SciSummPip: An Unsupervised Scientific Paper Summarization Pipeline

Jiaxin Ju[1], **Ming Liu** [2], **Longxiang Gao**[2], **and Shirui Pan**[1]
[1]Faculty of Information Technology, Monash University, Australia, VIC 3800
[2]School of Information Technology, Deakin University, Australia, VIC 3217
jjuu0002@student.monash.edu
{m.liu, longxiang.gao}@deakin.edu.au
shirui.pan@monash.edu

Abstract

The Scholarly Document Processing (SDP) workshop is to encourage more efforts on natural language understanding of scientific task. It contains three shared tasks and we participate in the LongSumm shared task. In this paper, we describe our text summarization system, SciSummPip, inspired by SummPip (Zhao et al., 2020) that is an unsupervised text summarization system for multi-document in news domain. Our SciSummPip includes a transformer-based language model SciBERT (Beltagy et al., 2019) for contextual sentence representation, content selection with PageRank (Page et al., 1999), sentence graph construction with both deep and linguistic information, sentence graph clustering and within-graph summary generation. Our work differs from previous method in that content selection and a summary length constraint is applied to adapt to the scientific domain. The experiment results on both training dataset and blind test dataset show the effectiveness of our method, and we empirically verify the robustness of modules used in SciSummPip with BERTScore (Zhang et al., 2019a).

1 Introduction

Text summarization aims at automatically generating a fluent and coherent summary that mainly contains the salient information from the source document(s). Two main categories are typically involved in the text summarization task, one is extractive approach (Luo et al., 2019; Xu and Durrett, 2019) which directly extracts salient sentences from the input text as the summary, and the other is abstractive approach (Sutskever et al., 2014; See et al., 2017; Sharma et al., 2019) which imitates human behaviour to produce new sentences based on the extracted information from the given document.

In order to meet the requirements of modern data-driven methods, several large datasets have been presented. The majority of those datasets are for generic domain, but few available corpora from other task-specific domains. Most of existing state-the-art summarization systems (Liu and Lapata, 2019; Zhou et al., 2020; Wang et al., 2020) target news or simple documents, and they are less adequate for summarizing scientific work due to the length and complexity. Those summarization systems cannot provide sufficient information conveyed in the scientific paper.

The general domain have been paid enough attention, whereas the attention in scientific domain is far from enough. To address this point, the Scholarly Document Processing (SDP) workshop (Chandrasekaran et al., 2020) is held to accelerate scientific discovery in research community, they appeal to researchers for designing a summarization system that can generate a relatively long summary for scientific work.

Since the release of Transformer (Vaswani et al., 2017) and BERT (Devlin et al., 2018), much research has been carried out on involving them in their system. Liu (2019) modified the input sequence embedding and built several summarization-specific layers for extractive summarization. Similarly, Liu and Lapata (2019) present a novel document-level encoder based on BERT (Devlin et al., 2018) for both extractive summarization and abstractive summarization. In their model structure, the lower transformer represents adjacent sentences and the higher layer with self-attention mechanism represents the multi-sentence discourse. These works leverage the advantage of deep neural network, not taking into account the linguistic information. In contrast, Zhao et al. (2020)[1] construct semantic clusters and sentence graphs for multi-document summarization, which involves linguistic information and discourse markers. In this paper,

[1]https://github.com/mingzi151/SummPip

Proceedings of the First Workshop on Scholarly Document Processing, pages 318–327

https://doi.org/10.18653/v1/P17

we followed the framework of Zhao et al. (2020) to construct our own unsupervised text summarization system. However, our model is different from the previous work: we modify the pipeline structure of multi-document summarization in the field of news to the single-document summarizer for summarizing scholarly documents, and we introduce two new steps to control the length of generated summary and to remove irrelevant sentences.

