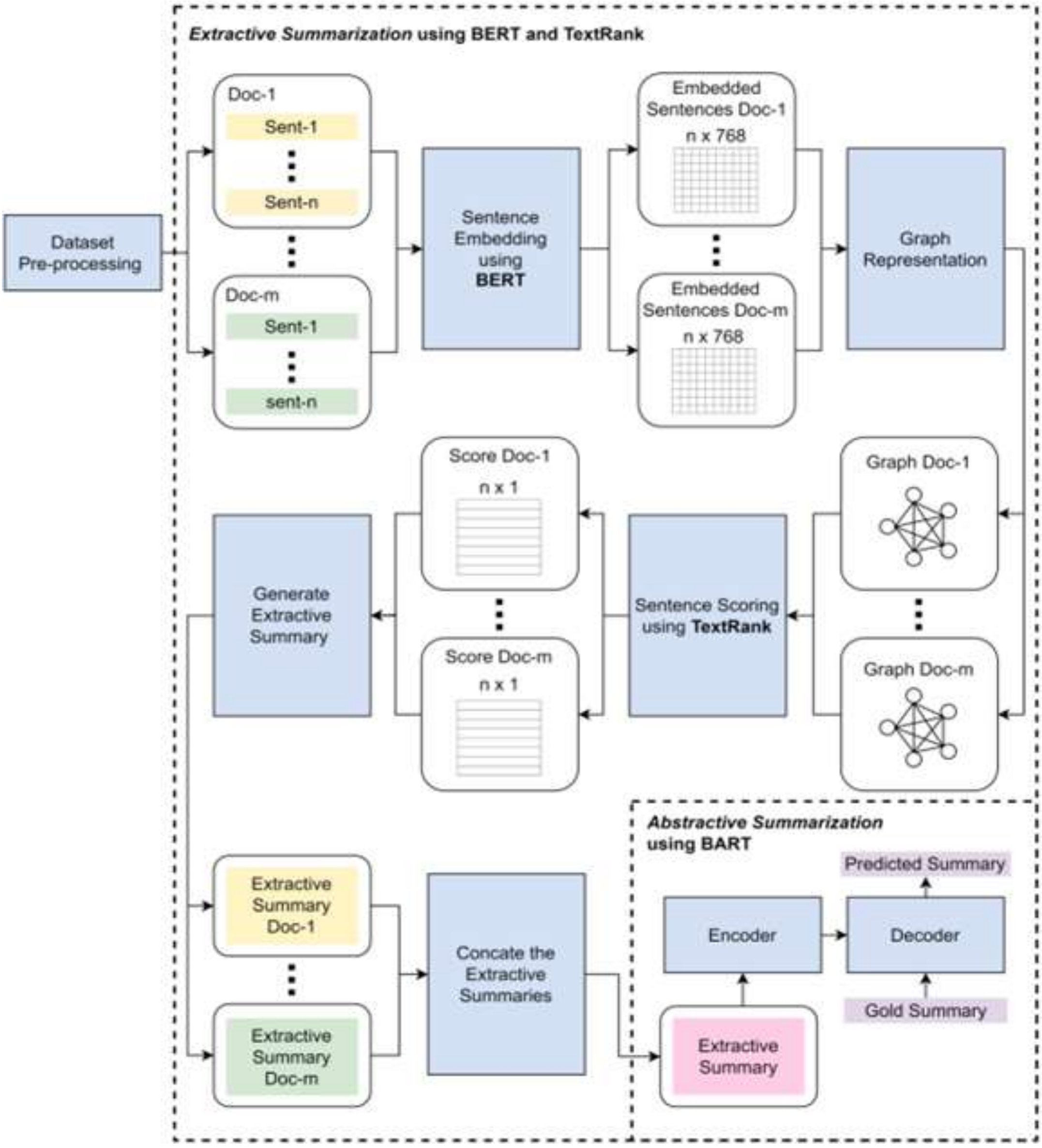

Figure 1. System flowchart

Subsequently, the PageRank algorithm is employed on this graph to identify sentences that are central or important within the document. These important sentences are determined by their PageRank scores. The formula for these scoring methods can be seen in (2). The sentences are sorted according to their scores, and the highest-scoring sentences are chosen as the summary.

