SIMSUM: Document-level Text Simplification via Simultaneous Summarization

Natural language processing (CS5808) course Project

Akshay kumar
Computer Science and Engineering
IIT Hyderabad
cs23mtech11022@iith.ac.in

Arnab Ghosh
Computer Science and Engineering
IIT Hyderabad
cs23mtech11025@iith.ac.in

Sanket Rathod
Computer Science and Engineering
IIT Hyderabad
cs23mtech11033@iith.ac.in

ABSTRACT

Document level text simplification is one of the types of test simplification where the whole document is simplified to comprehend easily, while retaining the content. Here, we have reimplemented the original two stage SIMSUM model with certain modification in loss function and keyword prompting technique to enhance the simplicity and improve the performance of the existing model.

Here the same two existing benchmark datasets for text simplification task are used, namely D-Wikipedia and Wiki-Doc. We have evaluated the performance using SARI, DSARI and FGGL score and found a better result as compared to the existing model results.

1 Introduction

Text simplification is a technique which focuses on making documents easier to understand.

In text simplification there are two main approaches: Sentence simplification and Document simplification. Sentence simplification involves simplifying individual sentences within the text which keeps the original and simplified sentence length the same. The document simplification aims to shrink the overall content by reducing the number of sentences in the output. Here we are concentrated in Document simplification.

Paper structure: In section 2 we presented the method we have used. Then in section 3 we presented experiments and the dataset description which we have used and the evaluation matrices. In section 4 Our findings and future works are presented.

2 Method

We have used the same architecture of SIMSUM model as given in the original paper. It consists of two-level architecture having summarizer in the first level and simplifier in the second level. The first level summarizes the document and output from it passed to the second level where the summarized text is simplified line by line like sentence level text simplification. Figure 1[1] shows the architecture of the SIMSUM model.

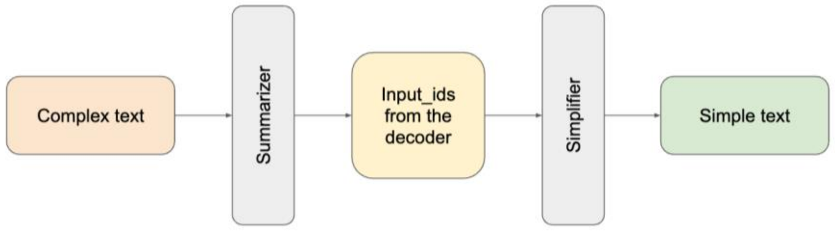

Figure 1: The workflow of the proposed framework. It contains two stages: the first one is *Summarizer*, and the second one is *Simplifier*. The generated output of the *Summarizer* is fed to the *Simplifier* without tokenizer's decoding since it does not allow the gradients to flow back to the *Summarizer* during the training stage.

The model is trained end to end considering both levels as a unit.

2.1 Keyword prompt

Inspired from the original paper which introduced method to introduce important keywords of the document to be extracted and feed the model with the document and important keyword to force model to look for those words and relevant information while summarizing the document. It helps the model to get a better insight of the document and retrieve all the important information. The original paper used KeyBERT method to generate important. keywords and they embed those keywords in input before feeding it to the document.

As the model is enforced by the keywords to generate summary and retrieve information, we can improve models' performance by providing best possible keyword as input. Hence, we tried to use TextRank which is a graph-based algorithm that analyzes the co-occurrence patterns of words in the input to identify important keywords based on their semantic relationships and importance. Then, we embed those keywords along with the document to provide input to the model.

2.2 Embedding similarity and loss function

The original SIMSUM models use standard loss function cross-entropy with an addition term L_{CosSim} that measures the cosine similarity of generated text with the original text. This new loss function helps the model to increase similarity between generated text and the reference text. What we thought of is a part of our problem is not considered in the above loss function. The simplicity of the generated text is not measure used as loss function to generate simple texts.

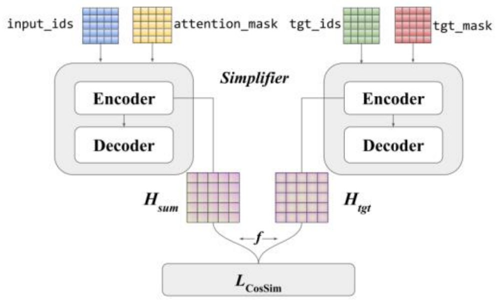

Figure 2: The embedding similarity computation process. `tgt_ids` and `tgt_mask` are obtained by tokenizing the targets. From both **Encoder**s the last hidden state H_{sum} and H_{tgt} are obtained. Then, after transformation f the cosine similarity is computed.

