MULTIBERTSUMM: EXTRACTIVE MULTI-DOCUMENT SUMMARIZATION USING BERT

by

Junqiao Qu

Submitted in partial fulfillment of the requirements
for the degree of Bachelor of Computer Science with Honours

at

Dalhousie University
Halifax, Nova Scotia
April 2022

Table of Contents

Abstract

With the advancements of Transformer-based methods, there has been a lot of research focused on extractive summarization system of single document. However, the area of multi-document summarization is less explored.

Motivated by the above observation, we proposed MultiBERTSumm, an extractive multi-document summarization system using BERT based model. In this thesis, we leverage BERT based model as a sentence encoder to generate a sentence embedding, which contains contextual semantic information and relation features. Then a Transformer model is applied as a sentence extractor to determine whether it is selected as a summary with the input of sentence embedding. The sentence encoder and sentence extractor are jointly fine-tuned and trained on CNN/DailyMail dataset, then generate a summary for each document. After that, a synthesis module is proposed to calculate a score, which assesses the commonalities between the current sentence and sentences in the other document's summary. The score is based on a similarity metric ROUGE, and sentences that have the highest score will be selected to form the final summary. The overall performance of the system is evaluated on the DUC2004 dataset. The result from our experiments shows that our system achieved a competitive performance and is capable of synthesizing the key information from multiple documents.

Acknowledgements

First and foremost, I would like to thank my supervisor, Professor Evangelos Milios, who has provided valuable guidance to the research and thesis writing process. Your insightful feedback gives me a deeper understanding of automatic text summarization and brought my work to a higher level.

I would also like to show gratitude to all academic resources provided by Dalhousie University. During my undergraduate study, I have learned a lot in the subject of computer science.

Most importantly, none of this could have happened without the support of my family. Thank my parents for giving me unceasing encouragement and continuous support for my further study.

Chapter 1

Introduction

With the explosive growth of text information in recent years, people have access to a large amount of text, such as news, blog, report, and paper on a variety of topics. However, due to the redundancy in the information, it becomes difficult to extract useful information from huge text. It calls for a robust system collecting key phrases and sentences that best represent the content, which can generate a succinct summary of documents by reducing information redundancy. There has been a lot of research focused on extractive summarization system of single document. However, the area of multi-document summarization is less explored.

Multi-document summarization is defined as the process of generating a summary from a number of related documents, which can be useful in many scenarios, for example summarizing all restaurant reviews, or multiple news articles about an event. It is more challenging than a single-document summary since it has to solve the problem of overlapping information among sentences from different documents [1]. Building such a system can be accomplished through two paradigms. The first approach is abstractive summarization, which is the task of generating a short and concise summary that captures the salient ideas of the source text. The generated summaries potentially contain new phrases and sentences that may not appear in the source text. The second approach is extractive summarization, which selects a subset of sentences from the text to form a summary [12]. Abstractive summarization is very complex and relatively more difficult than extractive summarization because it requires extensive natural language processing. As of now, extractive summarization yield better and more grammatically correct results than the abstractive paradigm [5].

Several methods have been proposed for extractive summarization such as the clustering based methods, graph based methods, and deep learning based methods [7]. Recently, the advancements of Transformer-based methods, such as Bidirectional

Encoder Representations from Transformers [3], have pushed the state-of-the-art performance in many natural language processing tasks ranging from sentiment analysis, to question answering, natural language inference, named entity recognition, and textual similarity. Many studies have applied BERT based model to extract contextual representation for extractive summarization and enhance the ability to extract important information from a single document. However, how to synthesize key information from a multiple document cluster is not well explored. Therefore, we will focus on improving the performance of multi-document summarization system and making the final summary contain more comprehensive information. More specifically, we will explore how to leverage the high performance of BERT based model in the single document summarization and apply it in the multi-document summarization task.

In this paper, we propose an extractive multi-document summarization system, MultiBERTSumm, to explore the relationship between multiple document texts and the final summary. The system could take documents of any number, which is a document cluster, as the input. By using BERT based model as a sentence encoder, we are able to create a sentence embedding that includes context and relational information. Then Transformer model is used as a sentence extractor to generate a score for each sentence with the input of sentence embedding. We trained and fine-tuned sentence encoder and sentence extractor on a large corpus, CNN/Daily Mail Dataset, to enhance their ability to extract semantic relations and contextual information. According to the score of each sentence, we can select n sentences that have the highest score as the summary of each document. After that, a synthesis module is then proposed to compare the similarities and differences between each candidate single document summary. To examine the performance of the system, we use ROUGE score to compare generated summaries with human-written summaries in DUC2004.

This paper is organized as follows. The next Chapter 2 Related Work provides a literature review of the relevant method for extractive summarization, including clustering based extractive summarization, graph based extractive summarization and deep learning based summarization. Then Chapter 3 Method will discuss the detailed method used by our model, including the overall architecture, sentence encoder, sentence extractor, synthesis module. The design of the experiments, dataset, and

results are discussed in Chapter 4 Experiment. Conclusion about works and future research will be included in the last Chapter 5.

