SIMSUM: Document-level Text Simplification via Simultaneous Summarization

Sofia Blinova[*]
EPFL
sofia.blinova@epfl.ch

Xinyu Zhou[*]
EPFL
xinyu.zhou@epfl.ch

Martin Jaggi
EPFL
martin.jaggi@epfl.ch

Carsten Eickhoff
University of Tübingen
carsten.eickhoff@uni-tuebingen.de

Seyed Ali Bahrainian
EPFL, University of Tübingen
seyed-ali.bahreinian@uni-tuebingen.de

Abstract

Document-level text simplification is a specific type of simplification which involves simplifying documents consisting of several sentences by rewriting them into fewer or more sentences. In this paper, we propose a new two-stage framework SIMSUM for automated document-level text simplification. Our model is designed with explicit summarization and simplification models and guides the generation using the main keywords of a source text. In order to evaluate our new model, we use two existing benchmark datasets for simplification, namely D-Wikipedia and Wiki-Doc. We compare our model's performance with state of the art and show that SIMSUM achieves top results on the D-Wikipedia dataset SARI (+1.20), D-SARI (+1.64), and FKGL (-0.35) scores, improving over the best baseline models. In order to evaluate the quality of the generated text, we analyze the outputs from different models qualitatively and demonstrate the merit of our new model. Our code and datasets are available [1].

1 Introduction

Text simplification is an important technique that aims to simplify a document to make it more understandable and accessible for people at different education and reading levels while still retaining the content of the original text (Woodsend and Lapata, 2011). It concentrates on lexical simplification (i.e., using simpler vocabulary and including definitions that provide explanations in simple terms) as well as syntactic simplification (i.e., using less complicated sentence structures and grammar) (Saggion, 2017). Simplification is considered as a sequence-to-sequence text generation problem, closely resembling other NLP generation tasks such as text summarization (Dong et al., 2018; Cao et al., 2020; Miller, 2019) and paraphrasing (Zhao et al., 2018).

The applications of text simplification are broad. It can be an important tool for assisting children (Kajiwara et al., 2013) and non-native speakers (Glavaš and Štajner, 2015; Paetzold, 2016) to understand advanced texts with ease. Additionally, it is helpful for enabling people suffering from aphasia (Carroll et al., 1999), autism (Barbu et al., 2015), or dyslexia (Rello et al., 2013). Moreover, text simplification can be applied as a pre-processing step in other downstream NLP tasks such as Parsing (Chandrasekar et al., 1996), Information Extraction (Miwa et al., 2010), Text Summarization (Siddharthan et al., 2004) and Machine Translation (Štajner and Popović, 2016).

Two types of simplification can be defined based on the source text: sentence simplification and document simplification (Sun et al., 2021). Sentence simplification can be applied to texts with several sentences, one at a time, meaning that the number of sentences in the input and output would be the same. Conversely, document simplification can reduce the number of sentences in the output text.

Currently, text simplification research has been more focused on sentence simplification (Sheang and Saggion, 2021; Martin et al., 2021). The most commonly used datasets for text simplification such as WikiLarge (Zhang and Lapata, 2017), TurkCorpus (Xu et al., 2016a), and Newsela (Xu et al., 2015) are originally designed for sentence simplification. However, various applications in the real world require document-level simplification rather than sentence-level processing. This is due to the need to understand the main points of several sentences at once and rewrite them in a simplified vocabulary and grammar structure without respecting a number of sentences. Thus, document-level text simplification may have more applications than text simplification at the sentence level.

In this paper, we concentrate on document-level text simplification. The main contributions of our work include:

[*] Equal Contribution.

[1] https://github.com/epfml/easy-summary/tree/main

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 9927–9944
July 9-14, 2023 ©2023 Association for Computational Linguistics

- We propose a two-stage model SimSum for document-to-document simplification tasks, which combines text simplification and summarization tasks innovatively. The main idea of the architecture is simultaneous summarization and simplification at the document level.
- We analyse and pre-process two document-level simplification datasets, and make the resulting datasets available for reproducibility.
- We propose two approaches including *Keyword Prompt* and *Embedding Similarity* to enhance the performance of our model.

The remainder of the paper is structured as follows: Section 2 presents related work on text simplification, text summarization as well as multi-stage generation, which all highlight the principles of our model. In Section 3, we present our novel architecture. Then, Section 4 presents our dataset descriptions and preprocessing steps. Section 5 includes the set of experiments. In Section 5.3, we combine insights from our study to obtain state-of-the-art results on two document-level benchmarks. Finally, we provide an ablation study on *Keyword Prompt* and *Embedding Similarity loss* in Section 6 and the human evaluation of different models' generations in Section 7.

2 Related Work

Since our proposed model combines a *Summarizer* and a *Simplifier* modules (as we describe in Section 3), we present the related work by focusing on both text simplification and text summarization tasks.

2.1 Text Simplification

The goal of sentence simplification is to simplify the original (usually complex) sentence into a more understandable sentence through several operations including deletion, addition, and splitting of words and phrases (Sun et al., 2021). Sentence simplification can be regarded as a machine translation task, mapping complex language to a simplified, albeit semantically similar, alternative. Several earlier approaches were inspired by statistical machine translation (SMT) (Coster and Kauchak, 2011; Wubben et al., 2012; Narayan and Gardent, 2014; Štajner et al., 2015; Xu et al., 2016a). Neural Text Simplification (Nisioi et al., 2017) shows a better performance than SMT. Also, reinforcement learning can be applied to obtain competitive results (Zhang and Lapata, 2017). In addition, Vu et al. (2018) applied memory-augmentation techniques to neural networks to improve the performance on sentence-level simplification. Kriz et al. (2019) proposed two main approaches to alleviate direct copy from original document issues. Dong et al. (2019) presented the first sentence simplification model that learns three explicit edit operations. Sheang and Saggion applied the large pre-trained language model T5 (Raffel et al., 2019) along with the controllable tokens on the sentence simplification task. In this paper, we also explore a number of prompting techniques in the context of text simplification.

Furthermore, several works concentrate on document-level text simplification. Alva-Manchego et al. (2019b) demonstrated that there are frequent rewriting transformations with no limit to sentence boundaries. Sun et al. (2021) investigated the task of document-level text simplification, provided a large-scale dataset called D-Wikipedia, and proposed a more suitable evaluation metric than SARI (Xu et al., 2016b) named D-SARI in the document-level simplification task.

2.2 Text Summarization

Summarization approaches are divided into two main categories, extractive and abstractive.

Extractive. Extractive summarization methods select the most important sentences within a text, therefore the resulting summary is a subset of the original sentences in the full text. Recently, BERT-based extractors (Devlin et al., 2018), (Zhong et al., 2020) achieved state-of-the-art performance in extractive summarization of relatively short documents from the CNN/DailyMail (Hermann et al., 2015) dataset. We design a similar component in SimSum to extract the most important keywords of a text.