$$S(V_i) = (1 - d) + d * \sum_{j \in In(V_i)} \frac{1}{|Out(V_i)|} S(V_j) \quad (2)$$

where,

$V_i = Vertex\ that\ represents\ each\ sentence$

$S(V_i) = The\ score\ of\ a\ vertex\ V_i$

$d = Damping\ factor\ that\ can\ be\ set\ between\ 0\ and\ 1$

$In(V_i) = Set\ of\ vertices\ that\ point\ to\ it\ (predecessors)$

$Out(V_j) = Set\ of\ vertices\ that\ vertex\ V_i\ points\ to\ (successors)$

$|Out(V_j)| = The\ number\ of\ vertices\ in\ the\ set\ Out(V_j)$

The extractive summary is constructed by selecting the n×compression ratio highest-scoring sentences from each document out of n sentences. These extractive summaries from each document are then combined to obtain an overall extractive summary for the entire cluster. This extractive summary subsequently becomes the input for the next stage.

2.2. Abstractive summarization

In this research, the abstractive summarization stage utilizes the BART model, focusing on generating new sentences that represent the core information of news documents with the aim of producing shorter yet coherent and informative summaries. The gold summary, or target summary, existing in the dataset serves as the input to the decoder model. This is done because during the training phase, the BART model is trained to learn from examples of summaries already present in the dataset. By using existing summaries as input to the decoder, the model is taught to understand the structure and writing style desired in the summaries. This helps the model learn linguistic patterns and important information to be included in the summaries. Meanwhile, the previously generated extractive summary is used as input to the encoder. This provides a contextual representation of the input text that is useful in constructing abstractive summaries. By incorporating the extractive summary as input to the encoder, the model can better understand the context of the input text and capture relevant information needed in summary construction.

Before entering the encoder and decoder, both input sequences undergo tokenization. During this stage, special tokens are also added, namely <s> and </s>. The <s> token is added at the beginning of each input sequence, and the </s> token is added at the end of each input sequence. Similar to tokenization in the previous sentence embedding process, this model also has a maximum limit for input tokens in the encoder and decoder. In this study, the maximum input sequence length for the encoder is 1,024, while the maximum input sequence length for the decoder is 128. Subsequently, the output from the encoder in the BART model is fed into the decoder so that the model can understand the context of the input text when constructing summaries. This process enables the decoder to generate relevant and informative summaries by considering contextual information such as topic, structure, and content provided by the original text. Thus, the output of the encoder serves as guidance in the autoregressive decoding process, assisting the model in constructing summaries appropriate to the given input text context. This allows the BART model to produce accurate and connected summaries with the original text, making it effective in handling abstractive summarization tasks. All processes in this abstractive summarization are performed using libraries available in Hugging Face.

2.3. Datasets

In this research, we employ multi-news datasets. The multi-news dataset, presented by [31], performs as a significant dataset for multi-document summarization. The dataset includes articles and attached summaries created by humans, using a format similar to the DUC 2004 dataset but on a bigger size. The dataset was divided into training (80%, 44,972), validation (10%, 5,622), and test (10%, 5,622) sets. The multi-news dataset includes scenarios with 2 to 10 source documents per summary, which corresponds with its goal of multi-document summarization (MDS). The frequency of each example is shown in Table 1.