Chapter 2

Related Work

2.1 Clustering Based Extractive Summarization

The clustering algorithm is a traditional but effective approach for extractive summarization. It considers both relevance and redundancy removal in the generated summary. Typically, clustering based extractive summarization contains the following steps. First, find a vector representation of each sentence. This could be embeddings generated by Word2Vec or other forms of vector representation, eg. Bag-Of-Words or TF-IDF. This vector representation is used to bridge the gap between human understanding of language to that of a machine. Second, using a clustering algorithm to cluster the input sentences. Generated clusters containing similar sentences are considered proxies for topics. And clusters with many sentences represent important topic themes in the input. Selecting several representative sentences from each main cluster is one way to produce an extractive summary while minimizing possible redundancy in the summary. The representative sentence is the most central sentence, which can cover the important information related to the topic of the cluster. The common way to measure sentence centrality is to look at the centroid of the cluster in a vector space. The closer a sentence is to the document centroid the more important it is. Then n representative sentences could be selected from each cluster.

Clustering based summarization is suitable for multi-document summarization because it groups different sentences about the same topic in the documents. However, it requires prior specification of the number of clusters [4]. The highly scored sentences may be similar, therefore redundancy removal techniques are required. Some sentences may express more than one topic but each sentence has to be assigned to only one cluster. As for the performance, the recent deep learning based models usually perform better than the centroid or clustering based methods.

2.2 Graph Based Extractive Summarization

Graph based models have been a popular way to extractive document summarization for decades. In these methods, documents or clusters of documents are represented using sentence based graphs (e.g. LexRank [6], TextRank [14]). Efforts have been made to improve the performance of summarization systems by using discourse relationships between sentences. For example, in LexRank, the model first uses an undirected graph to represent the document sentences. In this graph, each node in the graph represents a sentence from the input text and for each pair of sentences, the weight of the connecting edge is the semantic similarity between the two corresponding sentences using cosine similarity. Second, a ranking algorithm is used to determine the importance of each sentence[6]. The sentences are ranked based on their LexRank scores in a similar way to the PageRank algorithm except that the LexRank graph is undirected.

Recently, there is a considerable amount of research in applying the graph with neural networks, leveraging Graph Convolutional Network to solve the extractive summarization task. Many graph based models give new state-of-the-art performance [9] [22] [20]. These new graph based methods exploit the representational power of deep neural networks and the sentence relation information encoded in graph representations of document clusters. In most cases, a graph based on sentence relations and semantic features will be constructed. Then a Graph Convolutional Network is imposed to capture long-range interactions based on the constructed graph. Through multiple layer-wise propagations, the GCN generates high-level hidden sentence features for salience estimation [22].

2.3 Deep Learning Based Extractive Summarization

In recent years, deep learning based extractive summarization has become popular in the research of extractive summarization. Thanks to the emergence of large-scale news article datasets, more complex networks could be trained. In these methods, a common way is to generate embedding to get a representation of text, which could be the input of a deep network. A document is considered as a bag-of-sentences and a sentence as a bag-of-words. A deep learning model could be trained to generate

a better sentence embedding during a supervised training process. Each sentence in the document will be transformed into a representation vector containing contextual and semantic information. And then a classifier could be trained to classify whether a sentence should be included in the final summary. The task is formalized as the problem of maximizing a sub-modular function defined by the negative summation of the nearest neighbors distances on embedding distributions [4]. Cheng and Lapata [2] developed a sentence extractive model that uses a word level CNN to encode sentences and a sentence level sequence-to-sequence model to predict which sentences to include in the summary. Nallapati et al. [15] proposed a different model using word-level bidirectional RNNs along with a sentence level bidirectional RNN for predicting which sentences should be extracted. Their sentence extractor creates representations of the whole document and computes separate scores for salience, novelty, and location. Typically, a deep learning based extractive summarization will have a sentence encoder and sentence extractor architecture to enable the model to be trained and fine-tuned on large-scale datasets.

With the advancement of Transformer model, deep learning based model has yielded better results compared with other methods, and many novel models achieved new state-of-art performance on extractive summarization tasks. Therefore, due to the high performance, we will build our multi-document summarization system based on a deep learning model.

Chapter 3

Methodology

The problem of multi-document summarization could be defined as the selection process a summary consisting of n sentences from a cluster of documents, which is assumed to contain the most important information. The Fig. 3.1 shows the process of multi-document summarization task. Let $C = D_1, D_2, ..., D_m$, where D_i is $i-th$ document in this cluster. And we can obtain a candidate sentence set CS from all documents from D_1 to D_m, where $CS = S_1, S_2, ..., S_n$, containing n sentences. The task of multi-document extractive summarization can be described as assigning a label $y_i \in [0, 1]$ to each S_i indicating whether or not it should be included in the summary. Finally, the selected sentences form the summary set $SS = S_x, S_y, S_z, ...$ of size L. Note that $L << n$ and $SS \subset CS$ [24].

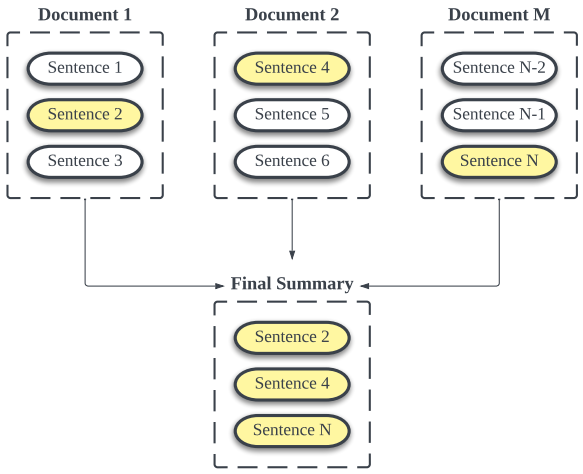

Figure 3.1: The Definition of Multi-Document Summarization System. White rectangles indicate sentences in the document. Yellow rectangles indicate the selected sentences to form the final summary. The goal is to generate a final summary containing comprehensive information from multiple documents.

