Improving Copy-oriented Text Generation via EDU Copy Mechanism

Tianxiang Wu, Han Chen, Luozheng Qin, Ziqiang Cao*, Chunhui Ai
School of Computer Science & Technology, Soochow University, SuZhou, China
{txwu, hchen03, 20225227060, 20215227120}@stu.suda.edu.cn
zqcao@suda.edu.cn

Abstract

Many text generation tasks are copy-oriented. For instance, nearly 30% content of news summaries is copied. The copy rate is even higher in Grammatical Error Correction (GEC). However, existing generative models generate texts through word-by-word decoding, which may lead to factual inconsistencies and slow inference. While Elementary Discourse Units (EDUs) are outstanding extraction units, EDU-based extractive methods can alleviate the aforementioned problems. As a consequence, we propose EDUCopy, a framework that integrates the behavior of copying EDUs into generative models. The main idea of EDUCopy is to use special index tags to represent the copied EDUs during generation. Specifically, we extract important EDUs from input sequences, finetune generative models to generate sequences with special index tags, and restore the generated special index tags into corresponding text spans. By doing so, EDUCopy reduces the number of generated tokens significantly. To verify the effectiveness of EDUCopy, we conduct experiments on the news summarization datasets CNNDM, NYT and the GEC datasets FCE, WI-LOCNESS. While achieving notable ROUGE and M^2 scores, GPT-4 evaluation validates the strength of our models in terms of factual consistency, fluency, and overall performance. Moreover, compared to baseline models, EDUCopy achieves a significant acceleration of 1.65x.

Keywords: Text Generation, Copy Mechanism, Elementary Discourse Units

1. Introduction

Many text generation tasks are copy-oriented, where a significant portion of the target sequences can be obtained by directly copying some vital text spans in the input sequences. For example, nearly 30% content of the summaries in two popular new summarization benchmarks, CNNDM (Nallapati et al., 2016a) and NYT (Sandhaus, 2008b), is directly copied. As a consequence, directly employing extractive models can also achieve good results in summarization. Furthermore, the copy rate is even higher in other text generation tasks, such as Grammatical Error Correction (GEC) (Gu et al., 2016; Lichtarge et al., 2019), style transformation (Jin et al., 2022), etc. Unfortunately, existing text generation methods generate target sequences through word-by-word decoding (Lewis et al., 2019; Zhang et al., 2020; Liu et al., 2022), which may lead to factual inconsistencies (Cao et al., 2018) and slow inference. Meanwhile, extractive methods generate target sequences by re-organising some extracted vital text spans (Liu, 2019; Xu et al., 2019) that are factual consistent with input sequences, also faster but less flexible than generative methods.

As a consequence, summarization researchers try to add some extractive properties into the abstractive summarization, such as copying tokens (See et al., 2017) or entities (Xiao and Carenini, 2022). However, these text spans are still too short to represent complete semantics. As EDUs are outstanding extraction units (Wu et al., 2022), we propose to introduce copy mechanism to copy-oriented text generation tasks. Therefore, we propose an EDU-based copy mechanism (EDUCopy) that combines extractive and generative methods.

Taking text summarization as an example, seen from Table 1, we first use an extractive model to select some critical EDUs and enclose them in the original text using index markers, forming EDU-Source. Then, we replace the spans in the Target that directly copy EDUs from the source text with the corresponding identifiers, getting EDU-Target. After that, we finetune existing generative models using the processed (EDU-Source, EDU-Target) pairs. During testing, we restore the generated index tags to their respective text spans. Since EDUCopy only modifies the input/output text, it can apply to any generative model.

To verify the effectiveness of EDUCopy, we conduct experiments on the news summarization datasets CNNDM (Nallapati et al., 2016a), NYT (Sandhaus, 2008b) and the GEC datasets FCE (Yannakoudakis et al., 2011), WI-LOCNESS (Bryant et al., 2019a). While achieving notable ROUGE (Lin, 2004) and M^2 (Dahlmeier et al., 2013) scores, GPT-4 (OpenAI, 2023) evaluation validates the strengths of EDUCopy in terms of factual consistency, fluency, and overall performance. Moreover, compared with the baseline models, EDUCopy achieves a significant speedup by 1.65x in summarization and 1.24x in GEC, respectively. We mainly have the following