Our contributions in this work can be summarized in the following aspects:

- We highlight the importance of sentence embedding for scientific work. A variety of works focus on facilitating the process of obtaining sentence representation from a pretrained language model on generic domain, while less attention is paid on other task-specific domains.
- We compare the performances between PageRank (Page et al., 1999) and the Maximal Marginal Relevance (MMR) (Carbonell and Goldstein, 1998) in the content selection module. To our knowledge, no previous work compares their performances on scientific long document summarization task with deep neural representation.
- We experimentally verify that the effectiveness of the proposed model. We achieve better ROUGE results than original model on both training dataset and blind test dataset. Besides, our model is also evaluated on the BERTScore metric (Zhang et al., 2019a) and the results indicate that our model is more robust to generate high quality summary.

2 Related Work

Text Summarization System Most of recent text summarization systems leverage the advantages of deep neural networks, their encoder-decoder structures use either recurrent neural networks (Cheng and Lapata, 2016; Nallapati et al., 2016) or Transformer encoders (Zhang et al., 2019b; Khandelwal et al., 2019). Benefit of the sequence-to-sequence structure, a great progress in both extractive and abstractive document summarization is achieved. Though abstractive summarization has more potentials to generate interpretations in a human-like fashion, it has been found that sometimes repeatedly produces the same phrase or sentence (Suzuki and Nagata, 2016), which greatly reduces the comprehensibility and readability. In contrast, extractive summarization performs better in fluency aspect and it can grammatical and accurately represent the source text. One potential issue in extractive summarization is that not all of information from the extracted sentence is important, which leads more redundancy in the generated summary.

In the work of Zhao et al. (2020), they apply graph structure and consider the discourse relationship between sentences rather than using encoder-decoder structure, and text compression is implemented in the final stage to reduce the redundancy in the generated sentences. However, their model is designed for multi-document summarization in the news domain, we extend their SummPip to single-document settings for scientific long articles.

Sentence Embedding Method Term frequency–inverse document frequency (TF-IDF) is widely used in traditional NLP, but it cannot capture the semantic information and contextual relationship between sentences. Word2Vec (Mikolov et al., 2013) is used in SummPip (Zhao et al., 2020) to capture contextualized relationship, but this embedding method cannot solve the polysemous problem. More recently, BERT (Devlin et al., 2018) has achieved better performance in many NLP downstream tasks, but it is difficult to derive sentence embeddings. To solve this limitation, single sentences are passed to the BERT and two common ways to extract sentence representation are widely used: averaging the outputs and using the output of the [CLS] token (May et al., 2019; Zhang et al., 2019a).

Xiao (2018) develops a repository, bert-as-a-service[2], which accelerates the process of extracting token and sentence embeddings from BERT (Devlin et al., 2018). Lately, in order to find a better way to derive semantically similar sentence from language models, Reimers and Gurevych (2019) present SBERT. However, above works help facilitate workload in generic domain rather than task-specific domain.

Content Selection Graph is an intuitive structure for utilizing the relation information between sentences. Some work (Mihalcea and Tarau, 2004; Erkan and Radev, 2004) focuses on selecting salient sentences by leveraging graph-based rank-

[2]https://github.com/hanxiao/bert-as-service/

Characteristics	Extractive		Abstractive		Test dataset
	Sci_P	Ref_S	Sci_P	Ref_S	Sci_P
Range of corpus size (sentences)	[37,629]	[9,48]	[24,792]	[1,87]	[119,345]
Median value of corpus size (sentences)	186	31	201	31	219
Range of sentence length (words)	[12,51]	[15,48]	[10,44]	[0.5,54]	[18,27]
Median value of sentence length (words)	26	27	26	21	22

Table 1: Elementary data statistics for the LongSumm shared task of the Scholarly Document Processing @ EMNLP 2020. Sci_P and Ref_S represent scientific paper and reference summary, respectively.

ing methods. Inspired by PageRank algorithm (Page et al., 1999), they consider the document as a graph where sentences are vertices and edges represent the relations between two sentences. Shortly thereafter, some researchers (Carbonell and Goldstein, 1998; Kurmi and Jain, 2014; Mao et al., 2020) involved a query-biased strategy, the Maximal Marginal Relevance (MMR) (Carbonell and Goldstein, 1998), in their summarizers. MMR tries to balance the relevance and diversity by controlling the trade-off parameter λ. The first part of the formula controls query relevance and the second part controls diversity.

$$MMR = \underset{S_i \in \mathcal{C}}{\operatorname{argmax}} \lambda Sim_1(S_i, Q) - (1-\lambda) \underset{S_j \in \mathcal{S}}{\operatorname{argmax}} Sim_2(S_i, S_j)$$

Where C is the set of candidate sentences, S is the set of extracted sentences, Q is the query embedding, S_i, S_j are sentence embeddings of candidate sentences i and j, respectively. Sim indicates the cosine similarity between two embeddings.