3.1 Overall Architecture

We take the multiple documents of the same domain as input of the whole system. However, we will summarize each document separately, then use a synthesis module to find what is common in each summary. The overall architecture of MultiBERTSumm has shown in Fig. 3.2. In the summarization process, we use a sentence tokenizer to split a document into multiple sentences. Then each sentence will be sent to a sentence encoder, which is a pre-trained BERT based model to get the embedding of each sentence. A Transformer could be used as a sentence extractor to capture inter-sentence relations and predict the probability that a sentence would be selected in summary. With a parameter n controlling the length of summary, we could generate a candidate summary consisting of n sentences that have the highest probability. Finally, the summary of each document will be used as input for the synthesis module to find the commonalities of each candidate summary.

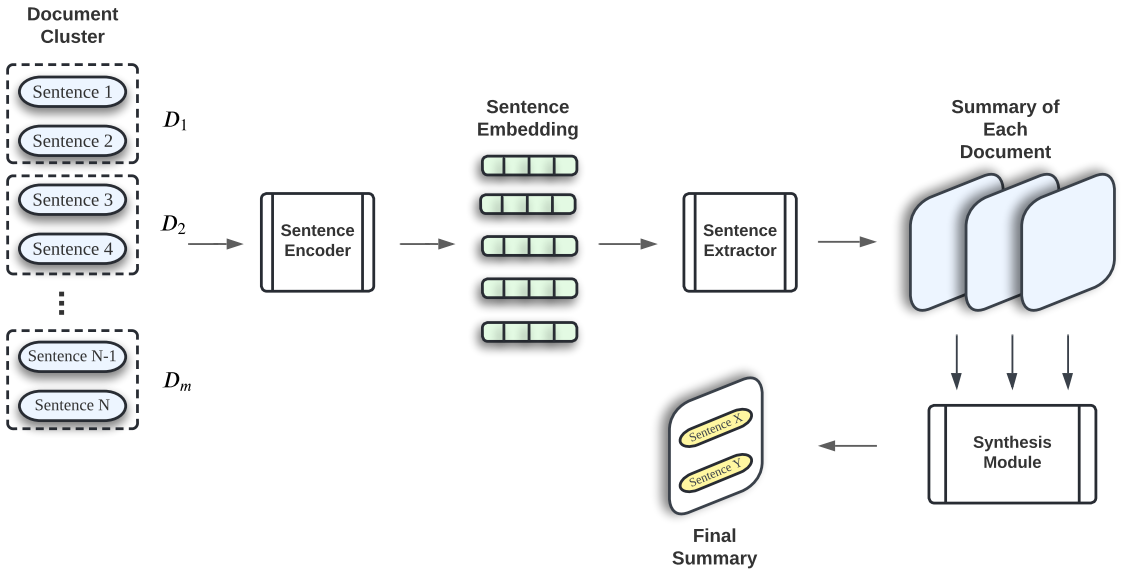

Figure 3.2: Overview of MultiBERTSumm. The blue rectangle indicates each sentence in the document. Green blocks indicate the sentence embeddings generated by the sentence encoder. Blue squares indicate candidate summaries of each document generated by the sentence encoder. The white square is the final summary containing sentences selected by the synthesis module.

3.2 Sentence Encoder

In natural language processing, it is hard to make use of text directly. A common and effective method in deep learning is to find a representation for the text input. An embedding is a learned representation for text where texts of similar meaning will have a similar representation. It is this approach to representing words and documents that may be considered one of the key breakthroughs of deep learning on challenging natural language processing problems. The sentence encoder is therefore used to find such an embedding for each sentence.

Bidirectional Encoder Representations from Transformers (BERT) is a Transformer-based model, which has reportedly obtained some very significant results on many tasks. In original BERT, it is used to generate embedding at the word level, which means the output vectors are limited to tokens instead of sentences. The modified mechanism for summarization tasks has shown in Fig. 3.3. Compared with the original version of BERT, the interval segment embedding was added along with other embeddings. In the interval segment embeddings, it uses different embeddings A, and B to encode sentence boundary information and can distinguish multiple sentences within a document[12]. Therefore, the modified mechanism is applied in our multi-document summarization system, which gives us the ability to find a sentence level representation for extractive summarization tasks.

Apart from the original BERT model, there are many improved versions of BERT recently proposed by researchers. RoBERTa [13], DistilBERT [16] are both models which is proved to be effective on many tasks. For our choice of sentence encoder, we will experiment with these BERT based models to achieve better performance.

The input of the sentence encoder should be the raw text from each document. If a sentence length is smaller or larger than the limit of BERT, it will be padded or truncated. After the pre-processing and cleaning of data, the sentence of the document will be put into the BERT model. Ideally, the encoder could capture some of the semantics of the text and contextual features.