Abstractive. In recent years, the success of transformer-based architectures in different natural language generation tasks (Vaswani et al., 2017) has inspired researchers to utilize such architectures for the abstractive summary generation problem. BART-based models (Lewis et al., 2019), or (Liu et al., 2022) which is one of the top baselines in text simplification corrupted text with an arbitrary noising function and learned to reconstruct the original text. For generation tasks, the noising function was text infilling which used single mask tokens to mask randomly sampled spans of text. T5 (Raffel et al., 2019) generalized the text-to-text framework to a variety of NLP tasks and showed the advantage of scaling up model pre-training corpus sizes. T5 was pre-trained with randomly corrupted text spans of varying mask ratios and sizes of spans. PEGASUS (Zhang et al., 2019) masks multiple whole sentences rather than smaller continuous text spans. It does not reconstruct full input sequences and only generates the masked sentences as a single output sequence. Other flavors of abstractive summarization (Bahrainian et al., 2021b, 2022) involve controlling the generation process to highlight specific topics (Bahrainian et al., 2021a) in the output summary via prompting or modifying the standard attention mechanism.

2.3 Multi-Stage Generation

Multi-stage coarse-to-fine frameworks were studied in different natural language generation tasks, which can in part resemble our model's two-stage architecture. Chen et al. (2020) proposed a dialogue state tracking approach, Fan et al. (2018) explored the story generation task, Xu and Lapata (2020) designed a *coarse-to-fine* framework for multi-document summarization. Recently, Zhang et al. (2022) proposed a simple and effective multi-stage framework to handle longer input texts for language models in a text summarization task.

In this paper, we present the first model that explicitly incorporates both summarization and simplification components for multi-stage generation and as a result achieves top performance in simplicity and readability metrics.

3 Method

We introduce a new model for document-level text simplification consisting of two main components: A *Summarizer* transformer and a *Simplifier* transformer, which jointly aim to address the document-level simplification task trained in an end-to-end fashion. The motivation behind such a framework is that the document-level simplification (Sun et al., 2021) task requires retaining the primary information from the original text (where a summarization model can be useful) while making text comprehension easier (where a simplification model can help). Figure 1 demonstrates the workflow of our model. The first stage is the pre-trained *Summarizer*. Then, the output from the *Summarizer* without tokenizer's decoding feeds into the second stage – a pre-trained sentence-to-sentence simplification transformer. This enables end-to-end training for our model. If we retokenize the decoded sentence after the *Summarizer* step, gradients are restricted from flowing through both models during the training. First summarizing text and then simplifying it makes intuitive sense due to the fact that using a summarizer at the second stage may result in rewriting a simplified text in a complex language. We also observed this issue experimentally and therefore only proceed with the current order.

Furthermore, existing datasets on both the text summarization task and sentence-level simplification task indicate that we can fine-tune each module on the corresponding task.

3.1 Backbone

BART (Lewis et al., 2019) and T5 (Raffel et al., 2019) have both shown high performances on various NLP tasks including text summarization. We use the pre-trained versions of both of these architectures to initialize our model SIMSUM.

In detail, for the simultaneous summarization and simplification stages in one version of our model, we use pre-trained T5 models (i.e., with the summarization stage using a pre-trained summarization T5) for both stages and in another version of SIMSUM we use BART pre-trained models in the same way. Subsequently, we fine-tune both model variants on the WikiLarge (Zhang and Lapata, 2017) dataset for sentence-level simplification task.

3.2 Keyword Prompts

Inspired by the Controllable Sentence Simplification (Sheang and Saggion, 2021) approach, we use the *Keyword Prompt* notion to force the model to focus more on important keywords in each input text. In order to extract those main keywords we use KeyBERT (Grootendorst, 2020), which derives the most important themes discussed in the original text in the form of keywords.

We examine two different strategies for prompting. The first one is `kw_score`, which adds keywords with their similarity score in front of the input text. We examine this type of prompting to investigate the effectiveness of keywords extracted by KeyBERT in order to guide the generation task. Each keyword is followed by a salience score as computed by KeyBERT. The second one is `kw_sep`, which adds keywords and EOS (End Of Sentence) tokens `</s>` in front of the input text. In this variation, we use the same keywords without including the salience score. In the latter setting, we use the

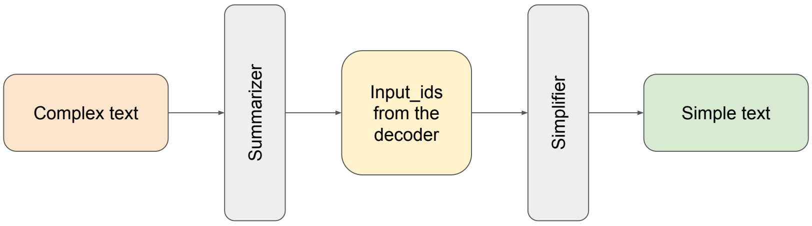

Figure 1: The workflow of the proposed framework. It contains two stages: the first one is *Summarizer*, and the second one is *Simplifier*. The generated output of the *Summarizer* is fed to the *Simplifier* without tokenizer's decoding since it does not allow the gradients to flow back to the *Summarizer*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

Input text (original)
a goatee is a style of facial hair incorporating hair on one 's chin but not on one 's cheeks . the exact nature of the style has varied according to time and culture .
Input text with `kw_score` as prompt
one_0.06 varied_0.07 goatee_0.76 a goatee is a style of facial hair incorporating hair on one 's chin but not on one 's cheeks . the exact nature of the style has varied according to time and culture .
Input text with `kw_sep` as prompt
one varied goatee </s> a goatee is a style of facial hair incorporating hair on one 's chin but not on one 's cheeks . the exact nature of the style has varied according to time and culture .

Table 1: *kw_score* and *kw_sep* prompting examples

EOS token to separate the prompts (keywords) and source sentences. Table 1 shows examples of modification of input text with `kw_score` prompt and `kw_sep` prompt, respectively.

3.3 Embedding Similarity

One of the most common approaches for training sequence-to-sequence Transformer models is the use of standard maximum likelihood measures and the cross-entropy loss (Raffel et al., 2019). However, this method can be improved with an additional loss term that forces the model to generate texts more similar to targets. Therefore, we propose a new loss function that consists of $\mathcal{L}_1$ – the original cross-entropy loss and $\mathcal{L}_{\text{CosSim}}$ – new additional term:

$$\mathcal{L} = \mathcal{L}_1 + \lambda \cdot \mathcal{L}_{\text{CosSim}} \tag{1}$$

Also, we design the hyper-parameter $\lambda > 0$ as a control mechanism for changing the degree of contribution of the additional term.

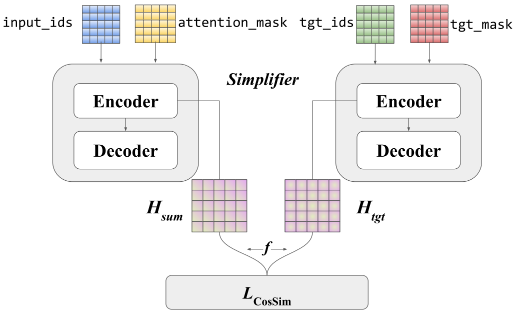

Figure 2: The embedding similarity computation process. `tgt_ids` and `tgt_mask` are obtained by tokenizing the targets. From both **Encoder**s the last hidden state H_{sum} and H_{tgt} are obtained. Then, after transformation f the cosine similarity is computed.