Table 1. The frequency of multi-news dataset based on the number of sources

#of source	Frequency	#of source	Frequency
2	23894	7	382
3	12707	8	209
4	5022	9	89
5	1873	10	33
6	763		

2.4. Evaluation metrics

This study adopts established evaluation metrics in the text summarization literature, namely, ROUGE. The ROUGE evaluation metrics utilized include ROUGE-N, measuring the similarity of n-grams between sentences. This study employs a supervised dataset annotated by human evaluators for evaluation purposes. The ROUGE calculations in this evaluation phase use the recall formula in (3), precision in (4), and F1-measurement in (5).

$$recall = \frac{number\ of\ overlapping\ words}{total\ words\ in\ reference\ summary} \tag{3}$$

$$precision = \frac{number\ of\ overlapping\ words}{total\ words\ in\ system\ summary} \tag{4}$$

$$F1 - measure = 2\,\frac{precision \times recall}{precision + recall} \tag{5}$$

3. RESULTS AND DISCUSSION

This section presents the findings obtained from running experiments on the proposed summarizer. Additionally, it covers the experimental settings during these experiments to provide context for the results. The evaluation of the summarizer's performance was conducted using ROUGE scores, examining the impact of various compression ratios on the quality of the generated summaries. The experiments aimed to identify the optimal balance between summary length and information retention. The results demonstrate the effectiveness of combining extractive and abstractive approaches in achieving high-quality summarization.

3.1. Experimental settings

In this study, we carried out it in the Google Colab environment using a V100 GPU (16 GB of RAM). Based on the methods and resources we use, we use several hyper-parameters, as mentioned in Table 2. In this table, we outline the key hyper-parameters employed in this study, providing insights into the configuration of our experimental setup.

Table 2. Hyper-parameter setup

Hyper-parameter	Chosen value
BERT setup	bert-base-uncased
BART setup	bart-large-cnn
Batch size	4
Learning rate	0.00005
Weight decay	0.01

Two prominent language models, BERT and BART, serve as the foundational setup for the investigation. Specifically, we have chosen "bert-base-uncased" for the BERT model. The $\text{BERT}_{\text{BASE}}$ has 12 transformer blocks, a hidden size of 768, 12 self-attention heads, and 110 M parameters. This model is used in the sentence embedding process for the extractive summarization part to represent the value of each sentence in the document. For the BART model, we have chosen "bart-large-cnn" for the BART model. The BART-large model, 406 million parameters, features 16 attention heads for each attention layer in the Transformer encoder and decoder, with a hidden size of 1,024 in the transformer blocks. Additionally, critical training parameters are disclosed, including a batch size of 4, a learning rate set at 0.00005, and a weight decay of 0.01.

The disclosed hyper-parameter values reflect a thoughtful selection process, indicating a balance between computational efficiency and model expressiveness. The small batch size of 4 indicates a method that is efficient in terms of resources, possibly designed to match the processing capabilities of the systems being used. Simultaneously, the learning rate parameter is configured to its default value while utilizing the

AdamW optimizer. Furthermore, the weight decay value is selected based on the commonly employed value in the seq2seq trainer.

3.2. Results

This section describes our experiment's performance using ROUGE scores. We utilize five compression ratios to summarize the dataset in the extractive summarization section. The compression ratio is the number of generated summary sentences divided by the number of original sentences. Compression ratios include 75%, 50%, 25%, 20%, and 15%. Table 3 shows the extractive summarization experiment results. This table shows ROUGE scores for extractive summaries at different compression ratios.

Table 3. ROUGE Scores from extractive summary

Extractive compression ratio	ROUGE-1	ROUGE-2
75%	28.53	12.79
50%	33.53	**12.86**
25%	38.18	11.95
10%	**38.48**	11.45
15%	38.11	10.76

The compression ratio significantly impacts results of the extractive summary. As the compression ratio increases, the length of the summary also increases; however, the summary contains more information. In contrast, a lower compression ratio yields shorter summary results, but sacrifices the amount of available information. The average number of words in one document cluster at each compression ratio can be seen in Table 4. The target summary, also known as the gold summary, has an average word count of 217 words. With this golden summary length, the highest R1 score of 38.48 is achieved at a compression ratio of 20%. The highest R2 score of 12.86 is achieved at a compression ratio of 50%.