Though this approach have been proved that it outperforms generic summarization approaches in the information retrieval task, to our knowledge, there is no previous work compared it with PageRank algorithm on scientific long document summarization task. Our work incorporates deep neural representations into both PageRank algorithm and MMR strategy and shows the comparison between these two methods in the field of scientific work for both extractive and abstractive summarization.

3 Dataset Pre-processing

The training dataset provided by the LongSumm shared task consists of 2236 scientific papers, of which 1705 are for extractive method and 531 are for abstractive method. The reference extractive summaries are generated by TalkSumm (Lev et al., 2019) that extracts sentences appeared in associated conference videos, while the abstractive summaries are collected from blogs written by researchers.

Download paper We download the training corpus from the given URLs (for abstractive) and the script (for extractive).

Paper Parsing All of papers are parsed from PDF form into JSON structure by using Science-Parse[3]. It outputs a JSON file for each PDF, which contains the title, abstract text, metadata, and the text of each section in the paper.

Text processing We concatenate each section text as the paper text. Then sentences are segmented by using the NLTK library, and each sentence is tokenized as well. Table 1 reports the result of the statistics analysis for both training dataset and test dataset, and we can see that the number of sentences in some reference summaries is far less than required length of generated summary, 600 words, which may lead a bias in the evaluation.

4 System overview

We adopt the SummPip (Zhao et al., 2020) as our baseline model, and we modify the pipeline architecture for summarizing scholarly documents. Two new steps are introduce for adapting scientific domain, one is to remove irrelevant sentences and the other is to control the length of generated summary. In the following subsections, we will specify each component in the SciSummPip.

4.1 Embedding Method

Pretrained language model In this paper, we apply a publicly available large-scale language model, SciBERT (Beltagy et al., 2019), which is pretrained based on BERT (Devlin et al., 2018) and extends the idea of word embeddings by learning

[3]https://github.com/allenai/science-parse

contextual representations from large-scale scientific corpora. This is implemented in Pytorch using Transformers established by Wolf et al. (2019)[4].

Sentence embedding Using more accurate sentence embeddings can improve the performance of summarization system in language understanding. In SciSummPip, we average the output of SciBERT from the second layer to the last layer. In addition, we also experiment with other embedding methods and the the results show that this is a more accurate way to represent scientific sentences.

4.2 Sentence Graph Construction

Content selection Not all of sentences should be involved in the summary, so we include content selection step before constructing sentence graph. We build a matrix to store the similarity between each two sentences, then PageRank (Page et al., 1999) algorithm is implemented to rank all of sentences. Sentences with lower score will be deleted from the candidate list, here we introduce a new step to control the ratio of removed sentences.

Graph construction We construct the sentence graph, where each node represents a sentence, and nodes are connected if they meet the linguistic requirements. To identify this structure, we borrow the components from the previous work (Zhao et al., 2020). Specifically, this pipeline consists of discovering deverbal noun reference, finding the same entity continuation, recognizing discourse markers, and calculating sentence similarity by taking the cosine similarity.

4.3 Text Generation

Spectral clustering After identifying pairwise sentence connection, we involve a new step for determining the number of clusters. This is to control the length of generated summary so that the summary varies with the length of the original paper.

Multi-sentences compression This module (Boudin and Morin, 2013) is to generate a single summary sentence from each sentence cluster. Sentences with similar semantic information will be compressed by building a word graph. Considering the key phrases and discourse structure, so that the reconstructed sentence will have higher score. Select the sentence with the highest score as the summary sentence, and then combine all reconstructed summary sentences as the generated summary.