Our idea is to increase the similarity between the final output's embeddings and the target's embeddings during training. To obtain the target embeddings, we feed the target to the *Simplifier* as input and take the embeddings of the last hidden state of the encoder as the input to $\mathcal{L}_{\text{CosSim}}$ loss term.

To this end, the cosine distance is chosen to measure the similarity. Since we can only get the summarization's encoding representation generated from *Summarizer*, we apply the function $f(\cdot)$ to transform the embeddings to simplified-text space:

$$\mathcal{L}_{\text{CosSim}} = -\text{CosSim}(f(H_{\text{sum}}), f(H_{\text{tgt}})) \tag{2}$$

where H_{sum} and H_{tgt} represent the summarization and target embeddings, respectively. Both H_{sum} and H_{tgt} are in $\mathbb{R}^{B \times L \times D_1}$, where B, L, D_1 denote the batch size, sequence length, and hidden size respectively. Figure 2 shows the details of the embedding similarity calculations.

In our experiments, we set the transformation function $f(\cdot)$ as:

$$f(H) = \text{ReLU}(HW) \tag{3}$$

where $W \in \mathbb{R}^{D_1 \times D_2}$ denotes a learnable transition matrix. To keep the important information and filter out unimportant pieces of information, we rely on ReLU (Fukushima, 1975) activation functions in $f(H)$.

4 Datasets

Most of the widely used datasets, such as WikiLarge (Zhang and Lapata, 2017), TurkCorpus (Xu et al., 2016a) and Newsela (Xu et al., 2015), are designed for sentence-level text simplification and are not applicable to our document-level text simplification task.

Fortunately, Sun et al. (2021) has already adjusted the pre-processed dataset D-Wikipedia for the document-to-document simplification task. However, the dataset requires additional pre-processing since there exist noisy samples with a lot of mismatches in the information presented in the source and target pairs (See Section 4.1 for a more detailed discussion).

The second dataset for document-level simplification is text articles from Wikipedia created by (Kauchak, 2013), which we refer to as Wiki-Doc. It contains 59,749 samples which we split 8:1:1 into training (47,799 samples), validation (5,975 samples), and test (5,975 samples) sets. The Wiki-Doc dataset contains unaligned pairs, texts with a length greater than 3,000 tokens, and pairs where the simple text is longer than a complex one. These observations motivate several pre-processing steps described in the next section.

4.1 Pre-processing

In this section, we introduce two steps to pre-process D-Wikipedia and Wiki-Doc datasets.

4.1.1 Filtering

We assume that simplified texts should be shorter than the corresponding original documents since they consist of fewer and simpler sentences. By lowering the information load on the reader, his or her ability to comprehend the text increases (Chamovitz and Abend, 2022). Furthermore, we observe that there exist "extremely noisy" pairs where simple texts are as much as two times longer than the original source documents because of external knowledge or errors during the dataset collection (see Appendix A for examples). The number of documents where the length of the simplified reference is longer than the original in the Wiki-Doc is 6,476(13.54%) in the training set, 797(13.33%) in the validation set, and 802(13.42%) in the test set. The same statistics in the D-Wikipedia dataset are 39,017(29.55%) in the training set, 894(29.8%) in the validation set, and 2,377(29.71%) in the test set. Given the large percentages of these instances, they should be removed from the datasets.

However, there are still many reasonable samples in which simplified texts are longer than original documents due to the conceptual simplification (Gooding, 2022) (similar to Appendix A Example 1), which helps explain complex concepts via simple words. Therefore, considering the above cases we keep the pairs where the simple text is at most 5 words longer than the source text.

4.1.2 Re-alignment

We observed that there are misaligned pairs in both datasets, i.e., pairs where the complex source text does not correspond to the simple target. To identify if the pairs are aligned correctly, we apply the KeyBERT model (Grootendorst, 2020) to extract top-k keywords from both source and target texts. Here we set $k = 5$. Then, we compare the two sets of keywords. If there is at least one overlapping keyword, we assume this source-target pair to be aligned correctly, otherwise, we remove the pair from the dataset. Examples of unaligned pairs in the D-Wikipedia dataset are presented in Appendix A. Moreover, we show the output keywords produced by KeyBERT for align-check in Appendix E.

After the pre-processing steps, D-Wikipedia contains 97,074 training samples, 2,183 validation samples, and 5,836 test samples. Wiki-Doc contains 13,973 training samples, 1,768 validation samples, and 1,704 test samples.

Table 2 shows the basic statistics of the D-Wikipedia vs. Wiki-Doc datasets after pre-processing.

The pre-processed datasets described above are available at `https://github.com/epfml/easy-summary/tree/main` along with our entire codebase to facilitate reproducing our results, as well as, to contribute the clean versions of simplification datasets to the community in order to advance document simplification research.

5 Experiments

5.1 Baselines

In this evaluation, we compare our novel model against state-of-the-art text simplification ap-

	D-Wikipedia		Wiki-Doc	
	Complex	*Simple*	*Complex*	*Simple*
Total sentences	546,744	349,561	258,303	55,885
Total words	17,740,142	703,550	5,927,616	906,988
Avg sents per article	5.20	3.33	14.81	3.20
Avg words per sent	32.45	20.24	22.95	16.23

Table 2: Basic statistics of D-Wikipedia vs. Wiki-Doc datasets after pre-processing. Wiki-Doc has more sentences per article on average than D-Wikipedia in complex articles, but for simple articles, the average sentence number is almost the same.

	D-Wikipedia			Wiki-Doc		
model	SARI↑	D-SARI↑	FKGL↓	SARI↑	D-SARI↑	FKGL↓
T5	45.64	36.23	8.36	50.63	41.05	6.79
BART	47.05	38.13	8.14	49.55	40.95	7.93
$\text{BART}^{\dagger}_{CNN}$	44.52	36.01	8.32	49.39	40.98	7.70
BRIO	48.24	29.86	6.39	48.65	33.06	6.84
MUSS	39.45	26.43	12.72	35.99	27.94	10.91
$\text{SimSum(T5)}^{\clubsuit}$	49.04	39.54	6.04	50.20	40.32	**6.75**
$\text{SimSum(BART)}^{\clubsuit}$	48.33	37.11	6.48	**50.67**	41.42	7.55
$\text{SimSum(T5)}^{\ddagger}$	**49.44**	**39.77**	**6.04**	49.11	**41.53**	6.79

Table 3: Our SimSum models' performance compared with the baselines. †: BART-base fine-tuned on CNN/Dailymail summarization dataset (See et al., 2017). T5 and BART in brackets mean SimSum model takes T5 or BART as the backbone in both *Summarizer* and *Simplifier*. ♣: Vanilla SimSum model without *Keyword Prompt* and *Embedding Similarity loss*. ‡: SimSum model with *Keyword Prompt* (4 `kw_score div`=0.9) and *Embedding Similarity loss* (λ=0.001).

proaches:

MUSS (Martin et al., 2021) is a Transformer-based multilingual sentence simplification system that uses multiple training strategies for simplification and achieves state-of-the-art results on the text-simplification task.

BRIO (Liu et al., 2022) is also a pre-trained model with top performance on various sequence-to-sequence tasks. Here we fine-tune their provided model checkpoint(`Yale-LILY/brio-cnndm-uncased`), which is based on BART-large.