Table 4. Summary length average from extractive summary

Data	Summary length average				
	75%	50%	25%	20%	15%
Train	1326.372	871.778	450.849	367.937	288.631
Validation	1293.418	849.398	439.804	358.900	281.526
Test	1307.599	858.451	444.286	362.629	284.139

The result derived from the extractive summarization process then serves as the input for the abstractive summarization process. The optimal values for the maximum input sequence and output sequence in the abstractive summarization process were determined by a process of trial and error. The maximum input sequence value was found to be 1,024, while the output sequence value was determined to be 128. When the length of the input sequence is greater than this value, the length of the input will be truncated. The results of the abstractive summarization experiment can be seen in the Table 5. According to the findings of the evaluation of the abstractive summary, it is possible to observe that the rouge value is improved in proportion to the compression ratio of the extractive summary. This is due to the fact that a high compression ratio still stores a significant amount of information, which consequently allows the summary results from the abstractor to contain a greater amount of information. In this study, the optimal rouge value was determined by using a compression ratio of 75%, where R1 was 41.95 and R2 was 14.18.

Table 5. ROUGE scores from abstractive summary

Extractive compression ratio	R1	R2
75%	**41.95**	**14.81**
50%	40.91	13.39
25%	37.50	11.23
20%	38.73	11.78
15%	37.44	10.76

The baseline for this study is previous research by Fabbri *et al.* [31]. This research presents the results of summarization using extractive and abstractive approaches, each separately. Table 6 shows the comparison of evaluation results of the proposed model with other models and the baseline [31]. The table presents the evaluation results of models using extractive and abstractive approaches separately from the research conducted by Fabbri *et al.* [31]. From Table 6, the evaluation results of the proposed model are superior to those of models using extractive approaches. The extractive methods compared include First-1, First-2, First-3, LexRank, TextRank, and maximal marginal relevance (MMR). First-1, First-2, and First-3 are extractive summaries that take the First 1, 2, and 3 sentences, respectively. LexRank and TextRank are graph-based summarization methods that consider relationships between sentences. MMR is an approach to combining query relevance with information novelty in the summarization context. MMR produces a ranked list of candidate sentences based on their relevance and redundancy to the query. The top-ranked sentences are then extracted to form the summary.

Table 6. Comparison of model evaluation results

Model	R1	R2
Extractive methods		
First-1	26.83	7.25
First-2	35.99	10.17
First-3	39.41	11.77
LexRank [36]	38.27	12.70
TextRank [37]	38.44	13.10
MMR [38]	38.77	11.98
Abstractive methods		
PG-Original [39]	41.85	12.91
PG-MMR [39]	40.55	12.36
PG-BRNN [40]	42.80	14.19
CopyTransformer [40]	43.57	14.03
Hi-MAP [31]	43.47	14.89
Proposed method		
TextRank-BART	41.95	14.81

Meanwhile, when compared to models using an abstractive approach, the evaluation results of the proposed model achieve competitive performance. The abstractive methods compared include PG-Original, PG-MMR, PG-BRNN, CopyTransformer, and Hi-MAP. PG-Original and PG-MMR are pointer-generator network models. PG-BRNN is the pointer-generator implementation from OpenNMT2 [13]. CopyTransformer is a model utilizing a transformer architecture with four layers of encoder and decoder. Hi-MAP is a model built on top of PG-BRNN, constructed from a single layer of BiLSTM with a hidden state dimension of 256. In terms of the ROUGE-1 evaluation metric, the proposed model achieves better results than PG-Original and PG-MMR but still falls below PG-BRNN, CopyTransformer, and Hi-MAP. However, concerning the ROUGE-2 evaluation metric, the proposed model outperforms PG-Original, PG-MMR, PG-BRNN, and CopyTransformer but remains below Hi-MAP. This discrepancy may be due to the input sequence of the previous research models being longer than that of the proposed model. Hi-MAP extracts a maximum of 500 tokens from each document in each cluster, while the proposed model extracts a maximum of 1,024 tokens from the entire document in each cluster. Thus, the previous models may capture more information from the original text.