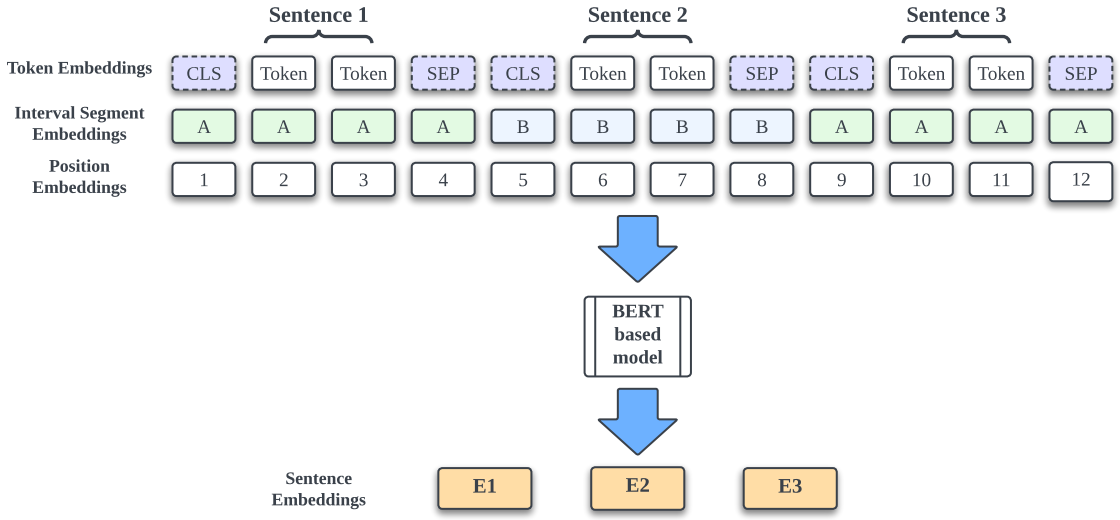

Figure 3.3: The Sentence Encoder Architecture for our Sentence Encoder. It uses a modified mechanism in[12]. In addition to token embeddings and positional embeddings, Interval segment embeddings are added to the original BERT based model.

3.3 Sentence Extractor

With the output embedding of the sentence encoder, a sentence extractor is used to capture inter-sentence relationships for extracting summaries and find out which sentence should be included in the summary of each document. It works as a classifier, which is responsible for removing the hidden features from each sentence embedding and converting them to a single number.

A Transformer is a deep learning model that adopts the mechanism of self-attention, differentially weighting the significance of each part of the input data[19]. Several experiments demonstrate that it achieves good performance at finding inter-sentence relationships and extracting document-level features, and outperforms simple sigmoid classifier and LSTM classifier. Accordingly, we will use the Transformer model as the sentence extractor in the MultiBERTSumm. And we decide to apply the architecture of Transformer defined in [12].

In Transformer, for each sentence S_i, it will output the final predicted score y_i. It is the probability that the sentence will be included in the current single document summary. In the end, the sentence extractor will extract n top-ranked sentences by model predicted score as the summary of the current document.

3.4 Synthesis Module

The sentence extractor will generate a summary for each document in the cluster. However, it still has some redundant information and does not fully utilize important information from multiple documents. Therefore, we proposed a synthesis module to find the commonalities between each summary of single document, and then synthesize them into a final summary of multi-document clusters. Across the summaries of different articles, there will be sentences that contain similar information.

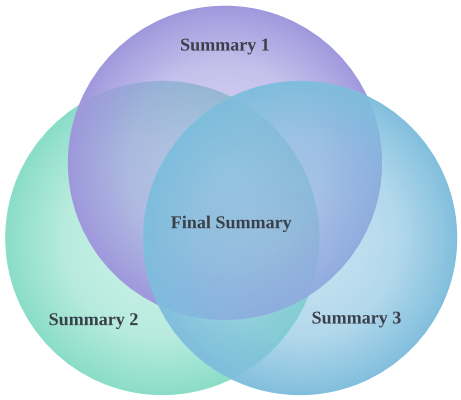

Figure 3.4: The Assumption of our Synthesis Module. Circles indicate the summary of each document in a cluster. The overlapping information in each summary is more comprehensive and worth being selected in the final summary.

The assumption for synthesis module is shown in the Fig. 3.4. Our synthesis module is based on such assumption that information that occurs in many of the input summaries is likely important and worth selecting in the final summary. Therefore, the module will generate a score for each sentence in the single document summary.

The ROUGE scoring and sentence selecting process of our synthesis module is shown in Fig. 3.5. The score is calculated by aggregating ROUGE scores between the current sentence and the other document summary. ROUGE-N is an n-gram recall between a candidate summary and a set of reference summaries. Here we use ROUGE-1 score to assess how much overlap the current sentence has with the other document summaries. If a sentence contains a large amount of information, which is also mentioned by other document summaries, then we assume the sentence has more comprehensive information and could be selected to form the final summary.