We introduced a new term to the loss function $L_{simplicity}$ that measures the simplicity of the generated text and forces our model to generate simple text along with similarity of generated text.

$$L = L1 + \lambda_1 \cdot L_{CosSim} + \lambda_2 \cdot L_{simplicity}$$

Here, λ_1 and λ_2 are hyperparameters that helps to adjust the degree of addition terms in the loss function. The original paper has an idea of finding the similarity between the generated text and reference text is shown in Figure 2[1] that find the cosine similarity between final output and target embedding.

$$L_{CosSim} = -\mathrm{CosSim}(f(H_{sum}), f(H_{tgt}))$$
$$f(H) = \mathrm{ReLU}\ (HW)$$

We then tried using the FKGL score to measure the simplicity of generated text. The simple document will have a lower FKFL score. We calculated $L_{simplicity}$ term by inversing the FKGL score. The epsilon terms here help us to handle zero FKGL score in the given below expression of $L_{simplicity}$.

$$L_{simplicity} = - 1 / (\mathrm{FKGLscore} + \epsilon)$$

3 Experiments

3.1 Data

The dataset used for our experiment are D-Wikipedia and wiki-Doc that are pre-processed data given in the original paper. Both the datasets are divided into simple and complex. The complete statistic of the dataset is provided in Table 1 and Table 2.

D-Wikipedia:

	Simple	Complex
Total Sentences	349,561	546,744
Total words	703,550	17,740,142
Avg. sentence/article	3.33	5.20
Avg. words present	20.24	32.45

Table1

Wiki-Doc:

	Simple	Complex
Total Sentences	55,885	258,303
Total words	906,988	5,927,616
Avg. sentence/article	3.20	14.81
Avg. words present	16.23	22.95

Table 2

3.2 Baseline

We are evaluating our idea along the baseline models in simplification, summarization tasks and with the results of the original SIMSUM mode. Given below the details of each:

T5: It is known as Text-to-Text Transfer Transformer which is an encoder-decoder model used for language tasks.

BART: It is powerful pretrained mode on huge dataset. It is mainly used for sequence-to-sequence tasks like text simplification.

SIMSUM(T5): SIMSUM model taking T5 as backbone in both summarizer and simplifier.

SIMSUM(BART): SIMSUM model taking BART as backbone in both summarizer and simplifier.

3.3 Evaluation metrices

We are using the given below standard metric to evaluate the performance of our model:

SARI: It is a popular sentence simplification task metric that compares the reference text with generated and find how good is generated text by looking at various metric like important words added, deleted etc. The higher the score means the more similar.

D-SARI: It is a modification of SARI specifically for document level simplification tasks that works similar work as SARI and has an addition parameter for length of generated text as compared to original document. The higher the core means the more document level simplification.

FKGL: It is a score which measures how simple is the text while reading and understanding without liking for grammar and meaning. The higher the score means less simple.

3.4 Results

We have obtained a better result as compared to the results obtained in the original paper. The summary of the results is shown in Table 3

	D-Wikipedia			Wiki-Doc		
Model	**SARI**	**D-SARI**	**FKGL**	**SARI**	**D-SARI**	**FKGL**
SimSum(T5)	48.34	**39.83**	5.76	51.31	41.25	**6.37**
SimSum(BART)	**49.16**	38.26	**5.61**	**51.78**	**42.17**	6.53

Table 3

4 Conclusion:

4.1 Main findings

This paper introduces changes to the existing SIMSUM model to improve its performance and give better results. We have introduced an updated loss function to measure the simplicity of the generated summary and penalize it accordingly. Hence, now we have a better loss function that takes care of simplicity along with semantics, grammar and content that is taken care of on original loss function given in the original paper.

Also, we have tried to improve the performance of the model by using advanced keyword generation technique 'TextRank' to extract better keywords and prompt it as input to the model.

4.2 Future works

In future we are planning to use transformer-based models such as GPT and Transformer-XL for better summarization and simplification performance. Also, we would like to explore domain specific document summarization with domain specific keyword extraction methods. We are planning to look for some new evaluation metric that better capture the quality of simplification in terms of readability, coherence, and information retention could help in fine-tuning the model more effectively.

REFERENCES

[1] Blinova, Zhou, Jaggi, Eickhoff, & Bahrainian, (2023). SIMSUM: Document-level Text Simplification via Simultaneous Summarization. Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics, Volume 1, pages 9927–9944. URL: https://aclanthology.org/2023.acl-long.552

[2] Yusen Zhang, Ansong Ni, Ziming Mao, Chen Henry Wu, Chenguang Zhu, Budhaditya Deb, Ahmed Awadallah, Dragomir Radev, and Rui Zhang. 2022. Summn: A multi-stage summarization framework for long input dialogues and documents. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, Volume 1, pages 1592–1604, Association for Computational Linguistics.