* Corresponding Author

LREC-COLING 2024, pages 8768–8780

EDU-Source: ... In 2014, one expert predicted consumers would pay more for some groceries <edu4> because of the California drought . </edu4> He was often right, according to statistics gathered by Timothy Richards, agribusiness professor at Arizona State University . Prices rose last year for these items on your kitchen table: . • Berries rose in price by about 80 cents per clamshell to $3.88 . • Broccoli by 11 cents per pound to $1.89 ... <edu5> overall prices are expected to rise this year, </edu5> because of inflation, U.S. Department of Agriculture economist Annemarie Kuhns said ...
Target: Americans paid more for some fruits and vegetables last year because of the drought . Tourists will now have to ask for a glass of water at a California restaurant . Perhaps the only good thing is another g̈reatẅine grape harvest last year.
EDU-Target: Americans paid more for some fruits and vegetables last year <edu4> Tourists will now have to ask for a glass of water at a California restaurant . Perhaps the only good thing is another g̈reatẅine grape harvest last year.
EDU-Output: The drought in California has had a ripple effect on other states in the West and Southwest, causing higher food prices over the past year. However, the USDA predicts that overall prices will continue to rise this year, partly <edu4>
Output: The drought in California has had a ripple effect on other states in the West and Southwest, causing higher food prices over the past year. However, the USDA predicts that overall prices will continue to rise this year, partly because of the California drought .

Table 1: A data sample to illustrate the pipeline of EDUCopy. The blue font in the table represents the copy part. EDU-Source and EDU-Target constitute the source-target pairs used for training the generative model. In testing, the generative model generates EDU-Output where the tag "<edu4>" will be mapped back to the highlighted section in the EDU-Source finally.

contributions:

- We combine extractive and generative methods by modeling the EDU copy behavior during generation.
- The proposed framework can be adapted to any generative model.
- Many evaluation metrics have confirmed the effectiveness and efficiency of our method, especially in the summarization domain, where EDUCopy surpasses human-level summarization.

2. Related Work

2.1. Natural Language Generation

Natural language generation, as the core task of NLP (Chen et al., 2019), has been studied for many years. Existing text generation models mainly adopts the paradigm of pre-training and fine-tuning, demonstrating considerable performance on multiple NLG tasks. Specifically, a typical pre-trained generative model, such as BART (Lewis et al., 2019) and T5 (Raffel et al., 2020), is trained by maximizing the probability of the next target token given the input text and the labeled text prefix. However, because labels are unknown during inference, the model inevitably makes subsequent decisions based on prior outputs that might be flawed. To address this problem, discriminative reranking was proposed and widely used in various NLG tasks (Shen et al., 2004; Wan et al., 2015). For example, (Mizumoto and Matsumoto, 2016) proposes a reranking method for GEC. Specifically, this reranking method is used to rescore the N best results of statistical machine translation and reorder the results. Moreover, in summarization, BRIO (Liu et al., 2022) improves the training by assuming a non-deterministic target distribution so that different candidate summaries are assigned probability mass according to their quality.

2.2. Copy Mechanism

In the aforementioned classic generative models, an input-output mapping within a fixed-size vocabulary will be learned. However, such an approach may encounter difficulties when there is crucial information or proper noun in the input text. To alleviate this problem, researchers try to combine extractive and generative method. For example, (See et al., 2017) proposes to use copy mechanism by directly copying these crucial information and proper noun. (Hsu et al., 2018) incorporates sentence-level attention to identify important sentences and uses inconsistency loss to encourage the generated summaries to be factual consistent. In addition to these verbatim decoding methods, there has been some works introducing sequential copying mechanisms. For instance, (Liu et al., 2021b) tags the target sequence in the BIO format to determine the extraction span, while (Xiao and Carenini, 2022) uses entities as copy units.

2.3. EDU-level Extractive Summarization

Our inspiration mainly comes from extractive summarization, where most conventional works are based on sentence-level extraction (Liu, 2019; Zhong et al., 2020; Liu et al., 2021a; Ruan et al., 2022). Recently, as EDU is proven to be a more concise semantic unit than sentence, some researchers use EDU as extraction unit (Alonso i Alemany and Fuentes Fort, 2003; Yoshida et al., 2014). (Liu and Chen, 2019; Huang and Kurohashi, 2021) validates the effectiveness of EDU extraction on large datasets. DISCOBERT (Xu et al., 2019) using structural discourse graphs based on