5 Experiment Setup

5.1 Implementation Details

Extractive summarization Task We use SciBERT for sentence embedding in our pipeline, so for extractive text summarization task we directly use Scibert-summarizer[5] with the fixed length range (from 60 to 600 words).

Abstractive summarization Task We implement our pipeline, SciSummPip, in abstractive summarization task, and we compare the performances of PageRank algorithm and of MMR strategy in the content selection module. For PageRank algorithm, we set a cutoff ratio that is a new introduced parameter for removing irrelevant sentences and the empirical results show that setting it as 0.25 achieves better performance. For the MMR strategy, we set 0.2, 0.5, 0.8 for the trade-off parameter in the experiment, respectively. To control the generated summary length, we introduce another new parameter, extended ratio, to modify the number of clusters based on the number of ranking sentences. In our pipeline,we set it as 0.3.

5.2 Comparison Systems

For extractive task, we compare our model with the following unsupervised summarization models:

TextRank (Barrios et al., 2016) TextRank (Mihalcea and Tarau, 2004) applies a variation of PageRank algorithm (Page et al., 1999) over a graph-based structure, and it produces a list of ranked elements in the graph without the need of a training corpus. TextRank implemented in this paper is produced by Barrios et al. (2016), they change the similarity function to Okapi BM25 so that the performance is better than the original textRank model. We set the output summary with the fixed length 600 words.

LexRank (Erkan and Radev, 2004) Similar with textRank (Mihalcea and Tarau, 2004), LexRank also applies PageRank algorithm and leverages a graph structure for summarization. Differently, textRank calculate the similarity based on the number of words two sentences have in common, while LexRank uses cosine similarity of TF-IDF vectors.

[4]https://github.com/huggingface/transformers

[5]bert-extractive-summarizer: https://pypi.org/project/bert-extractive-summarizer/

	R1_F	R1_R	R2_F	R2_R	RL_F	RL_R
Extractive dataset						
Scibert-summarizer	**58.13**	**57.53**	**27.20**	**26.82**	**28.65**	**28.29**
TextRank	57.42	57.31	26.38	26.48	28.48	27.59
LextRank	46.23	36.38	20.71	16.33	21.34	16.76
MMR_{Sci}(λ=0.5)	55.24	55.48	23.74	23.85	21.00	21.11
Abstractive dataset						
SummPip_{+PR}	36.17	32.73	8.36	7.27	14.80	13.83
SciSummPip_{PR}	**40.90**	**43.09**	**9.52**	**9.83**	**15.47**	**17.26**
$\text{SciSummPip}_{MMR0\cdot2}$	32.34	28.31	6.54	5.48	13.60	12.37
$\text{SciSummPip}_{MMR0\cdot5}$	30.69	25.63	6.64	5.32	13.37	11.56
$\text{SciSummPip}_{MMR0\cdot8}$	33.06	27.88	7.58	6.17	14.18	12.39
Blind Test Dataset						
Scibert-summarizer	**49.16**	**49.35**	**12.80**	**12.76**	**18.31**	**18.33**
SciSummPip	47.37	40.89	13.35	11.40	17.54	15.02
SummPip	38.62	30.16	9.01	6.95	15.15	11.74

Table 2: ROUGE scores reported on the training dataset and the blind test dataset. Best results are in **boldface**. The reference extractive summary and abstractive summary are generated by TalkSumm (Lev et al., 2019) and collected from online blogs, respectively. MMR_{Sci} indicates we implement MMR algorithm with sentence embeddings derived from SciBERT(Beltagy et al., 2019). $SciSummPip_{PR}$ and $SciSummPip_{MMR}$ are our model with different content selection modules, and the number follow the MMR is the setting for trade-off parameter λ. As SummPip cannot effectively run on large scale corpora of long document, we add content selection module and shown as SummPip_{+PR}.

MMR (Carbonell and Goldstein, 1998) MMR is a query-biased summarization approach, it tries to balance the relevance and diversity by controlling the trade-off parameter λ. In the previous works, the similarity usually calculate based on TF-IDF, but in our implementation we use sentence embeddings derived from the output of SciBERT (Beltagy et al., 2019). In addition, we set the document title as the query and the fixed length of generated summary is set as 600 words.