4. CONCLUSION

This study proposes a multi-document summarization system that integrates both extractive and abstractive methods, leveraging the TextRank algorithm for extractive summarization and the BART model for abstractive summarization. The system addresses the challenges posed by lengthy input documents and potential redundancy by using the TextRank algorithm to extract crucial sentences from each document, which are then aggregated and fed into the BART model for further summarization. The evaluation of the proposed system using the ROUGE metric yields competitive results, with R1 and R2 scores of 41.95 and 14.81, respectively. In conclusion, the hybrid approach presented in this study demonstrates the potential of combining extractive and abstractive methods to address the challenges of multi-document summarization. The proposed TextRank-BART model offers a balanced and effective solution, opening avenues for future research in improving and refining multi-document summarization systems.

REFERENCES

[1] A. P. Widyassari *et al.*, “Review of automatic text summarization techniques & methods,” *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 4, pp. 1029–1046, Apr. 2022, doi: 10.1016/j.jksuci.2020.05.006.

[2] M. F. Mridha, A. A. Lima, K. Nur, S. C. Das, M. Hasan, and M. M. Kabir, "A survey of automatic text summarization: progress, process and challenges," *IEEE Access*, vol. 9, pp. 156043–156070, 2021, doi: 10.1109/ACCESS.2021.3129786.

[3] N. I. Altmami and M. El B. Menai, "Automatic summarization of scientific articles: a survey," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 4, pp. 1011–1028, Apr. 2022, doi: 10.1016/j.jksuci.2020.04.020.

[4] C. Ma, W. E. Zhang, M. Guo, H. Wang, and Q. Z. Sheng, "Multi-document summarization via deep learning techniques: a survey," *ACM Computing Surveys*, vol. 55, no. 5, pp. 1–37, May 2023, doi: 10.1145/3529754.

[5] M. Afsharizadeh, H. Ebrhimpour-Komeleh, A. Bagheri, and G. Chrupała, "A survey on multi-document summarization and domain-oriented approaches," *Journal of Information Systems and Telecommunication (JIST)*, vol. 10, no. 37, pp. 68–78, Feb. 2022, doi: 10.52547/jist.16245.10.37.68.

[6] R. Rani and D. K. Lobiyal, "An extractive text summarization approach using tagged-LDA based topic modeling," *Multimedia Tools and Applications*, vol. 80, no. 3, pp. 3275–3305, Jan. 2021, doi: 10.1007/s11042-020-09549-3.

[7] T. Uçkan and A. Karcı, "Extractive multi-document text summarization based on graph independent sets," *Egyptian Informatics Journal*, vol. 21, no. 3, pp. 145–157, Sep. 2020, doi: 10.1016/j.eij.2019.12.002.

[8] R. Liang, J. Li, L. Huang, R. Lin, Y. Lai, and D. Xiong, "Extractive-abstractive: a two-stage model for long text summarization," in *CCF Conference on Computer Supported Cooperative Work and Social Computing*, 2021, pp. 173–184, doi: 10.1007/978-981-19-4549-6_14.

[9] D. Suleiman and A. Awajan, "Deep learning based abstractive text summarization: approaches, datasets, evaluation measures, and challenges," *Mathematical Problems in Engineering*, vol. 2020, pp. 1–29, Aug. 2020, doi: 10.1155/2020/9365340.

[10] M. Zhang, G. Zhou, W. Yu, N. Huang, and W. Liu, "A comprehensive survey of abstractive text summarization based on deep learning," *Computational Intelligence and Neuroscience*, vol. 2022, pp. 1–21, Aug. 2022, doi: 10.1155/2022/7132226.

[11] A. Ghadimi and H. Beigy, "Hybrid multi-document summarization using pre-trained language models," *Expert Systems with Applications*, vol. 192, p. 116292, Apr. 2022, doi: 10.1016/j.eswa.2021.116292.

[12] V. Kosaraju, Y. D. Ang, and Z. Nabulsi, "Faster transformers for document summarization," *Vineet Kosaraju*, no. 8, pp. 1–14, 2019.