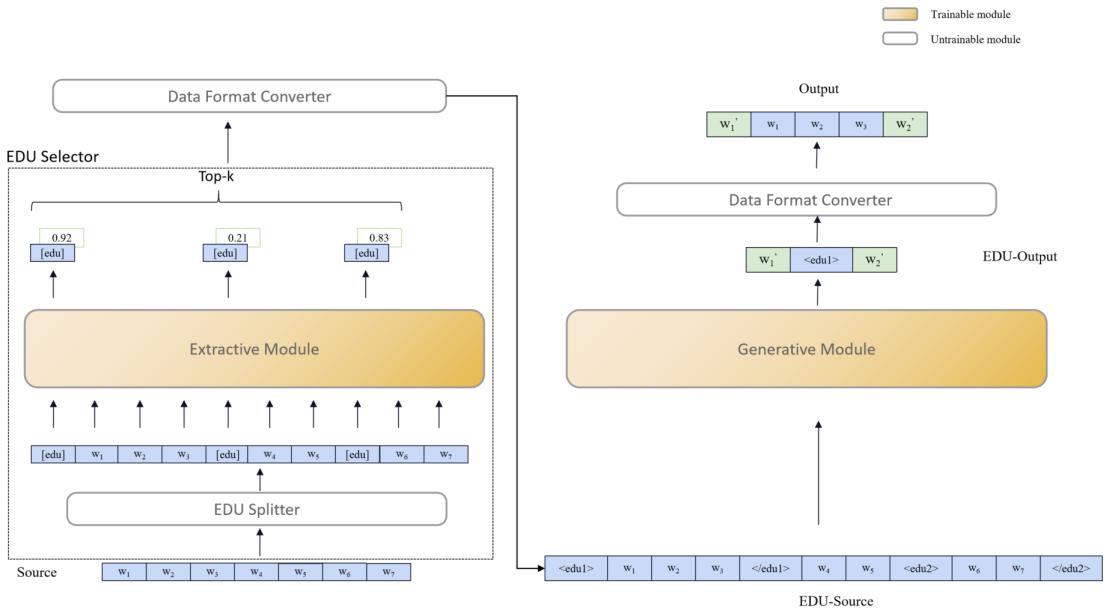

Figure 1: Flowchart of our framework. Firstly, the source is divided into EDUs and then put into the extractive module to pick up the top k (k=2 in the figure) EDUs. Then we use index tags to wrap these EDUs in the source to form the EDU-Source. Next, the generative module reads the EDU-Source and generates the EDU-Output. In the end, we restore the EDU index tags in the EDU-Output to construct the final output.

RST trees, co-referential references, and graph convolutional network encoding to capture these long-range dependencies. (Li et al., 2020) introduces a two-step method that involves selecting and grouping informative EDUs, and then fusing them into a coherent summary sentence, enhancing the generative ability of the model in terms of readability and non-redundancy. Moreover, (Wu et al., 2022) proves that EDU is a more suitable unit for extractive summarization from both theoretical and experimental aspects. Considering the success of previous works, our proposed framework, EDUCopy, also use EDU as the extraction unit.

3. Method

In this section, we propose a detailed description of our proposed framework, EDUCopy. Our motivation stems from the observation that in certain summarization datasets, approximately 30% of the golden summaries is directly copied from source EDUs. Additionally, in tasks like GEC and style transfer, copy behavior is more frequent. Consequently, we propose incorporating the behavior of copying EDUs into existing generative models. The flow chart of EDUCopy is shown in Figure 1. Specifically, EDUCopy consists of three modules, an EDU selector, a data format converter, and an generative module. The EDU selector extracts important EDUs. Then, the extracted EDUs will be enclosed by index tags to form the EDU-Source using the data format converter. After that, the generative module reads the EDU-Source and generates the sequence with the index tags, called EDU-Output. Ultimately, the data format converter restores the index tags in the EDU-Output to source's corresponding text. Since EDUCopy only modifies the input/output texts, it can apply to any generative approach. It is worth mentioning that in GEC tasks, due to the short source input length, we remove the extraction module and directly use the generative module to generate the target sequence.

3.1. EDU Selector

The EDU Selector detects salient EDUs from the source sequence. It comprises two components: an EDU splitter and an extractive module. The EDU splitter segments the source sequence into EDUs. The extractive model is responsible for extracting important EDUs.

EDU Splitter Given a source sequence with tokens $[w_1, w_2, ..., w_n]$, we utilize the tool proposed by Yu et al. (2022) to segment the source sequence into multiple EDUs. We add a tag "[edu]" in front of each EDU in the source sequence. The extractive model will utilize this tag to predict the saliency score of an EDU.

Extractive Module A document contains a large number of unimportant EDUs, which may confuse the generation model. Therefore, we utilize the extractive model to select the top representative EDUs. Since any extractive approach can achieve this goal, we choose the SOTA model MATCHSUM (Zhong et al., 2020) as our extractive module. MATCHSUM is a candidate sequence reranking approach that consists of two pre-trained transformer encoders: the EDU-encoder and Pair-encoder. The EDU-encoder is responsible for calculating the importance of each EDU. Specifically, it selects the top k most important EDUs and includes them in the set E in the order they appear in the source sequence. Then, starting from the EDU closest to the end, EDUs are sequentially removed in E to obtain k sets $E_k, E_{k-1}, ..., E_1$. Finally, the Pair-encoder reads the k (source sequence, EDU set) pairs and picks up the EDU set that is semantically closest to the document as our extraction result $E_m, m \in [1, k]$.