For abstractive task, we apply different sentence embedding methods in SciSummPip:

- SciBERT (Beltagy et al., 2019): We implement two common strategies for sentence embeddings derived from SciBERT model: averaging the output from the second to the last layer and using [CLS] token embedding.
- SummPip (Zhao et al., 2020): We use the same embedding method with the original pipeline to compare the performance.
- SBERT (Reimers and Gurevych, 2019): This is a modification of the BERT network using siamese and triplet networks in order to find semantically similar sentences in vector space. Their empirical results indicate that their method is better than those two common embedding strategies, so we incorporate it into SciSummPip as a comparison.

6 Evaluation and Results

6.1 Experiment result on training dataset

Extractive summaries The training dataset for extractive method consists of 1705 papers, of which one paper cannot be parsed. Thus, we evaluate 1704 papers with the ROUGE metric(Lin and Hovy, 2003) in our experiments.

As displayed in Table 2, the Scibert-summarizer achieves better ROUGE scores than all other compared systems. We implement MMR algorithm with sentence embedding derived from averaging SciBERT (Beltagy et al., 2019) output, and we can see it performs better than LexRank (Erkan and Radev, 2004) but worse than the textRank model (Barrios et al., 2016) with the Okapi BM25 similarity function. Therefore, we can verify that PageRank ranking algorithm performers better than MMR strategy in extractive task.

Abstractive summaries For abstractive experiments, we collect 530 summaries in total as one paper cannot be parsed by Science-parse.

Sentence Embedding	R1_F	R2_F	RL_F
Avg. SciBERT embeddings	40.90	9.52	15.47
Special token embedding	39.27	8.81	15.09
Word2Vec	36.17	8.36	14.80
SBERT	39.75	9.41	15.27

Table 3: ROUGE F1 scores for SciSummPip with different sentence embedding methods. Special token embedding method is extracting [CLS] token embedding from SciBERT (Beltagy et al., 2019) output.

Sentence Embedding	R1_R	R2_R	RL_R
Avg. SciBERT embeddings	43.09	9.83	17.26
Special token embedding	39.99	8.75	16.13
Word2Vec	32.73	7.27	13.83
SBERT	41.53	9.56	16.73

Table 4: ROUGE Recall results for SciSummPip with different sentence embedding methods.

We implement SciSummPip with different parameter settings to find out the best one. The number of words in each sentence is set from 15 to 29, then we observe that the summary with 26 words in each sentence achieves the best performance. We incorporate PageRank algorithm (Page et al., 1999) and MMR algorithm (Carbonell and Goldstein, 1998) into SciSummPip content selection module, respectively. As displayed in Table 2, it is not surprising to see SciSummPip with PageRank algorithm outperforms all of settings for SciSummPip with MMR algorithm, because the performance of textRank is better than that of MMR in the extractive task.

6.2 Experiment result on test dataset

The blind test dataset consists of 22 scientific papers[6]. It does not declare the blind test data is for extractive summarizer or abstractive summarizer, so we implement both Scibert-summarizer and SciSummPip on it. Comparing with the SummPip (Zhao et al., 2020), the experiment results verify that our new pipeline architecture significantly improve the performance. In addition, we try different number of words generated in each sentence and we find that setting it closes to the median value of that in scientific papers would gain higher score. Besides, although extractive model gains the highest ROUGE score, we still can see our SciSummPip is competitive.

[6]Test dataset: https://github.com/guyfe/LongSumm

	Precision	Recall	F1-Score
SciSummPip	**0.807**	0.800	**0.815**
SciSummPip$_{MMR}$	0.806	0.810	0.808
SummPip$_{+PR}$	0.794	0.813	0.806
SBERT	0.795	**0.814**	0.804

Table 5: BERTScore reported on abstractive training dataset to investigate text generation ability of our model. SBERT means we use use SBERT sentence embedding method in SciSummPip.