Suppose cluster contains n documents. S_i represent the summary of i^{th} document

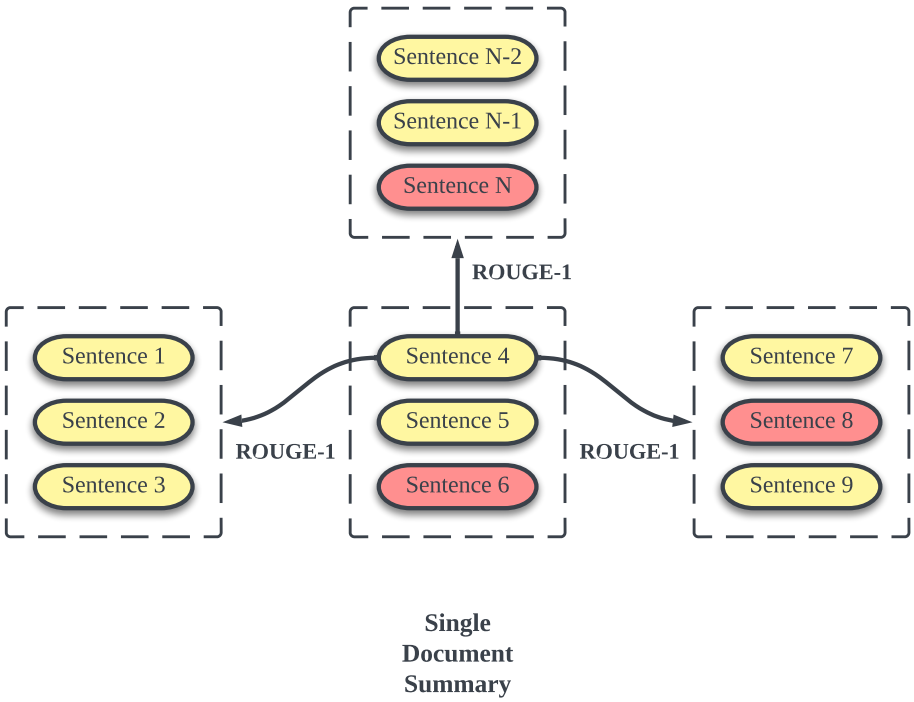

Figure 3.5: The Scoring and Sentence Selecting Process of Synthesis Module. Yellow rectangles indicate each sentence in the single document summary. Red rectangles indicate the final selected sentences in the final summary. This process will calculate ROUGE-1 score between a sentence and other single document summaries to measure the commonalities.

in cluster, and S_{ij} represent the j^{th} sentence in S_i. Formally, a score for a sentence can be defined as follows:

$$Score(S_{ij}) = \sum_{k,k \neq i}^{n} ROUGE\text{-}1_{F1}(S_{ij}, S_k)$$

Details of the calculation of ROUGE score will be discussed in Section 4.2.2. After obtaining the score for each sentence, we could use Top-N strategy to select sentences with highest score to generate the final summary of a multi-document cluster.

Chapter 4

Experiments

4.1 Dataset

4.1.1 CNN/Daily Mail

The CNN/Daily Mail Dataset is an English-language dataset containing just over 300k unique news articles as written by journalists at CNN and the Daily Mail. Each article has a brief human-written summary. The topics of articles are a mix of politics, sports, and entertainment. The articles have an average of 766 words and 29.74 sentences in articles, 53 words, and 3.72 sentences in human-written summaries.The corpus has 286,817 training pairs, 13,368 validation pairs, and 11,487 test pairs [8],[17].

4.1.2 DUC 2004

The DUC2004 dataset is a dataset for document summarization. It is designed and used for evaluation only. The documents are all from the news domain and a collection of documents is grouped into a cluster. The dataset has 50 clusters and each cluster has 4 reference summaries written by human experts. These summaries were not focused in any particular way beyond the documents. Each document in a cluster is describing the same topic and has a medium length, and therefore DUC 2004 is a good dataset to evaluate the multi-document summarization system.

4.2 Metrics

4.2.1 Baseline

We use the lead summary as our baseline, i.e. taking the first n sentences of the document as a summary, where n is the summary length parameter used in the sentence extractor. The lead summary will contain the same number of sentences as model generated summary. Even though it is extremely simple, this method is still

a strong baseline for document summarization, especially for documents in the news domain.

4.2.2 ROUGE

We evaluate summary quality using ROUGE score in our experiments. ROUGE-N is an n-gram recall between a candidate summary and a set of reference summaries. Original ROUGE-N score is defined as follows [11]:

$$ROUGE\text{-}N_{Recall}(candidate, reference)$$

$$= \frac{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count_{match}\left(gram_n\right)}{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count\left(gram_n\right)}$$

In this formula, n stands for the length of the n-gram ($gram_n$). And $Count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in a candidate summary and a set of reference summaries. However, ROUGE-N F1 score is a more popular metric for the performance of the extractive summarization system used in many research papers. So in our paper, we use ROUGE-N F1 to represent ROUGE-N score. ROUGE-N F1 is defined as follows:

$$ROUGE\text{-}N_{Precision}(candidate, reference)$$

$$= \frac{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count_{match}\left(gram_n\right)}{\sum_{S \in \{SystemSummaries\}} \sum_{gram_n \in S} Count\left(gram_n\right)}$$

$$ROUGE\text{-}N_{F1}(candidate, reference) = 2 \times \frac{ROUGE\text{-}N_{Precision} \times ROUGE\text{-}N_{Recall}}{ROUGE\text{-}N_{Precision} + ROUGE\text{-}N_{Recall}}$$