To train MATCHSUM, we label each EDU with a ground truth saliency score g^E based on its ROUGE (Lin, 2004) scores:

$$g^E = R_{1_p} + 2 * R_{2_p} + R_{L_p} \quad (1)$$

where $R_{1_p}, R_{2_p}, R_{L_p}$ means ROUGE-1/2/L precisions for short. Likewise, each EDU set is then scored based on the ground truth summary using the formula:

$$g^S = R_{1_f} + 2 * R_{2_f} + R_{L_f} \quad (2)$$

where we use the ROUGE F1 score. g^E is the learning objective of the EDU-encoder, while g^S is the learning goal of the Pair-encoder.

3.2. Data Format Converter

The data format converter offers multiple modes for EDU-text transformation, including:

1. EDU-Source construction: source+EDUs $\longrightarrow$ EDU-Source.
2. Output EDU-Output reconstruction during testing: EDU-Output + EDU-Source $\longrightarrow$ Output.
3. EDU-Target construction during training: Target + EDUs $\longrightarrow$ EDU-Target.

Given the EDU set extracted by the EDU selector, Mode 1 embraces an EDU with the index tag (<edu$i$>EDU text span</edu$i$>) in the source to form the EDU-Source. During testing, Mode 2 recovers the index tags in the EDU-Output from the input EDU-Source to construct the final Output. These two modes use simple rules shown in Figure 1. In contrast, Mode 3 is the core step in EDUCopy, reflecting the EDU copy behavior in text generation.

Algorithm 1: Target Conversion

Input : Golden Target S, EDU set $[e_1, ..., e_m]$.
Output Golden EDU-Target S^E
:

```
for e_i in [e_1, ..., e_m] do
    U_i = ∅
    for p in Range(|S|) do
        if S_p not in e_i[: 3] :
            continue
        for q in Range(p, |S|) do
            if S_q in e_i[−3 :] and
              R_{L_f}(S_pq, e_i) >= 0.7 :
                U_i.add(S_pq)
            else:
                continue
        end
    end
    U_i = Rule-Check(U_i, e_i)
    for s in Sorted(U_i,key=len,reverse=True) do
        S = S.replace(s,<edui>)
    end
end
S^E = S
Return S^E
```

EDU-Target Construction During the construction of the EDU-Target, we determine whether a segment is considered a copy based on the ROUGE score between the selected EDUs and the golden target fragments. The specific process is described in Algorithm 1. Let's assume that the extracted EDU set contains m EDUs $E_m = [e_1, e_2, ..., e_m]$. To construct the EDU-Target, we iterate through each element e_i in E_m and go through all the fragments in the golden target. We calculate the ROUGE-L F1 score between each fragment and e_i. If the score exceeds the Copy Threshold, we add the fragment to the copy candidate set U_i. After the traversal is complete, we apply certain rules to remove strings from set U_i that break the coherence. Assuming U and e represent the preliminary selected segment set and the EDU to be copied, respectively, with elements u in U, the specific details of the filtering process are as follows:

- The middle of u must not contain punctuation marks.
- Entities in u must also be found in e.
- If the end of e is a punctuation mark, then the end of u must be consistent with e.

After applying the filtering rules, we sort the strings in each set U_i in descending order based on their lengths. Finally, we iteratively replace these strings in the golden target with EDU index tags to get the golden EDU-target.

Table 2 shows how the change in Copy Threshold affects EDU-summaries' performance. We find that

Dataset	Copy Threshold	SumLen	EDU-SumLen	Copy Rate	ROUGE R-1	ROUGE R-2	ROUGE R-L
CNNDM	0.6	57.87	44.02	40.53%	90.60	82.48	90.40
	0.7	57.87	48.24	29.11%	95.71	91.22	95.63
	0.8	57.87	51.32	21.13%	98.14	95.76	98.12
	0.9	57.87	53.65	12.24%	99.70	99.27	99.70
NYT	0.6	117.90	83.59	43.99%	91.80	83.66	91.17
	0.7	117.90	90.62	35.89%	95.33	89.55	94.99
	0.8	117.90	97.93	27.42%	97.53	93.87	97.35
	0.9	117.90	107.20	14.41%	99.41	98.41	99.36

Table 2: Analysis of summary copy behavior based on setting different copy thresholds. Here we use the golden EDU set for candidates of copying. R-1, R-2, R-L are abbreviations for the F1 score of ROUGE (1/2/L). We use the data format converter to get the restored summary from the EDU-summary, and calculate its ROUGE score measured by the original golden summary.

with the decrease of Copy Threshold, the length of EDU-summaries drops obviously while the ROUGE scores keep high. We ultimately set it to 0.7 in the experiments. During the training phase, the labels are visible, and we can extract EDUs based on these labels. This way, the extracted EDUs are the most suitable. However, during the testing phase, where the labels are not visible, we must rely on the results extracted by the extraction module. Therefore, when training the generative module, We always use the EDUs selected by the extractive module to reduce the discrepancy between training and testing.