6.3 Different Sentence Embedding Methods

To find out a more accurate method for representing scientific sentences, we incorporate different embedding strategies into SciSummPip. Performances reported in Table 3 and Table 4 indicate that our model ranks highest with averaging the output of SciBERT (Beltagy et al., 2019) method. SBERT (Reimers and Gurevych, 2019) shows competitive performance even though it is designed for generic domain. In fact, utilizing SBERT significantly reduce the workload of extracting sentence embedding, but it is not sufficient enough for representing scientific sentence.

6.4 BERTScore Evaluation

We evaluate models on BERTScore (Zhang et al., 2019a), an automatic evaluation metric for text generation, to investigate the ability of writing abstractive summary. BERTScore calculates a similarity score for each token in the candidate sentence with each token in the reference sentence by leveraging contextual embeddings. As can be seen in Table 5, SciSummPip achieves highest precision and F1-score while SBERT gains the highest recall. This proves that the summary generated by our model is more informative and representative. Since BERTScore utilizes Bert (Devlin et al., 2018) to calculate similarity score, the max length of input sequence is 512 tokens, which limits the performance of relatively long summary.

We further investigate the distribution of F1-score from BERTScore evaluation. As shown in figure 1, although these models achieve similar performance, the F1-score distribution of SciSummPip obviously more stable than others. SciSummPip achieve the highest frequency in the range of 0.80-0.82, which means near 140 generated summaries gain around 0.81 F1-score. Therefore, we can say that our model is more robust for summarizing scientific work in abstractive task.

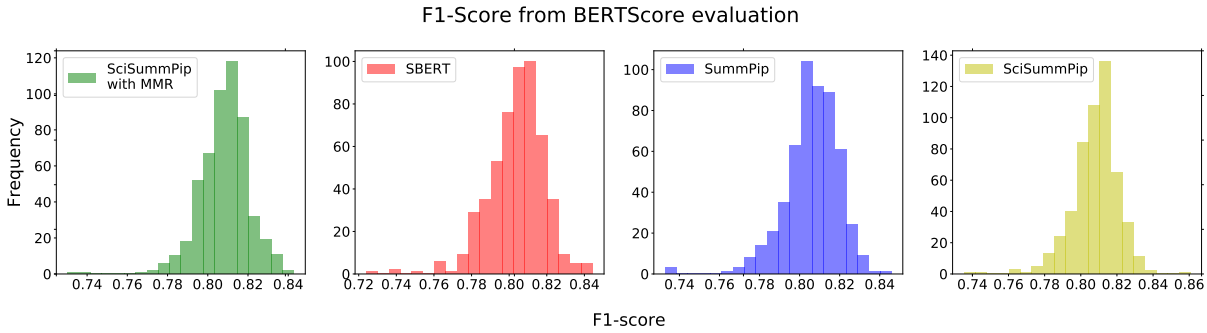

Figure 1: The histogram distribution of F1-score evaluated by BERTScore metric for each model reported in Table 5. X-axis indicates data range of F1-score and Y-axis indicates the frequency of the data in each bin. In order to ensure the bin data range for each distribution is same, we set the data range of each bin as 0.005 so that the parameter, bins, is set as $int(data\ range\ of\ F1-score/0.005)$.

Extractive Reference Summary:
The analysis of emotions in texts is an important task in NLP. Traditional studies treat this task as a pipeline of two separated sub-tasks: emotion classification and emotion cause detection. The former identifies the category of an emotion and the latter detects the cause of an emotion. This separated framework makes each sub-task more flexible to deal with, but it neglects the relevance between the two sub-tasks. In this paper, we use the human-labeled emotion corpus provided by Cheng et al. (2017) as our experimental data (namely Cheng emotion corpus). Cheng emotion corpus can be considered as a collection of subtweets. For each emotion in a subtweet, all emotion keywords expressing the emotion are selected, and then the class and the cause of the emotion are annotated. (...)

Scibert-summarizer:
The analysis of emotions in texts is an important task in NLP. Cheng emotion corpus can be considered as a collection of subtweets. Given an instance which is a pair of <an emotion keyword, a clause in the subtweet>, ECause assigns a binary label to the instance to indicates the presence of a causal relation. The input text of an ECause instance also has three sequences of words: the emotion keyword (i.e. EmoKW), the current clause (i.e. CauseCL) and the context between EmoKW and CauseCL. The BiLSTM layer focuses on the extraction of sequence features, and the attention layer focuses on the learning of word importance (weights). (...)