BART (Lewis et al., 2019) is an effective model pre-trained on a large corpus that achieves excellent results on various sequence-to-sequence tasks including the text-simplification task on the sentence level (Clive et al., 2021). Here we select the BART-base version.

T5 (Raffel et al., 2019) is an encoder-decoder model proposed by Google pre-trained on a multi-task mixture of unsupervised and supervised tasks. Here we also select the T5-base version.

5.2 Evaluation Metrics

Following previous work (Sun et al., 2021), we use standard text simplification evaluation metrics:

SARI (Xu et al., 2016b) compares the system output against references and against the input sentence, which explicitly measures the goodness of words that are added, deleted, and kept by the systems. It is the most popular used metric for text simplification task.

D-SARI (Sun et al., 2021) is a modified SARI score with additional penalty factors based on text length and specially designed for the document-level text simplification task.

FKGL (Kincaid et al., 1975) is used to measure readability but does not consider grammar or meaning preservation.

We compute SARI and FKGL using EASSE (Alva-Manchego et al., 2019a), a Python3 package created to standardize automatic evaluation and comparison of sentence simplification systems.

5.3 Results

The results of our models' performance along with baselines' scores are shown in Table 3. Details on hyperparameter choices and model configuration are presented in Appendix B.

	D-Wikipedia			**Wiki-Doc**		
model	SARI↑	D-SARI↑	FKGL↓	SARI↑	D-SARI↑	FKGL↓
Without prompting(Vanilla)	49.04	39.54	**6.04**	**50.20**	40.32	6.75
3 kw_score div=0.5	49.07	39.69	6.4	49.92	41.68	6.48
3 kw_score div=0.7	**49.18**	39.65	6.38	49.90	**41.96**	6.66
3 kw_sep div=0.7	48.53	38.85	6.11	47.69	39.58	**6.05**
4 kw_score div=0.7	49.01	39.97	6.33	49.74	41.85	6.48
3 kw_score div=0.9	49.12	39.65	6.32	49.90	41.94	6.63
4 kw_score div=0.9	49.13	**40.07**	6.42	49.71	41.89	6.48

Table 4: Results on D-Wikipedia and Wiki-Doc by SimSum (T5-backbone) with Keyword Prompts. `div` denotes the parameter of the diversity of the extracted keywords in KeyBERT. 3 `kw_score` means 3 keywords in `kw_score` strategy.

	D-Wikipedia			**Wiki-Doc**		
model	SARI↑	D-SARI↑	FKGL↓	SARI↑	D-SARI↑	FKGL↓
$\lambda = 0$ (Vanilla)	49.04	39.54	6.04	**50.20**	**40.32**	6.75
$\lambda = 0.001$	**49.21**	38.51	6.12	49.88	40.03	6.65
$\lambda = 0.01$	48.94	38.27	6.26	50.02	40.15	6.75
$\lambda = 0.1$	49.02	39.39	6.09	49.92	39.90	6.69
$\lambda = 0.5$	49.16	**39.85**	5.98	46.09	36.25	6.48
$\lambda = 0.5^\dagger$	36.48	24.78	**1.47**	35.61	26.37	**5.67**
$\lambda = 1.0$	48.82	38.38	6.31	39.79	31.86	6.57

Table 5: Results on D-Wikipedia and Wiki-Doc by SimSum (T5-backbone) with Embedding Similarity loss. λ denotes the hyper-parameter that controls the contribution of the additional term. †: Identity map, i.e. $f(H) = H$.

Document Simplification on D-Wikipedia dataset. In Table 3, it can be seen that all SimSum models outperform the baselines on the SARI scores. Moreover, SimSum models with T5 backbone outperform all the baselines on D-SARI and FKGL scores. In detail, SimSum (T5 backbone) with *Keyword Prompt* and *Embedding Similarity loss* improves the SARI (+1.20), D-SARI (+1.64), and FKGL (-0.35) compared to the best baseline performances (BRIO, BART and BRIO models respectively). Therefore, SimSum (T5 backbone) with *Keyword Prompt* and *Embedding Similarity loss* archives state-of-the-art results on SARI, D-SARI, and FKGL on the D-Wikipedia dataset.

Document Simplification on Wiki-Doc dataset. Our SimSum models show superior results in terms of SARI, D-SARI, and FKGL metrics on the Wiki-Doc dataset. Specifically, the SimSum‡ with T5 backbone improves D-SARI (+0.48) compared to the best baseline performance (T5 model).

We conclude that our model performs better than baseline models in terms of SARI, D-SARI, and FKGL scores on two important simplification datasets. We present qualitative examples generated by the various models in Appendix D and additional statistics of the outputs of the models in Appendix C.

6 Ablation Study

Given that T5 is pre-trained on a mixture of supervised and unsupervised tasks, as well as making relatively lower computational demands than BART, in the following experiments with *Keyword Prompt* and *Embedding Similarity*, we only demonstrate the performance of the T5-based variant of SimSum.

6.1 Impact of Keyword Prompt

In this section, we explore the influence of the various *Keyword Prompting* strategies on our SimSum (T5-backbone) model. Table 4 shows the relevant comparisons.

On D-Wikipedia, the use of *Keyword Prompt* improves the model's performance on the SARI and D-SARI scores with `kw_score` with four keywords in comparison to the Vanilla model. Also, on the Wiki-Doc dataset, the use of *Keyword Prompt* improves the model's performance on the D-SARI score with 3 keywords in `kw_score` strategy in comparison to the Vanilla model. Examples D.3, and D.4 in Appendix D demonstrate that it can help the model extract correct and important information.

Specifically, the `kw_score` prompting strategy achieves superior results compared to `kw_sep` on SARI and D-SARI scores on both datasets. One possible explanation is that the sequence of the key-

Model	S	C	F
T5	0.36	0.78	0.80
BART	0.44	**0.90**	**0.88**
BRIO	0.46	0.42	0.74
MUSS	0.42	0.80	0.72
SimSum(T5)$^{\clubsuit}$	**0.82**	0.68	0.80
SimSum(BART)$^{\clubsuit}$	0.58	0.56	0.82
SimSum(T5)$^{\diamondsuit}$	**0.82**	0.84	**0.88**

Table 6: Human evaluation average results on D-Wikipedia. ♣: Vanilla SimSum model. ♢: SimSum model (T5-backbone) with *Keyword Prompt* (4 `kw_score` `div`=0.9). S, C, and F denote Simplicity, Correctness, and Fluency, respectively.

Model	S	C	F
T5	0.48	0.64	0.72
BART	0.60	0.72	0.78
BRIO	0.42	0.58	0.54
MUSS	0.48	0.68	0.56
SimSum(T5)$^{\clubsuit}$	0.52	0.68	0.64
SimSum(BART)$^{\clubsuit}$	0.56	**0.78**	**0.82**
SimSum(T5)$^{\diamondsuit}$	**0.66**	0.64	0.68

Table 7: Human evaluation average results on Wiki-Doc. ♣: Vanilla SimSum model. ♢: SimSum model (T5-backbone) with *Keyword Prompt* (3 `kw_score` `div`=0.7). S, C, and F denote Simplicity, Correctness, and Fluency, respectively.

words may be regarded as a disordered sentence, which confuses our model. In addition, we make an interesting observation in Table 4 that increasing the diversity of keywords (i.e. a hyperparameter of KeyBERT) improves the D-SARI score on both datasets.