[13] A. See, P. J. Liu, and C. D. Manning, "Get to the point: summarization with pointer-generator networks," *arXiv preprint arXiv:1704.04368*, 2017, doi: 10.48550/arXiv.1704.04368.

[14] Z. Cao, W. Li, S. Li, and F. Wei, "Retrieve, rerank and rewrite: soft template based neural summarization," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 152–161, doi: 10.18653/v1/P18-1015.

[15] M. Yang, Q. Qu, W. Tu, Y. Shen, Z. Zhao, and X. Chen, "Exploring human-like reading strategy for abstractive text summarization," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 7362–7369, Jul. 2019, doi: 10.1609/aaai.v33i01.33017362.

[16] M. Gui, J. Tian, R. Wang, and Z. Yang, "Attention optimization for abstractive document summarization," *arXiv preprint arXiv:1910.11491*, 2019, doi: 10.18653/v1/D19-1117.

[17] A. Vaswani *et al.*, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 2017-Decem, no. Nips, pp. 5999–6009, 2017.

[18] W. Liu, Y. Gao, J. Li, and Y. Yang, "A combined extractive with abstractive model for summarization," *IEEE Access*, vol. 9, pp. 43970–43980, 2021, doi: 10.1109/ACCESS.2021.3066484.

[19] M. A. Habib, R. R. Ema, T. Islam, M. Y. Arafat, and M. Hasan, "Automatic text summarization based on extractive-abstractive method," *Radioelectronic and Computer Systems*, no. 2, pp. 5–17, May 2023, doi: 10.32620/reks.2023.2.01.

[20] J. Devlin, M.-W. Chang, K. Lee, K. T. Google, and A. I. Language, "BERT: pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018, doi: 10.48550/arXiv.1810.04805.

[21] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "XLNet: generalized autoregressive pretraining for language understanding," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[22] M. Lewis *et al.*, "BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019, doi: 10.48550/arXiv.1910.13461.

[23] C. Raffel *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020.

[24] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: the long-document transformer," *arXiv preprint arXiv:2004.05150*, 2020, doi: 10.48550/arXiv.2004.05150.

[25] R. Pasunuru, M. Liu, M. Bansal, S. Ravi, and M. Dreyer, "Efficiently summarizing text and graph encodings of multi-document clusters," in *NAACL-HLT 2021 - 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, 2021, pp. 4768–4779, doi: 10.18653/v1/2021.naacl-main.380.

[26] W. Xiao, I. Beltagy, G. Carenini, and A. Cohan, "PRIMERA: pyramid-based masked sentence pre-training for multi-document summarization," *arXiv preprint arXiv:2110.08499*, 2022, doi: 10.48550/arXiv.2110.08499.

[27] S. S. Aote, A. Pimpalshende, A. Potnurwar, and S. Lohi, "Binary particle swarm optimization with an improved genetic algorithm to solve multi-document text summarization problem of Hindi documents," *Engineering Applications of Artificial Intelligence*, vol. 117, p. 105575, 2023, doi: 10.1016/j.engappai.2022.105575.

[28] M. Mojrian and S. A. Mirroshandel, "A novel extractive multi-document text summarization system using quantum-inspired genetic algorithm: MTSQIGA," *Expert Systems with Applications*, vol. 171, p. 114555, Jun. 2021, doi: 10.1016/j.eswa.2020.114555.

[29] J. M. Sanchez-Gomez, M. A. Vega-Rodríguez, and C. J. Pérez, "Parallelizing a multi-objective optimization approach for extractive multi-document text summarization," *Journal of Parallel and Distributed Computing*, vol. 134, pp. 166–179, Dec. 2019, doi: 10.1016/j.jpdc.2019.09.001.

[30] M. Tomer and M. Kumar, "Multi-document extractive text summarization based on firefly algorithm," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, pp. 6057–6065, Sep. 2022, doi: 10.1016/j.jksuci.2021.04.004.