Table 6: Example of the generated extractive summary compared with reference summary that is generated by TalkSumm (Lev et al., 2019). Text in the same color indicates the content they describe is the same. Due to the length constraint, we omit part of the generated summary and shown as (...).

Abstractive Reference Summary:
The paper proposes a two-stage synthesis network that can perform transfer learning for the task of machine comprehension. The problem is the following: We have a domain DS for which we have labelled dataset of question-answer pairs and another domain DT for which we do not have any labelled dataset. We use the data for domain DS to train SynNet and use that to generate synthetic question-answer pairs for domain DT. Now we can train a machine comprehension model M on DS and finetune using the synthetic data for DT. SynNet Works in two stages: Answer Synthesis - Given a text paragraph, generate an answer. (...) After the word vector, append a '1' if the word was part of the candidate answer else append a '0'. Feed to a Bi-LSTM network (encoder-decoder) where the decoder conditions on the representation generated by the encoder as well as the question tokens generated so far. (...)

SciSummPip:
the ability to quickly use a mc model trained on one domain to answer questions over paragraphs from another with no annotated data. recent work generated synthetic data generated questions leads to improved performance, we use a model where the answer synthesis and question types. we generate the answer first because answers are usually key semantic concepts, while questions can transfer a mc model trained on another domain. when we ensemble a bidaf model fs we use the two-stage synnet to generate data tuples to directly boost performance boost. (...) however, unlike machine translation , for tasks like mc, we need to synthesize both the question and answers given the context paragraph. (...) the first stage of the model, an answer synthesis module , uses a Bi-directional LSTM to predict iob tags on the input paragraph, which mark out key semantic concepts that are likely answers.(...)

Table 7: Example of the generated abstractive summary compared with reference summary that is collected from researcher's blog. Text in the same color indicates the content they describe is similar. Due to the length constraint, we omit part of the generated summary and shown as (...).

6.5 Human Analysis

We further manually inspect the generated summary to explore if our model can capture the salient information from given document. Table 6 and Table 7 display an example of generated summary compared with the corresponding reference summary in the training dataset. The abstractive reference summary is collected from the online blog written by the researcher, so it is more difficult to capture the similar description in the generated

summary. However, As shown in table 7, our model successfully write some similar context in the final output. Notwithstanding, we have to say the readability and grammatically of the generated summary still need to be improved.

For blind test dataset, we also inspect the extractive summary and abstractive summary for the same paper. We find that the Scibert-summarizer tends to extract the sentence appeared in the early part of the paper, and the generated summary usually lack of logicality and consistency. In contrast, the summary produced by SciSummPip is more logical and contains more salient information about the methodology and the experiment. Although Scibert-summarizer gains higher ROUGE score on the blind test dataset, the summary generated by our model is more consistent with the purpose of the LongSumm Shared Task.

7 Conclusion and Limitation

In this paper, we have presented the modified unsupervised pipeline architecture, SciSummPip, that leverages a transformer-based language model for summarizing scientific papers. We add content selection module and two steps to remove irrelevant sentences and to control the length of generated summary. After that, the linguistic knowledge will be incorporated into the process of multi-sentences compression for summarizing scientific work. The experiment results of automatic evaluation prove that our new pipeline significantly improves the overall performance on both training and blind test dataset. Besides, through manual inspection we find that our model indeed capture the salient information from the given source document. However, we have to admit that the readability of generated summary needs to be improved.

We incorporated deep neural representation into both MMR (Carbonell and Goldstein, 1998) strategy and PageRank (Page et al., 1999) algorithm. Even though MMR strategy performs better in information retrieval task, we empirically verified that it is not sufficient for our model to summarize scientific work. MMR is a query-biased approach and we chose the title as query in our implementation, thus the potential reason for worse performance may be the query we chose is not effective enough.