It is worth noting that in DUC 2004, there exist multiple human written reference summaries in each document cluster. We adopt the standard calculating approach for multiple references ROUGE score mentioned in [11]. When multiple references are used, we compute pairwise summary level ROUGE-N between a candidate summary S and every human written reference summary R_i. We then take the maximum of pairwise summary-level ROUGE-N scores as the final multiple reference ROUGE-N score. It is defined as follows:

$$ROUGE\text{-}N_{multi} = argmax_i ROUGE\text{-}N\left(R_i, S\right)$$

We also considered ROUGE-LCS score, which is also a common metric to estimate the similarity between candidate summary and reference summary. It finds longest common subsequence (LCS) and take the union LCS matches between a reference summary sentence r_i and every candidate summary sentence c_j [11]. Suppose that the reference summary contains u sentences and m words in total, and the candidate summary contain v, n respectively. A formal definition of ROUGE-LCS could be given as [11]:

$$R_{lcs} = \frac{\sum_{i=1}^{u} LCS_{\cup}(r_i, C)}{m}$$

$$P_{lcs} = \frac{\sum_{i=1}^{u} LCS_{\cup}(r_i, C)}{n}$$

$$ROUGE\text{-}LCS = \frac{(1+\beta^2) R_{lcs} P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}}$$

In this formula, a common choice for β is $P_{lcs}/(R_{lcs} + 10^{-12})$. And an advantage of using LCS is that it does not require consecutive matches but in-sequence matches that reflect sentence level word order.

4.2.3 Gold Label

The supervised training needs a label for each sentence, which indicates whether or not it should be extracted. Since we are focusing on extractive summarization and not changing the original sentence, we only need to select sentences that will maximize ROUGE-1 score on human-written summary as gold summary. Therefore, a gold label for each sentence could be generated by the Oracle algorithm, which will greedily optimize ROUGE-1 score. It means if we had clairvoyant knowledge of the human reference summary, the oracle system achieves the maximum possible ROUGE scores[10]. Also, the Oracle Summary algorithm could be used to measure the performance ceiling of extractive summarization systems. See Appendix for the details of the Oracle Summary algorithm.

4.3 Training

We first split text in the CNN/DM dataset into sentences using the spaCy open source library and then input the raw text of sentences to the model to start training.

The sentence encoder will start from a pre-trained model. Then sentence extractor combined with the sentence encoder will generate a probability y_i that indicates how likely is it to be selected to form the summary. And gold label Y_i of each sentence in a document could be generated by the Oracle Summary algorithm. Then we can calculate the loss of the whole model using the Binary Classification Entropy of y_i against gold label Y_i. This is a supervised training process, and the sentence encoder and sentence extractor will be jointly fine-tuned on the CNN/DM dataset using several epochs.

4.4 Results

We experiment with common choices of BERT based model, original BERT, RoBERTa, and DistilBERT for sentence encoder to find a better sentence representation. In this experiment, the layer of sentence extractor Transformer is set to 2, which is reported as the best choice for Tranformer classifier[12]. The experimental results of sentence encoder and sentence extractor on CNN/Dailymail datasets are shown in Table 4.1. In our experiment, original BERT achieves 42.56 in ROUGE-1, 19.59 in ROUGE-2, and 38.96 in ROUGE-L. And DistilBERT achieves 42.44 in ROUGE-1, 19.50 in ROUGE-2, and 38.85 in ROUGE-L. There is no significant difference between the result of these two models. But DistilBERT has fewer parameters and it needs less training time. So it would be a better choice for the sentence encoder. The RoBERTa, however, has a much better performance according to ROUGE-N results. It achieves 43.32 in ROUGE-1 score, 20.43 in ROUGE-2 score, and 39.75 in ROUGE-L score, which outperforms baseline and other classic models. Also, it has an improvement on other BERT based models of more than 0.7 ROUGE-1 score. A possible drawback for RoBERTa model could be its much longer training time, due to more parameters.

After that, We further evaluated the performance of our MultiBERTSumm on the multi-document summarization dataset DUC2004. Each cluster contains 10 articles and 4 human-written summaries. We use our proposed multi-document summarization system to generate a summary for each cluster. Then calculate ROUGE score between generated summary and reference summaries in turn. We use the reference summary that can maximize ROUGE-1 score, and discard the rest of three. It means we will take the maximum ROUGE-1 score as the evaluation result of a cluster.

Model	ROUGE-1	ROUGE-2	ROUGE-L
Lead*	40.43	17.62	36.67
BANDITSUM* [21]	41.50	18.70	37.60
JECS* [21]	41.70	18.50	37.90
DistilBERT	42.44	19.50	38.85
BERT	42.56	19.59	38.96
RoBERTa	**43.32**	**20.43**	**39.75**
Oracle*	52.59	31.23	48.87

Table 4.1: Fine-tune Results on the CNN/DailyMail dataset. Training parameters can be found in Appendix C. ROUGE F1 is reported. Results with * are taken from the corresponding papers. Bold indicates the best result among all methods. RoBERTa achieves the best performance on CNN/Daily Mail in all evaluation metrics.