6.2 Impact of Embedding Similarity loss

In this section, we explore the influence of the *Embedding Similarity loss* on our SimSum (T5-backbone) model. Table 5 shows the result comparisons. It can be seen that the optimal choice of λ for the D-Wikipedia dataset is 0.5.

In addition, the experiments with identity mapping function $f(H) = H$ show a significant drop in the performance on SARI and D-SARI scores on both datasets, which indicates that directly making summarization embeddings H_{sum} closer to target embeddings H_{tgt} is not proper and it may reduce the efficacy of the *Simplifier*.

7 Human Evaluation

In addition to the automatic evaluation, we performed a human evaluation of the outputs from different models. We run the assessment on 50 randomly selected samples from each dataset, thus 100 in total. We recruited two expert human evaluators to independently evaluate the generated texts from seven models. Following (Sheang and Saggion, 2021), we select three aspects to define our evaluation criteria: (1) Simplicity (**S**): is the output simpler than the original document?, (2) Correctness (**C**): Does the output have factual errors compared to the original document?, and (3) Fluency (**F**): is the output grammatically correct and well-formed? We chose the binary evaluation system to decrease the bias in the 5-point evaluation system. Table 6 and Table 7 report the average results in D-Wikipedia and Wiki-Doc, respectively.

D-Wikipedia dataset. SimSum with *Keyword Prompt* shows the highest values on Simplicity and Fluency. Although BART presents a better capacity to preserve the information of original texts, its simplification performance is much worse (-0.38) than SimSum.

Wiki-Doc dataset. SimSum (with BART-backbone) shows the best results in Correctness and Fluency. After adding the keywords, SimSum's simplification power improves. At the same time, BART outperforms other baselines in terms of all three criteria.

8 Discussion

In this section, we discuss three main points that we observed as a result of our experiments:

(1) The first point we discuss here is returning to where we started, namely, the idea of simplification through simultaneous summarization and the relationship between summarization and simplification. We discuss this point in terms of our observations with our SimSum model, as well as a general understanding of the connections between summarization and simplification among various baseline models. First, with SimSum, we observed that a two-stage summarization and simplification model introduces substantial quantitative improvements in terms of SARI, D-SARI, and FKGL on the two datasets. The two-stage generation process stems from the idea that gathering the main highlights of an input document in a summary and then simplifying them can be a useful technique for capturing the main highlights and improving comprehension at the same time.

(2) This observation leads to the question of whether a simplification model can benefit from summarization pre-training in general as our second discussion point. In other words, we would like

to investigate if a standard language model such as BART, initially pre-trained on a summarization task and subsequently fine-tuned on a simplification task, can demonstrate superior performance in terms of simplification metrics over another BART model that was not pre-trained on summarization but was only fine-tuned on the same simplification task. As shown in Table 3, the SARI, D-SARI, and FKGL scores were worse for the BART model pre-trained on summarization and then fine-tuned on simplification (i.e., BART$_{CNN}$) as compared with BART. Therefore, based on the results of this experiment on two datasets, we can conclude that a single transformer model does not benefit from being pre-trained on a summarization task before being fine-tuned for a simplification end task. However, to gain more conclusive insight into this problem, we also conducted a comparison between the two models in terms of the BLEU metric, which is more commonly used to evaluate summarization tasks than simplification tasks. The result of this experiment showed that the BLEU metric negatively correlates with SARI, D-SARI, and FKGL on both the D-Wikipedia dataset and the Wiki-Doc dataset. In other words, the BART model that was pre-trained on a summarization task showed a higher BLEU score than the one without the summarization pre-training.

(3) The third discussion point is related to the Keyword prompting mechanism that we introduced in SimSum. Despite the simplicity of this approach, it improved the SARI and D-SARI scores on the D-Wikipedia dataset and the D-SARI and FKGL scores on the Wiki-Doc dataset. Finally, SimSum variants showed superior results in terms of simplicity and fluency compared to all baseline models on both datasets, and also demonstrated higher correctness scores on the Wiki-Doc dataset in an extensive human evaluation.

9 Conclusions and Future Work

In this paper, we propose SimSum, a new model for document-level text simplification. We demonstrate that SimSum sets a new state of the art on document simplification outperforming the previously competitive MUSS baseline in terms of SARI and D-SARI scores. We also release cleaned versions of two existing large-scale datasets for text simplification. Through extensive experiments, we show that *Keyword Prompt* and *Emebedding Similarity* are beneficial and have an impact on enhancing the model's performance. Finally, we conducted a human evaluation showing that SimSum's quantitative performance advantage translates into better output simplicity, correctness, and fluency.

In the future, we plan to investigate guiding the generation process by various prompting techniques, including extensions of the KeyBERT model, entities, dynamic prompts, and methods such as chain of thought for simplification.

10 Limitations

(1) In this paper, we tackle the problem of document-level simplification. This consists in simultaneous summarization and simplification. Applying the same model to sentence-level simplification needs to be further evaluated, as sentences naturally due to their shorter length may not require summarization.

(2) In addition, we did not explore various model sizes although we do conduct a fair comparison and show that even with a base-model size SimSum performs superior to baselines.

Acknowledgements

This research was supported in part by SNSF grant (200020_200342), SNSF (P2TIP2_187932), NSF (IIS-1956221), ODNI, and IARPA via the BETTER program (2019-19051600004). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of NSF, SNSF, ODNI, IARPA or the U.S. Government.

References

Fernando Alva-Manchego, Louis Martin, Carolina Scarton, and Lucia Specia. 2019a. EASSE: Easier automatic sentence simplification evaluation. 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): System Demonstrations*, pages 49–54, Hong Kong, China. Association for Computational Linguistics.

Fernando Alva-Manchego, Carolina Scarton, and Lucia Specia. 2019b. Cross-sentence transformations in text simplification. In *Proceedings of the 2019 Workshop on Widening NLP*, pages 181–184, Florence, Italy. Association for Computational Linguistics.

Seyed Ali Bahrainian, Sheridan Feucht, and Carsten Eickhoff. 2022. NEWTS: A corpus for news topic-focused summarization. In *Findings of the Associa-*

tion for Computational Linguistics: ACL 2022, pages 493–503, Dublin, Ireland. Association for Computational Linguistics.

Seyed Ali Bahrainian, Martin Jaggi, and Carsten Eickhoff. 2021a. Self-supervised neural topic modeling. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3341–3350. Association for Computational Linguistics.

Seyed Ali Bahrainian, George Zerveas, Fabio Crestani, and Carsten Eickhoff. 2021b. Cats: Customizable abstractive topic-based summarization. *ACM Transactions on Information Systems (TOIS)*, 40(1):1–24.

Eduard Barbu, M Teresa Martín-Valdivia, Eugenio Martinez-Camara, and L Alfonso Urena-López. 2015. Language technologies applied to document simplification for helping autistic people. *Expert Systems with Applications*, 42(12):5076–5086.