To investigate a sentence embedding method for sufficiently summarizing scholarly document, we compared the performances among several embedding strategies and we also evaluated their performances on both ROUGE metric and BERTScore metric. Although averaging the output of SciBERT (Beltagy et al., 2019) achieves better performance, the workload of using it to extract sentence embeddings is heavier than that of directly using SBERT (Reimers and Gurevych, 2019). There is enough work for generic domain while the attention paid for task-specific domain is far from enough, therefore we appeal to researchers for making more efforts on task-specific domain in their further research.

8 Future work

As the future, we will evaluate our pipeline on larger scientific datasets to show the effectiveness and robustness, and we also would like to conduct a analysis on the faithfulness and the level of abstraction for the generated summary.

Acknowledgments

We would like to thank the anonymous reviewer(s) for helpful comments and suggestions.

References

Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. 2016. Variations of the similarity function of textrank for automated summarization. *arXiv preprint arXiv:1602.03606*.

Iz Beltagy, Arman Cohan, and Kyle Lo. 2019. Scibert: Pretrained contextualized embeddings for scientific text. *arXiv preprint arXiv:1903.10676*.

Florian Boudin and Emmanuel Morin. 2013. Keyphrase extraction for n-best reranking in multi-sentence compression.

Jaime Carbonell and Jade Goldstein. 1998. 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*, pages 335–336.

Maroli Krishnayya Chandrasekaran, Guy Feigenblat, Hovy. Eduard, Anirudh Ravichander, Michal. Shmueli-Scheuer, and Anita De Waard. 2020. Overview and insights from scientific document summarization shared tasks 2020: Cl-scisumm, laysumm and longsumm. In *In Proceedings of the First Workshop on Scholarly Document Processing (SDP 2020)*.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.

Urvashi Khandelwal, Kevin Clark, Dan Jurafsky, and Lukasz Kaiser. 2019. Sample efficient text summarization using a single pre-trained transformer. *arXiv preprint arXiv:1905.08836*.

Rashmi Kurmi and Pranita Jain. 2014. Text summarization using enhanced mmr technique. In *2014 International Conference on Computer Communication and Informatics*, pages 1–5. IEEE.

Guy Lev, Michal Shmueli-Scheuer, Jonathan Herzig, Achiya Jerbi, and David Konopnicki. 2019. Talksumm: A dataset and scalable annotation method for scientific paper summarization based on conference talks. *arXiv preprint arXiv:1906.01351*.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 150–157.

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

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345*.

Ling Luo, Xiang Ao, Yan Song, Feiyang Pan, Min Yang, and Qing He. 2019. Reading like her: Human reading inspired extractive summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3024–3034.

Yuning Mao, Yanru Qu, Yiqing Xie, Xiang Ren, and Jiawei Han. 2020. Multi-document summarization with maximal marginal relevance-guided reinforcement learning. *arXiv preprint arXiv:2010.00117*.

Chandler May, Alex Wang, Shikha Bordia, Samuel R Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders. *arXiv preprint arXiv:1903.10561*.

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

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Eva Sharma, Luyang Huang, Zhe Hu, and Lu Wang. 2019. An entity-driven framework for abstractive summarization. *arXiv preprint arXiv:1909.02059*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Jun Suzuki and Masaaki Nagata. 2016. Cutting-off redundant repeating generations for neural abstractive summarization. *arXiv preprint arXiv:1701.00138*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuanjing Huang. 2020. Heterogeneous graph neural networks for extractive document summarization. *arXiv preprint arXiv:2004.12393*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Han Xiao. 2018. bert-as-service. https://github.com/hanxiao/bert-as-service.

Jiacheng Xu and Greg Durrett. 2019. Neural extractive text summarization with syntactic compression. *arXiv preprint arXiv:1902.00863*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019a. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019b. Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization. *arXiv preprint arXiv:1905.06566*.

Jinming Zhao, Ming Liu, Longxiang Gao, Yuan Jin, Lan Du, He Zhao, He Zhang, and Gholamreza Haffari. 2020. Summpip: Unsupervised multi-document summarization with sentence graph compression. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1949–1952.

Qingyu Zhou, Furu Wei, and Ming Zhou. 2020. At which level should we extract? an empirical study on extractive document summarization. *arXiv preprint arXiv:2004.02664*.