Our experiment result of MultiBERTSumm between difference choice of sentence encoder is given in the Table 4.2. The DistilBERT and RoBERTa both have improved on performance to some extent. It is quite surprising that DistilBERT achieves the best performance in terms of ROUGE-1 on the DUC2004 dataset. It performs even better than RoBERTa, which has more parameters and a better fine-tuned result on CNN/DailyMail. In terms of ROUGE-2 and ROUGE-L, RoBERTa model still has the best performance. It is corresponding to the fine-tuned result on CNN/DailyMail. Considering its higher performance in summarization, we will use RoBERTa as our sentence encoder.

Sentence Encoder	ROUGE-1			ROUGE-2			ROUGE-L		
	precision	recall	fmeasure	precision	recall	fmeasure	precision	recall	fmeasure
DistilBERT	**43.20**	**40.79**	**41.51**	13.25	**12.52**	12.71	29.07	27.58	28.00
BERT	42.59	40.12	40.90	12.66	11.73	12.03	29.88	28.15	28.69
RoBERTa	43.05	40.67	41.41	**13.32**	**12.52**	**12.76**	**30.08**	**28.69**	**29.07**

Table 4.2: Evaluation Results of MultiBERTSumm on DUC 2004 dataset. Bold indicates the best result among all methods. MultiBERTSumm with RoBERTa achieves the best performance on DUC 2004 in most evaluation metrics.

We also compare the MultiBERTSumm with other multi-document summarization systems. The experimental results are shown in Table 4.3. We can see our proposed model achieved 41.41 in ROUGE-1 score and 12.76 in ROUGE-2, which outperforms other multi-document summarization systems by more than 2 ROUGE-1 scores. In many related papers, they did not provide ROUGE-L data. So it is hard to

compare the model performance in terms of ROUGE-L. But we still provide ROUGE-L score for further evaluation purposes. Our model achieves a 29.07 ROUGE-L score, which outperforms the multi-document summarization system based on Maximum Independent Set [18] by 1.71 ROUGE-L.

Model	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**
G-Flow* (Christensen et al., 2013)[23]	35.30	8.27	-
Maximum Independent Set* [18]	35.88	7.79	27.36
GreedyKL* (Haghighi and Vanderwende, 2009)[23]	37.58	8.53	-
RegSum* (Hong and Nenkova, 2014)[23]	38.57	9.75	-
GRU+GCN: Personalized Discourse Graph*[23]	38.23	9.48	-
MultiBERTSumm	**41.41**	**12.76**	**29.07**

Table 4.3: Performance Comparison on the DUC2004 dataset. ROUGE F1 is reported. Bold indicates the best result among all methods. Results with * are taken from the corresponding experiment results in papers.

Chapter 5

Conclusion and Future Research

In conclusion, through the research on related work, we found that there is still a lot of room for improvement in the performance of extractive summarization. Besides, there have been many research focused on single document summarization, while the area of multi-document summarization is still less explored. Motivated by this, we proposed a new multi-document summarization system using BERT based model.

Our MultiBERTSumm system uses a BERT based model as sentence encoder, and a 2-layer Transformer model as sentence extractor. Then a synthesis module is proposed to find the commonalities between each document summary. A final summary could be generated by selecting n sentences that have the highest score. In the experiment, our model achieves a competitive ROUGE score on the single document summarization. Besides, in the task of multi-document summarization, it achieves a competitive and stable ROUGE score. The result shows that the model could generate a coherent and comprehensive summary that contains important information in a document cluster.

There are still many works in the future to improve the performance of the system. We could focus on three aspects. First, through the observation of the example final summary of document clusters, there is still some redundant information in the sentences. Since we are doing extractive summarization and not modifying the original sentence, a compression module will be feasible to reduce redundancy. Specifically, we can train a model to remove clauses from sentences that contain irrelevant or unnecessary information. Another aspect is the experiment on more choices with the sentence encoder. Since our synthesis module is highly based on the result of single document summary, finding a better representation of a sentence is the key to the quality of the final summary. More experiments could be performed on BERT based model to find a better choice for sentence encoder. Third, we need to evaluate our multi-document summarization system on more datasets to examine its robustness.

Bibliography

[1] Nouf Altmami and Mohamed Menai. Automatic summarization of scientific articles: A survey. *Journal of King Saud University - Computer and Information Sciences*, 05 2020.

[2] Jianpeng Cheng and Mirella Lapata. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252*, 2016.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[4] Wafaa S El-Kassas, Cherif R Salama, Ahmed A Rafea, and Hoda K Mohamed. Automatic text summarization: A comprehensive survey. *Expert Systems with Applications*, 165:113679, 2021.

[5] Ahmed El-Refaiy, A.R. Abas, and I. Elhenawy. Review of recent techniques for extractive text summarization. *Journal of Theoretical and Applied Information Technology*, 96:7739–7759, 12 2018.

[6] Günes Erkan and Dragomir R Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479, 2004.

[7] Vishal Gupta and Gurpreet Lehal. A survey of text summarization extractive techniques. *Journal of Emerging Technologies in Web Intelligence*, 2, 08 2010.

[8] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701, 2015.

[9] Ruipeng Jia, Yanan Cao, Hengzhu Tang, Fang Fang, Cong Cao, and Shi Wang. Neural extractive summarization with hierarchical attentive heterogeneous graph network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3622–3631, Online, November 2020. Association for Computational Linguistics.

[10] Chris Kedzie, Kathleen McKeown, and Hal Daume III. Content selection in deep learning models of summarization. *arXiv preprint arXiv:1810.12343*, 2018.