[31] A. R. Fabbri, I. Li, T. She, S. Li, and D. R. Radev, "Multi-news: a large-scale multi-document summarization dataset and abstractive hierarchical model," *arXiv preprint arXiv:1906.01749*, 2020, doi: 10.48550/arXiv.1906.01749.

[32] P. Muniraj, K. R. Sabarmathi, R. Leelavathi, and S. Balaji B, "HNTSumm: hybrid text summarization of transliterated news articles," *International Journal of Intelligent Networks*, vol. 4, pp. 53–61, 2023, doi: 10.1016/j.ijin.2023.03.001.

[33] A. Ghadimi and H. Beigy, "Deep submodular network: an application to multi-document summarization," *Expert Systems with Applications*, vol. 152, p. 113392, Aug. 2020, doi: 10.1016/j.eswa.2020.113392.

[34] C. Mallick, A. K. Das, M. Dutta, A. K. Das, and A. Sarkar, "Graph-based text summarization using modified TextRank," in *Soft Computing in Data Analytics: Proceedings of International Conference on SCDA 2018*, 2018, vol. 758, pp. 137–146, doi: 10.1007/978-981-13-0514-6_14.
[35] D. Miller, "Leveraging BERT for extractive text summarization on lectures," *arXiv preprint arXiv:1906.04165*, 2019, doi: 10.48550/arXiv.1906.04165.
[36] G. Erkan and D. R. Radev, "LexRank: graph-based lexical centrality as salience in text summarization," *Journal of Artificial Intelligence Research*, vol. 22, pp. 457–479, 2004, doi: 10.1613/jair.1523.
[37] R. Mihalcea and P. Tarau, "TextRank: bringing order into text," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004, pp. 404–411.
[38] J. Carbonell and J. Goldstein, "The use of MMR, diversity-based reranking for reordering documents and producing summaries," in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, 1998, pp. 335–336, doi: 10.1145/3130348.3130369.
[39] L. Lebanoff, K. Song, and F. Liu, "Adapting the neural encoder-decoder framework from single to multi-document summarization," *arXiv preprint arXiv:1808.06218*, 2018, doi: 10.48550/arXiv.1808.06218.
[40] S. Gehrmann, Y. Deng, and A. M. Rush, "Bottom-up abstractive summarization," *arXiv preprint arXiv:1808.10792*, 2018, doi: 10.48550/arXiv.1808.10792.

BIOGRAPHIES OF AUTHORS

Maulin Nasari is currently pursuing her master's degree in Computer Science at Bina Nusantara University in Jakarta, Indonesia. She earned her bachelor's degree in Telecommunication Engineering from the School of Electrical Engineering at Telkom University, Bandung, Indonesia, in 2022. Throughout her academic journey, she has demonstrated a passion for exploring the realms of machine learning, deep learning, computer vision, and natural language processing. Her dedication to these fields is evident through her diverse experiences, including participating in the Machine Learning Cohort at Bangkit Academy in 2022, serving as a Research Assistant at the IMV Laboratory from 2021 to 2022, and contributing as a Practicum Assistant at the Basic Computing Laboratory from 2019 to 2021. She can be contacted at email: maulin.nasari@binus.ac.id.

Abba Suganda Girsang is currently a lecturer at Master in Computer Science, Bina Nusantara University, Jakarta, Indonesia Since 2015. He got Ph.D. degree in 2015 at the Institute of Computer and Communication Engineering, Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan. He graduated bachelor from the Department of Electrical Engineering, Gadjah Mada University (UGM), Yogyakarta, Indonesia, in 2000. He then continued his master's degree in the Department of Computer Science at the same university in 2006–2008. He was a staff consultant programmer in Bethesda Hospital, Yogyakarta, in 2001 and also worked as a web developer in 2002–2003. He then joined the faculty of the Department of Informatics Engineering in Janabadra University as a lecturer in 2003-2015. His research interests include swarm, intelligence, combinatorial optimization, and decision support system. He can be contacted at email: agirsang@binus.edu.