3.3. Generative Module

The generative module primarily learns the mapping from EDU-Source to the EDU-Target. Any generative model can be integrated into the EDUCopy framework, and in this paper, we use the state-of-the-art summarization model BRIO (Liu et al., 2022) as the generative module. It involves two training processes: maximum likelihood training and sequence calibration.

In maximum likelihood training, we fine-tune a backbone seq2seq model on the (EDU-Source, EDU-Target) pairs. The learning goal is to maximize the likelihood of the EDU-Target generation. Subsequently, we use the generative model to build 16 EDU-Output for each document in the training set. A contrastive loss is introduced to align the generation probability of each EDU-Target and its actual quality measured by ROUGE. Specifically, we use Ranking loss to bring closer those EDU-Outputs among the generated 16 which have higher scores compared to the Target. Conversely, those dissimilar to the Target are pushed further away in the probability output distribution. This training method originates from the BRIO model. In this way, the generative model is guided to assign probability mass according to the qualities of generated targets.

The generative module generates an EDU-Output containing EDU index tags linked to the source sequence. During testing, we use the data format converter to recover these index tags into source text spans, yielding the final sequence. Since EDU-targets are about 20% shorter than the original target, our generative module runs much faster than general generative approaches.

4. Experiments

4.1. Datasets

To verify the effectiveness of EDUCopy, we conduct experiments on two popular summarization benchmarks, CNNDM (Nallapati et al., 2016a) and NYT (Sandhaus, 2008b). Since EDUCopy can apply to any seq2seq text generation tasks, we also conduct experiments on GEC datasets, FCE (Yannakoudakis et al., 2011) and WI-LOCNESS (Bryant et al., 2019a). The basic information of these datasets is shown in Table 3.

CNNDM CNNDM is a commonly used news summarization dataset, which includes news articles and corresponding multi-sentence highlights. Moreover, we used a pre-processed version of the dataset obtained from (See et al., 2017).

NYT NYT consists of articles from the New York Times and corresponding summaries. We followed the approach outlined by (Kedzie et al., 2018) to preprocess and split the data. For the summaries, we used the archival abstracts associated with each article.

FCE FCE is a part of the Cambridge Learner Corpus (CLC), containing about 30,995 parallel sentences for training and approximately 2,691 parallel sentences for testing.

WI-LOCNESS WI-LOCNESS originates from the GEC segment of the Building Educational Applications 2019 Shared Task (Bryant et al., 2019b). It is composed of two separate datasets: LOCNESS and the Cambridge English Write&Improve (W&I).

Datasets	Train	Valid	Test
CNNDM	287K	13K	11K
NYT	44K	5.5K	6.4K
FCE	26.6	2.0K	2.5K
WI	44K	-	4.2K

Table 3: Dataset Statistics.

4.2. Baselines

We select the following models as our baselines: **BRIO** (Liu et al., 2022) utilizes pre-trained models **BART** (Lewis et al., 2019) and **PEGASUS** (Zhang et al., 2020) as backbones to generate high-quality sequence through sequence correction. **SpanCopy** (Xiao and Carenini, 2022) is a network architecture that enables copying entities from the source text. Its copy mechanism is similar to EDUCopy, allowing the model to reproduce specific spans of text from the input. Besides these abstractive models, we also introduce some EDU-based extractive models, including **EDU-VL** (Wu et al., 2022) and our extractive module **EDU Selector**.

In addition to these models, we include the Large Language Model (LLM) **ChatGPT** as one of the baselines. ChatGPT is capable of generating responses that align with human preferences, which makes it a suitable candidate for comparison in our study.

4.3. Evaluation Metrics

For the experiments conducted on the summarzation datasets, we employ the widely adopted ROUGE (Lin, 2004) to evaluate the models automatically. Specifically, we report three ROUGE metrics, ROUGE-1, ROUGE-2, and ROUGE-L. In addition, LLMs such as GPT-4 (OpenAI, 2023) can align with human evaluation standards (Ouyang et al., 2022; Gao et al., 2023) and is highly consistent with human evaluation results (Li et al., 2023). Following their work, we employ GPT-4 as a means of overall summary quality evaluation. To enhance the stability of GPT-4 evaluation, following Li et al. (2023), we design specialized prompts to conduct the evaluation, as shown in appendix. Additionally, we also use an entity-level factual consistency metric introduced by SpanCopy (Xiao and Carenini, 2022). As exhibited in Equation 3, this metric measures generated summaries' entity coverage score compared to the source sequence.