[11] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[12] Yang Liu. Fine-tune bert for extractive summarization. *arXiv preprint arXiv:1903.10318*, 2019.

[13] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[14] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411, 2004.

[15] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents, 2016.

[16] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019.

[17] Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. *CoRR*, abs/1704.04368, 2017.

[18] Taner Uçkan and Ali Karcı. Extractive multi-document text summarization based on graph independent sets. *Egyptian Informatics Journal*, 21(3):145–157, 2020.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[20] Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. Discourse-aware neural extractive model for text summarization. *CoRR*, abs/1910.14142, 2019.

[21] Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. Discourse-aware neural extractive text summarization, 2019.

[22] Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. Graph-based neural multi-document summarization. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 452–462, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[23] Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. Graph-based neural multi-document summarization. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 452–462, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[24] Yong Zhang, Meng Joo Er, Rui Zhao, and Mahardhika Pratama. Multiview convolutional neural networks for multidocument extractive summarization. *IEEE Transactions on Cybernetics*, 47(10):3230–3242, 2017.

Appendix A

Code Repository

https://github.com/Joejoequ/MultiBERTSumm

Appendix B

Oracle Algorithm

The pseudocode is a revised version of Oracle algorithm in [10].

Algorithm 1: Oracle Gold Label Algorithm

Data: document sentences $s_1, s_2, ..., s_n$, human reference summary R, summary length n

Result: extract summary labels $y_1, ..., y_n$

`// extract summary labels indicate whether a sentence will be selected in summary`

$y_i := 0 \quad \forall i \in 1, \ldots, n$; `// Initialize extract labels to be 0.`

$S := []$; `// Initialize summary as empty list`

while $S.length < n$ **do**

  `/* Add the next best sentence to the summary if it will improve ROUGE score otherwise no improvement can be made so break */`

  $\hat{i} = argmax_{i \in \{1,...,n\}, y_i \neq 1} ROUGE\left(S + [s_i], R\right)$;

  if $ROUGE\left(S + [s_{\hat{i}}], R\right) > ROUGE(S, R)$ **then**

    `// Add` s^i `to the summary sentence list.`

    $S := S + [s_{\hat{i}}]$;

    `// Set the ^i-th extract label to indicate extraction.`

    $y_{\hat{i}} := 1$;

  else

    break; ;

  end

end

Appendix C

Training Parameter and Detailed Result

Training using a library TransformerSum that aims to make it easy to train, evaluate, and use machine learning Transformer models. Model is downloaded from Huggingface

C.1 BERT

Sentence Encoder = bert-base-uncased, batch size = 16 (limited to GPU RAM), optimizer=adamw, linear scheduler, gradient clipping value =1.0,warm up steps=1800, accumulate grad batches=2

Best found during training is epoch=2,step=6247.

'avg_test_acc_and_f1': 0.5773, 'rouge1-fmeasure': 0.4256,'rouge1-precision': 0.3810, 'rouge1-recall': 0.5223, 'rouge2-fmeasure': 0.1959, 'rouge2-precision': 0.1757,'rouge2-recall': 0.2405, 'rougeL-fmeasure': 0.2739, 'rougeL-precision': 0.2445, 'rougeL-recall': 0.3378, 'rougeLsum-fmeasure': 0.3896, 'rougeLsum-precision': 0.3490, 'rougeLsum-recall': 0.4776, 'test_acc': 0.9059, 'test_f1': 0.2486

C.2 RoBERTa

Sentence Encoder = roberta-base, batch size = 16 (limited to GPU RAM), optimizer=adamw, linear scheduler, gradient clipping value =1.0,warm up steps=1800, accumulate grad batches=2

Best found during training is epoch=3,step=26916

'avg_test_acc_and_f1': 0.5926, 'rouge1-fmeasure': 0.4332,'rouge1-precision': 0.3925, 'rouge1-recall': 0.5238, 'rouge2-fmeasure': 0.2043, 'rouge2-precision': 0.1858,'rouge2-recall': 0.2465, 'rougeL-fmeasure': 0.2766, 'rougeL-precision': 0.2501, 'rougeL-recall': 0.3360, 'rougeLsum-fmeasure': 0.3975, 'rougeLsum-precision': 0.3605, 'rougeLsum-recall': 0.4799, 'test_acc': 0.9091, 'test_f1': 0.2761

C.3 DistilBERT

Sentence Encoder = roberta-base, batch size = 32 (limited to GPU RAM), optimizer=adamw, linear scheduler, gradient clipping value =1.0,warm up steps=1800, accumulate grad batches=2

Best found during training is epoch=4,step=6247

'avg_test_acc_and_f1': 0.5499, 'rouge1-fmeasure': 0.4244,'rouge1-precision': 0.3806, 'rouge1-recall': 0.5199, 'rouge2-fmeasure': 0.1950, 'rouge2-precision': 0.1754,'rouge2-recall': 0.2388, 'rougeL-fmeasure': 0.2707, 'rougeL-precision': 0.2422, 'rougeL-recall': 0.3332, 'rougeLsum-fmeasure': 0.3885, 'rougeLsum-precision': 0.3487, 'rougeLsum-recall': 0.4754, 'test_acc': 0.9123, 'test_f1': 0.1875