Yue Cao, Hui Liu, and Xiaojun Wan. 2020. Jointly learning to align and summarize for neural cross-lingual summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6220–6231, Online. Association for Computational Linguistics.

John A Carroll, Guido Minnen, Darren Pearce, Yvonne Canning, Siobhan Devlin, and John Tait. 1999. Simplifying text for language-impaired readers. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 269–270.

Eytan Chamovitz and Omri Abend. 2022. Cognitive simplification operations improve text simplification.

Raman Chandrasekar, Christine Doran, and Srinivas Bangalore. 1996. Motivations and methods for text simplification. In *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*.

Zhi Chen, Lu Chen, Zihan Xu, Yanbin Zhao, Su Zhu, and Kai Yu. 2020. Credit: Coarse-to-fine sequence generation for dialogue state tracking. *arXiv preprint arXiv:2009.10435*.

Jordan Clive, Kris Cao, and Marek Rei. 2021. Control prefixes for parameter-efficient text generation.

William Coster and David Kauchak. 2011. Simple English Wikipedia: A new text simplification task. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 665–669, Portland, Oregon, USA. Association for Computational Linguistics.

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

Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Chi Kit Cheung. 2019. EditNTS: An neural programmer-interpreter model for sentence simplification through explicit editing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3393–3402, Florence, Italy. Association for Computational Linguistics.

Yue Dong, Yikang Shen, Eric Crawford, Herke van Hoof, and Jackie Chi Kit Cheung. 2018. Banditsum: Extractive summarization as a contextual bandit. *arXiv preprint arXiv:1809.09672*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Kunihiko Fukushima. 1975. Cognitron: A self-organizing multilayered neural network. *Biological cybernetics*, 20(3):121–136.

Goran Glavaš and Sanja Štajner. 2015. Simplifying lexical simplification: Do we need simplified corpora? In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 63–68, Beijing, China. Association for Computational Linguistics.

Sian Gooding. 2022. On the ethical considerations of text simplification. In *Ninth Workshop on Speech and Language Processing for Assistive Technologies (SLPAT-2022)*, pages 50–57, Dublin, Ireland. Association for Computational Linguistics.

Maarten Grootendorst. 2020. Keybert: Minimal keyword extraction with bert.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend.

Tomoyuki Kajiwara, Hiroshi Matsumoto, and Kazuhide Yamamoto. 2013. Selecting proper lexical paraphrase for children. In *Proceedings of the 25th Conference on Computational Linguistics and Speech Processing (ROCLING 2013)*, pages 59–73, Kaohsiung, Taiwan. The Association for Computational Linguistics and Chinese Language Processing (ACLCLP).

David Kauchak. 2013. Improving text simplification language modeling using unsimplified text data. In *Proceedings of the 51st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 1537–1546.

J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report, Naval Technical Training Command Millington TN Research Branch.

Reno Kriz, João Sedoc, Marianna Apidianaki, Carolina Zheng, Gaurav Kumar, Eleni Miltsakaki, and Chris Callison-Burch. 2019. Complexity-weighted loss and diverse reranking for sentence simplification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3137–3147, Minneapolis, Minnesota. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Yixin Liu, Pengfei Liu, Dragomir Radev, and Graham Neubig. 2022. Brio: Bringing order to abstractive summarization.

Louis Martin, Angela Fan, Éric de la Clergerie, Antoine Bordes, and Benoît Sagot. 2021. Muss: Multilingual unsupervised sentence simplification by mining paraphrases. *arXiv preprint arXiv:2005.00352*.

Derek Miller. 2019. Leveraging bert for extractive text summarization on lectures. *arXiv preprint arXiv:1906.04165*.

Makoto Miwa, Rune Saetre, Yusuke Miyao, and Jun'ichi Tsujii. 2010. Entity-focused sentence simplification for relation extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 788–796.

Shashi Narayan and Claire Gardent. 2014. Hybrid simplification using deep semantics and machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 435–445, Baltimore, Maryland. Association for Computational Linguistics.

Sergiu Nisioi, Sanja Štajner, Simone Paolo Ponzetto, and Liviu P. Dinu. 2017. Exploring neural text simplification models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 85–91, Vancouver, Canada. Association for Computational Linguistics.

Gustavo Henrique Paetzold. 2016. *Lexical simplification for non-native english speakers*. Ph.D. thesis, University of Sheffield.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer.

Luz Rello, Ricardo Baeza-Yates, Stefan Bott, and Horacio Saggion. 2013. Simplify or help? text simplification strategies for people with dyslexia. In *Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility*, pages 1–10.

Horacio Saggion. 2017. Automatic text simplification. *Synthesis Lectures on Human Language Technologies*, 10(1):1–137.

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

Kim Cheng Sheang and Horacio Saggion. 2021. Controllable sentence simplification with a unified text-to-text transfer transformer. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 341–352, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Advaith Siddharthan, Ani Nenkova, and Kathleen McKeown. 2004. Syntactic simplification for improving content selection in multi-document summarization. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 896–902, Geneva, Switzerland. COLING.

Sanja Štajner, Iacer Calixto, and Horacio Saggion. 2015. Automatic text simplification for Spanish: Comparative evaluation of various simplification strategies. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 618–626, Hissar, Bulgaria. INCOMA Ltd. Shoumen, BULGARIA.

Sanja Štajner and Maja Popović. 2016. Can text simplification help machine translation? In *Proceedings of the 19th Annual Conference of the European Association for Machine Translation*, pages 230–242.

Renliang Sun, Hanqi Jin, and Xiaojun Wan. 2021. Document-level text simplification: Dataset, criteria and baseline. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7997–8013, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Tu Vu, Baotian Hu, Tsendsuren Munkhdalai, and Hong Yu. 2018. Sentence simplification with memory-augmented neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 79–85, New Orleans, Louisiana. Association for Computational Linguistics.

Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 409–420, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Sander Wubben, Antal van den Bosch, and Emiel Krahmer. 2012. Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1015–1024, Jeju Island, Korea. Association for Computational Linguistics.

Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, 3:283–297.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016a. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016b. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Yumo Xu and Mirella Lapata. 2020. Coarse-to-fine query focused multi-document summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3632–3645, Online. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2019. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization.

Xingxing Zhang and Mirella Lapata. 2017. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 584–594, Copenhagen, Denmark. Association for Computational Linguistics.

Yusen Zhang, Ansong Ni, Ziming Mao, Chen Henry Wu, Chenguang Zhu, Budhaditya Deb, Ahmed Awadallah, Dragomir Radev, and Rui Zhang. 2022. Summ^n: 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: Long Papers)*, pages 1592–1604, Dublin, Ireland. Association for Computational Linguistics.

Sanqiang Zhao, Rui Meng, Daqing He, Andi Saptono, and Bambang Parmanto. 2018. Integrating transformer and paraphrase rules for sentence simplification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3164–3173, Brussels, Belgium. Association for Computational Linguistics.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208, Online. Association for Computational Linguistics.