As for the experiments conducted on GEC, we use M^2 (Dahlmeier et al., 2013) as the evaluation metric. The M^2 metric evaluates GEC by comparing system outputs to multiple correct annotations. It emphasizes accuracy using the F0.5 score to weigh precision over recall. Meanwhile, we also employ GPT-4 to compare the correction performance of EDUCopy with baseline models.

$$DOC_p = |NE(D) \cap NE(T)|/|NE(T)| \quad (3)$$

where $NE(*)$ stands for entity set of $*$, T stands for the generated target sequence, and D stands for the input document.

As one main advantage of EDUCopy is to speed up the summary generation process, we also list the inference time of EDUCopy and baseline models for comparison. Finally, due to the cumbersome and costly nature of human evaluation methods, we conducted manual evaluation on only 100 randomly sampled data examples from CNNDM.

4.4. Setting

We use the same hyperparameters as the original models (MATCHSUM, BRIO) in the extractive and generative modules. We use Adam (Kingma and Ba, 2014) as the optimizer with a learning rate 2e-5 and employ a cosine learning rate decay strategy. Furthermore, we achieve an early stopping strategy during inference using the F1-score of the overlap rate of the predicted EDU set and the label EDU set as the criterion. The training of EDUCopy is conducted on 4 GeForce RTX 3090 (24GB) with a total batch size of 16 for BART and 8 for PEGASUS. Because of the limitation of backbone models, we truncate the documents if they exceed the maximum input length of 1024. The scripts [1] for ROUGE scores calculation and candidate summaries generation are obtained from BRIO. As for the calculation of M^2, we use the implementation provided by errant (Bryant et al., 2017; Felice et al., 2016). As for the experiments conducted on FEC and WI-LOCNESS, we use NLTK (Bird et al., 2009) to split the original data into sentences, and each sentence was used as a data sample.

4.5. Result

Overall Evaluation To evaluate the overall performance of models, we conduct ROUGE evaluation on CNNDM and NYT datasets. The results are shown in Table 4. As can be seen, EDUCopy outperforms most other methods on ROUGE evaluation. Only BRIO is slightly better than EDUCopy but requires significantly longer inference time. Since Wang et al. (2023a); Yang et al. (2023) proves that ROUGE evaluation is insufficient for the overall evaluation of summarization models, we focus on using

[1] https://github.com/yixinL7/BRIO

Dataset	Model	ROUGE R-1	ROUGE R-2	ROUGE R-L	Time/s	DOC_p
CNNDM	ChatGPT	35.05	13.04	31.59	-	84.03
	BART	44.01	20.83	40.75	17,676	91.22
	PEGASUS	44.22	21.31	41.31	17,673	90.08
	EDU-VL*	44.70	21.63	42.46	-	-
	SpanCopy*	44.19	20.86	31.19	-	91.89
	EDU Selector	43.23	20.62	41.54	892	98.48
	BART-BRIO	47.72	23.57	44.39	17,059	93.11
	PEGASUS-BRIO	**47.80**	22.95	**44.46**	18,375	92.87
	BART-EDUCopy	46.39	23.12	43.55	12,099	**95.90**
	PEGASUS-EDUCopy	46.23	**23.62**	43.42	**11,134**	95.62
NYT	ChatGPT	40.50	15.32	29.47	-	75.74
	EDU Selector	51.58	31.69	41.12	753	99.18
	BART	53.97	34.99	41.80	14,716	82.53
	BART-BRIO	**56.61**	**37.54**	44.19	16,995	82.54
	BART-EDUCopy	56.37	36.93	**44.38**	**13,580**	**84.07**

Table 4: Experimental results on CNNDM and NYT. * Represents the quoted experimental results.

Dataset	Model	M^2	Time	win rate
FCE	BRIO	60	1910	44.89%
	EDUCopy	62.24	1638	55.11%
WI	BRIO	52.15	1042	43.12%
	EDUCopy	55.13	842	56.88%

Table 5: Experimental results on FCE and WI. Win rate is the quality evaluation result of GPT4 comparing BRIO and EDUCopy error correction results.

GPT-4 to evaluate overall performance. Specifically, we compare golden summaries, BART-BRIO (**BRIO**) and BART-EDUCopy (**EDUCopy**) in Table 6. The results show that EDUCopy significantly beats BRIO and even human-written golden summaries. At the same time, it can be seen from Table 5 that the saliency index and gpt evaluation of the GEC task are highly consistent with the summarization results. Therefore, EDUCopy can be used in any text conversion task with a high copy rate. We show a data sample in Table 11, and more samples in the appendix.