A Examples of Miss-aligned Pairs In D-Wikipedia Dataset

A.1 Example 1

Source (Original): andouille (; ; from vulgar latin verb " inducere " , meaning " to lead in ") is a smoked sausage made using pork , originating in france .
Reference (Simplified): andouille is a type of pork sausage . it is spicy (hot in taste) and smoked . there are different kinds , all with different combinations of pork meat , fat , intestines (tubes going to the stomach) , and tripe (the wall of the stomach) . andouille sausage first came from france or germany (no-one is sure)...
Analysis: In reference, it demonstrates the conceptual simplification, which aims to help people understand easier. However, the simplified referenced article is much longer than the source article, and it also contains impossible deduced information.

A.2 Example 2

Source (Original): creator is something or someone who brings something into being .
Reference (Simplified): a creator is a person who creates something.in some religions (judaism , christianity , islam) god (or allah meaning the god in arabic) is the most important and original creator of the whole universe - including man who is made " in his image " (see genesis) to observe it and control it like god . the idea that anything that a person is creating , like an idea , can be owned as property comes from the ethical traditions and legal codes that came from these religions.in other traditions (buddhism , native american mythology) anyone has this potential for creating , and can become part of the greater creating of the universe . stewardship of home , land and all of earth is a test for participating in this , or just good sense .
Analysis: This example is more "absurd", where the reference introduces much more external information which cannot be inferred from the source

article.

A.3 Example 3

Source (Original): evacuation or evacuate may refer to :
Reference (Simplified): evacuation is a term that comes from latin language . literally , it means “ emptying ” . most of the time it means that people leave an area or a zone because of an imminent danger or threat . evacuations are planned , and people should leave the zone of danger according to the plan . the causes for evacuations can be natural disasters such as flooding or volcano eruptions , or they can be man-made ones , like a bomb in a building , a fire on a train in a tunnel or an industrial accident .
Analysis: In this example, the source article is not complete. This problem might be caused by errors during the dataset collection.

A.4 Example 4

Source (Original): andradina is a municipality of the state of são paulo , brazil.the population is 57,250 (2015 est .) in an area of 964.23 km².the municipality contains the andradina biological reserve , a strictly protected area.the municipality can be accessed mainly by rodovia marechal cândido rondon/br-300 (marechal rondon highway) .
Reference (Simplified): edi carlo dias marçal (born 13 september 1974) is a brazilian football player . he plays for korona kielce .
Analysis: Source article is mainly about a state named *andradina*, but the reference actually describes a Brazilian football player. This pair is definitely not aligned correctly.

A.5 Example 5

Source (Original): sushun ’s reign spanned the years from 587 through 592 .
Reference (Simplified): the conventionally accepted names and sequence of the early emperors were not to be confirmed as “ traditional ” until the reign of emperor kammu , who was the 50th monarch of the yamato dynasty .
Analysis: This is also a miss-aligned pair and should be removed from the dataset.

hyperparameter	value
train_batch_size	6
valid_batch_size	6
learning rate	3e-4
adam epsilon	1e-8
weight decay	1e-4
warmup steps	5
training epochs	7
max seq length	256

Table 8: The hyperparameters of T5 model.

hyperparameter	value
train_batch_size	6
valid_batch_size	6
learning rate	1e-4
adam epsilon	1e-8
weight decay	1e-4
warmup steps	5
training epochs	7
max seq length	256

Table 9: The hyperparameters of BART model.

B Implementation Details

We used the HuggingFace[2] to implement T5, BART and BRIO with PyTorch Lightning[3], and used the code on GitHub to implement the MUSS[4]. Our SIMSUM is also implemented by the PyTorch Lightning framework. For each dataset, we fine-tuned each model individually. In detail, the hyperparameters are shown in Tables 8, 9, 10, 11, and 12.

hyperparameter	value
train_batch_size	6
valid_batch_size	6
learning rate	5e-5
adam epsilon	1e-8
weight decay	1e-4
warmup steps	5
training epochs	7
max seq length	256

Table 10: The hyperparameters of BRIO model.

[2] https://huggingface.co/
[3] https://www.pytorchlightning.ai/
[4] https://github.com/facebookresearch/muss

hyperparameter	value
train_batch_size	6
valid_batch_size	6
learning rate	3e-4
adam epsilon	1e-8
weight decay	1e-4
warmup steps	5
training epochs	7
max seq length	256

Table 11: The hyperparameters of SIMSUM (T5-backbone) model.

hyperparameter	value
train_batch_size	6
valid_batch_size	6
learning rate	5e-5
adam epsilon	1e-8
weight decay	1e-4
warmup steps	5
training epochs	7
max seq length	256

Table 12: The hyperparameters of SIMSUM (BART-backbone) model.

C Outputs' statistics

Table 13 represents the average number of tokens and sentences for each model for both datasets. In the following per model, the first number is mean±std for the number of tokens, and the second is mean±std for the number of sentences.

D Case Study

D.1 Example 1

Source (Original): marsupial moles are highly specialized marsupial mammals , known from two species found at the australian interior.notoryctids are small , fossorial mammals that anatomically converge on other fossorial (and distantly related) mammals , such as living golden moles (chrysochloridae) and extinct epoicotheres (pholidota) .

Reference (Simple): marsupial moles (notoryctidae) is a family of marsupials of the order notoryctemorphia . there are only two extant species :

Output (T5): a species of marsupial moles is a species of marsupial mammals. they are found in the australian interior.

Output (BRIO): id moles are highly specialized marsupial mammals, known from two species found only in the australian interior.notoryctids are small, fossorial mammals that anatomically converge on other fossorial (and distantly related) mammals, such as living golden moles (chrysochloridae) and extinct epoicotheres (pholidota).

Output (vanilla SIMSUM-T5 backbone): the notoryctids are small, fossorial mammals that converge on other fossorial (and distantly related) mammals. they are known from living golden moles (chrysochloridae) and extinct epoicotheres (pholidota). they are closely related to the frogs.

Analysis: The source-reference pair is taken from the D-Wikipedia dataset. The highlighted content in the ground-truth seems not completed. T5's generated text is more redundant, and BRIO even does not simplify the original text. SIMSUM model does a great job on simplifying, but it also introduces some unrelated sentences which we cannot infer from original texts.

D.2 Example 2

Source (Original): the phoenix dwarf is a dwarf irregular galaxy discovered in 1976 by hans-emil schuster and richard martin west and mistaken for a globular cluster . it is currently 1.44 mly away from earth . its name comes from the fact that it is part of the phoenix constellation .

Reference (Simple): the phoenix dwarf is a galaxy discovered as a mistaken globular cluster . it 's correctly 1.14 mly away the earth

Output (T5): the phoenix dwarf is a dwarf irregular galaxy discovered in 1976 by hans-emil schuster and richard martin west and mistaken for a globular cluster. it is currently 1.44 mly away from earth. its name comes from the fact that it is part of the phoenix constellation.

Output (BRIO): the phoenix dwarf is a dwarf galaxy discovered in 1976 by hans-emil schuster and richard martin west. it is mistaken for a globular cluster and is currently 1.44 million light years away from earth. its name comes from the fact that it is part of the pharus constellation.

Output (vanilla SIMSUM-T5 backbone): the phoenix dwarf is a dwarf irregular galaxy discovered in 1976 by hans-emil schuster and richard martin west. it is about 1,44 million light-years away from earth.