Dataset	**Rank**	**Golden**	**BRIO**	**EDUCopy**
CNNDM	1	2724	1770	**5086**
	2	3184	2938	3458
	3	3672	4872	1036
NYT	1	1707	1239	**2532**
	2	2450	1641	1387
	3	1321	2598	1559

Table 6: The overall evaluation results of GPT-4. The data in the table represent the number of times the summary gets the 1, 2, 3 rank.

Factual Consistency Evaluation Intuitively, EDUCopy can alleviate the problem of factual inconsistency by copying the EDUs in the original text. To verify this conjecture, we use GPT-4 to evaluate models from the perspective of factual consistency. Shown in Table 7, EDUCopy behaves more faithfully than baselines. The entity evaluation metric DOC_p shown in Table 4 also points out that our method exhibits the highest overlap rate with entity sets in the document.

Dataset	Win Rate BRIO	Win Rate EDUCopy
CNNDM	44.13	**55.87**
NYT	43.61	**56.39**

Table 7: The results of GPT-4 evaluation on factual consistency.

Efficiency Evaluation compared with baseline models, EDUCopy achieves a significant speedup by 1.65x in summarization and 1.24x in GEC, respectively. This indicates that copying EDUs makes it possible to reduce the output length and improve the efficiency of the generation process. We provide the specific time expenses of each component of EDUCopy during inference in Table 8. As can be seen, the time spent on the additional data processing in EDUCopy is trivial compared with the generation cost.

Coherence Evaluation During our experimental process, we also had concerns about the fluency of sentences. Because we used a rule-based approach to find matching EDUs during training, we cannot always guarantee that the generated

Dataset	Splitter	Extraction	Abstraction
CNNDM	742	892	10465
NYT	804	753	12023
FCE	58	-	785
WI-LOCNESS	101	-	1537

Table 8: Time cost (/s) of different modules in EDUCopy.

EDU-summary is fluent. In fact, during the human evaluation, we found that the architecture using EDUcopy is smoother and better at capturing details in sentences, such as names, locations, specific events, and so on. Compared with BRIO, we had an 62% win rate. This is a very good result. In such cases, our understanding is that for a span, the word-by-word decoding approach to generate the span requires multiple conditional probability decisions, while generating an EDU only requires a single decision. Regarding the decoding difficulty between these two methods, it is not necessarily the case that EDU decoding is more challenging.

4.6. Selection of k and Extraction Method

To investigate the impact of different k values, we first extracted the distribution of the number of EDUs, as shown in Figure 2. Even in news summarization tasks with relatively longer target sequences, the number of copied EDUs in most documents is basically less than ten. During the exploration process, we initially set k to a larger value, such as 50. We then use BERTSUM to extract k EDUs for each source, followed by applying MatchSUM to eliminate similar EDUs among the k extracted. Subsequently, we analyzed the final number of EDUs extracted from each data sample, as shown in Figure 2. It can be observed that when k is relatively small, such as $k = 2$, each data sample has at most two replicable EDUs. However, in fact, there are still many EDUs in these data that have not been identified as replicable units. When it becomes necessary to generate these unrecognized EDUs, decoding must be done character by character rather than directly generating EDUs. This goes against our experimental motivation. Therefore, for these copy-oriented natural language generation tasks, we need to set k around 10. In addition, to confirm that the gain effect is not particularly significant when k is greater than a certain value, we compared the effects of k=10 and k=15. The results in Table 9 show that EDUCopy is stable within the predictable range of k selection. In conclusion, we can easily choose k after measuring the distribution of EDU numbers, but this does not mean that k can be arbitrarily large, which will increase the pressure on the EDU selector.

Besides, to test the effect of the extractive

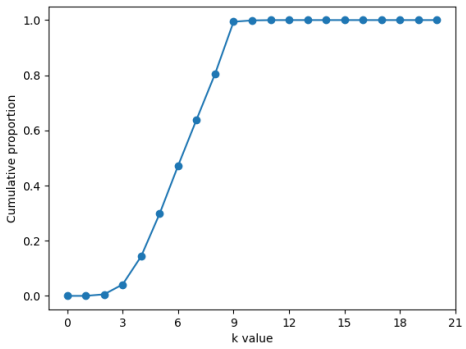

Figure 2: The cumulative distribution of the EDU number extracted by the EDU Selector. k represents the number of extracted EDUs, and the values on the y-axis represent the cumulative proportion of data whose number of extracted EDUs is not greater than k. It can be seen that the number of extracted EDUs of each data lies between 1 to 10.

k	ROUGE		
	R-1	R-2	R-L
10	46.39	23.12	43.55
15	46.23	23.62	43.42

Table 9: Results of EDUCopy on CNNDM with different k values.

method on EDUCopy, we compare the EDU selector with BERTSUM (Liu, 2019) and use the top 10 salient EDU measured by the golden summary as the upper bound. The results shown in Table 10 reveals that the performance of EDUCopy is positively correlated with the performance of the extractive model. This means that in the future, when better extractive models emerge, EDUCopy will also show better performance in text generation.