Analysis: The source-reference pair is taken from the D-Wikipedia dataset. The ground-truth removes the names and the last sentence from the

	D-Wikipedia		Wiki-Doc	
model	#tokens	#sentences	#tokens	#sentences
T5	68±44	3.1±1.7	44±40	2.6±2.0
BART	49±40	2.5±1.4	40±35	2.4±1.7
BRIO	74±39	4.0±1.8	80±48	4.9±2.5
MUSS	100±64	3.2±1.8	161±208	6.2±8.8
SimSum(T5)‡	49±29	3.0±1.9	35±26	2.3±1.5
Reference	67±57	3.3±2.4	51±51	3.1±2.8

Table 13: The average number of tokens and sentences for each model for both datasets. T5 in brackets means SimSum model takes T5 as the backbone in both *Summarizer* and *Simplifier*. ‡: SimSum model with *Keyword Prompt* (4 `kw_score div`=0.9) and *Embedding Similarity loss* (λ=0.001).

original text. T5 and BRIO models just copy the original article. BRIO generates similar texts with T5 but it splits the first long sentence. SimSum's generation is much better than others, closer to the ground-truth, and even explain abbreviations (from *mly* to *million light-years*) to help readers understand better.

D.3 Example 3

Source (Original): Robert Urich (December 19 , 1946 – April 16 , 2002) was an American actor . He played the starring role in the television series Spenser : For Hire (1985 – 1988) and Vega $ (1978 – 1981) . He also appeared in other television series over the years including : S.W.A.T. (1975) , Soap (1977) , and The Lazarus Man (1996) , as well as in several feature films including Turk 182 ! , The Ice Pirates , and Magnum Force . Urich died in 2002 in Thousand Oaks , California ...

Reference (Simple): Robert Urich (December 19 , 1946 â “ April 16 , 2002) was an Emmy-winning actor , famous for playing private investigators on the television series Spenser : For Hire (1985 â “ 1988) and Vega $ (1978 â “ 1981) . He also acted in many other television series over the years including : S.W.A.T. (1975) , Soap (1977) and The Lazarus Man (1996) .

Output (vanilla SimSum-T5 backbone): William Bill " Bartholomew (born May 1, 1973) is an American actor. He is best known for his roles in the television series Spenser : For Hire (1985 â 1988) and Vega $ (1978 â 1981). He also starred in other television series over the years including : S.W.A.T. (1975), Soap (1977), and The Lazarus Man (1996).

Output (SimSum-T5 backbone 3 kw_score div=0.7)): Robert Urich (December 19, 1946 - April 16, 2002) was an American actor. He played the starring role in the television series Spenser : For Hire (1985-1988) and Vega $ (1978-1981).

Analysis: The source-reference pair is taken from the Wiki-Doc dataset. Vanilla SimSum model makes factual errors badly, both from the name of the person and date. However, it still succeeds in keeping the important information correct as the reference. After applying the *keyword prompts*, SimSum is able to fix the errors.

D.4 Example 4

Source (Original): Brent Michael Kutzle (born August 3 , 1985) is an American musician , born in Newport Beach , California . Brent is best known for playing the bass guitar and cello for OneRepublic , who are signed to Interscope Records . He has written and performed with various other musicians from both underground music scenes and mainstream acts , including Vermeer , Augustine , Torrent , This Allure , Monarch , Venus Infers , and Jessica Dobson . When he was thirteen , he attended Sarah McGarvin School located in Westminster , California . He also attended La Quinta High School in Westminster , California , California Baptist University in Riverside and Vanguard University in Costa Mesa . Kutzle has a cameo appearance in the 2008 film , The Eye , starring Jessica Alba . He appears in the opening scene playing cello as a member of the orchestra . He can also be heard playing cello on the title track , “ Songs Like Rain ” for the 2006 independent film of the same name . Kutzle joined OneRepublic early in 2007 after the band 's former bassist left to pursue a solo career . He has played the cello since fourth grade and has commented that he would never play for a band that ‘ would not let him play his cello ’ . Kutzle has transformed the sound of the cello for a rock band by using delay , reverb and looping effects . He makes his mark on OneRepublic 's debut album Dreaming Out Loud on the track “ All Fall Down , ” which is based on a musical hook done in cello that is repeated throughout the whole track . In addition to playing the cello and the bass guitar , Kutzle , along with Zach Filkins

, also aids frontman , Ryan Tedder in songwriting and composing music for the band . Many of the songs on the band 's latest album Waking Up were co-written and co-produced by Kutzle , including the lead single All The Right Moves .

Reference (Simple): Brent Kutzle (born 1985) is an American musician , born in Newport Beach , California . He has written and performed for various musicians both underground music (Vermeer , Augustine , This Allure , Vanessa Jourdan , Venus Infers , Jessica Dobson , and Torrent) and mainstream . Brent is best known for playing the bass guitar and cello for OneRepublic (Interscope Records) . When he was 13 , he attended McGarvin Intermediate School located in Westminster , California . He also attended La Quinta High School in Westminster , California , California Baptist University in Riverside and Vanguard University in Costa Mesa .

Output (vanilla SIMSUM-T5 backbone): Brent Michael Kutzle (born August 3, 1985 in Los Angeles, California) is an American musician. He is best known for playing the bass guitar and cello for OneRepublic, who are signed to Interscope Records. He has written and performed with various other musicians from both underground music scenes and mainstream acts.

Output (SIMSUM-T5 backbone 3 kw_score div=0.7)): Brent Kutzle (born August 3, 1985 in Newport Beach, California) is an American musician. He is best known for playing the bass guitar and cello for OneRepublic. He has written and performed with various other musicians from both underground music scenes and mainstream acts.

Analysis: The source-reference pair is taken from the Wiki-Doc dataset. Vanilla SIMSUM model makes a factual error in the generation (instead of Newport Beach). In addition, SIMSUM with *keyword prompts* also manages to keep the important information that is also presented in ground-truth.

E Examples for checking misaligned pairs by KeyBERT

E.1 Example 1

Top 5 keywords from source text: [('andradina',0.6781), ('paulo',0.3045), ('population',0.2021), ('area',0.1203), ('br',0.112)]

Top 5 keywords from simple reference (target):[('edi',0.4892), ('dias',0.4876), ('marçal',0.4417), ('kielce',0.3581), ('carlo',0.3505)]

Analysis: This example is from A.4 that relates totally different contents as we showed before. We can notice that in the Top 5 keywords, there is no overlapping, which means that this pair should be removed from the original dataset in our preprocessing methods.

E.2 Example 2

Top 5 keywords from source text: [('phoenix',0.4448), ('galaxy',0.4211), ('dwarf',0.3657), ('constellation',0.3046), ('discovered',0.2794)]

Top 5 keywords from simple reference (target):[('galaxy',0.4049), ('phoenix',0.3354), ('dwarf',0.3252), ('globular',0.3043), ('discovered',0.2762)]

Analysis: This example is from D.2. The manual check (directly looking at the *source* and *reference*) and KeyBERT check both indicate that this pair is aligned and should be kept in the dataset.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
10

□ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

□ B1. Did you cite the creators of artifacts you used?
No response.

□ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

□ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

□ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

□ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

5, 6

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
6

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
5

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
5.2

D ☑ Did you use human annotators (e.g., crowdworkers) or research with human participants?
7

☑ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
7

☑ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
7

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.