Model	ROUGE		
	R-1	R-2	R-L
BertSum	45.33	21.79	42.29
EDU-Selector	46.39	23.12	43.55
Golden	55.12	32.78	52.21

Table 10: Results of EDUCopy on CNNDM with different extractive methods.

4.7. Case Study

As shown in Table 11, we study a sample from CNNDM. This news tells the story of the football player Danny Ings. We compare the results from EDUCopy, BRIO, and the human-written summary.

We can find the golden summary to be a good generalization. But it is somewhat sketchy compared with EDUCopy's output. For example, the golden summary does not mention Ings being linked with transfers, a crucial message in the document. The summary generated by BRIO contains factual errors. The document does not mention the goal of Danny Ings collaborating with Borussia Monchengladbach and David Moyes. In contrast, EDUCopy produces a more coherent and detailed summary, mentioning clubs that Ings is associated with, which provides a comprehensive overview of the future for Ings.

Document: Manchester United and Liverpool target Danny Ings insists his aim for next season is to play and develop wherever he ends up. The Burnley striker's future has been the subject of considerable speculation with the 22-year-old also linked with moves to Borussia Monchengladbach and David Moyes' Real Sociedad. However, Ings - who has scored nine goals during his debut Premier League season - is keen to keep his career moving forward and does not want sit on the bench. Burnley striker Danny Ings insists he is aiming to play and develop wherever he ends up next season . Ings, who has scored nine Premier League goals, is a target for Manchester United and Liverpool ...'
Golden summary: Danny Ings is a target for Manchester United and Liverpool this summer . The Burnley striker does not want to move just to sit on the bench . Ings keen to work with a manager who will help him develop as a player .
EDUCopy: Burnley striker Danny Ings is a target for Manchester United and Liverpool. The 22-year-old insists his aim for next season is to play and develop wherever he ends up. Ings has been linked with moves to Borussia Monchengladbach and Real Sociedad.
BRIO: Danny Ings has scored nine Premier League goals for Burnley this season. The Burnley striker is a target for Manchester United and Liverpool. Ings insists that his goal for the next season is to work with Borussia Monchengladbach and David Moyes.

Table 11: A sample display of the generated results. blue highlighting represents EDUCopy faithfully copying the content of the original text. The red sections represent factual errors that deviate from the original text.

5. Conclusion and Future Work

In this paper, we propose EDUCopy, a novel framework that combines extractive method and generative method by incorporating the behavior of copying EDUs into the generation process. Since our framework only changes the input/output text formats, it can apply to any generative approach. To verify the effectiveness of EDUCopy, We conduct thoroughly experiments on multiple natural language generation tasks. While achieving notable ROUGE and M^2 scores, GPT-4 evaluation validates the strength of our models in terms of factual consistency, fluency, and overall performance. Moreover, compared to baseline models, EDUCopy achieves a significant acceleration of 1.65x.

We believe our work can be extended in many aspects. On the one hand, we plan to use the popular ChatGPT to choose EDUs to form more fluent EDU-Targets. On the other hand, we are curious about the combination of EDUcopy with large language models.

6. Acknowledgments

The work described in this paper was supported by the National Natural Science Foundation of China (NSFC 62106165) and a Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions.

7. Bibliographical References

Laura Alonso i Alemany and Maria Fuentes Fort. 2003. Integrating cohesion and coherence for automatic summarization. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 2*, pages 1–8.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Christopher Bryant, Mariano Felice, Øistein E Andersen, and Ted Briscoe. 2019a. The bea-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75.

Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. 2019b. The BEA-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75, Florence, Italy. Association for Computational Linguistics.

Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*

(Volume 1: Long Papers), pages 793–805, Vancouver, Canada. Association for Computational Linguistics.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the original: Fact aware neural abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang. 2019. Few-shot nlg with pre-trained language model. *arXiv preprint arXiv:1904.09521*.

Sangwoo Cho, Kaiqiang Song, Chen Li, Dong Yu, Hassan Foroosh, and Fei Liu. 2020. Better highlighting: Creating sub-sentence summary highlights. *arXiv preprint arXiv:2010.10566*.

Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 93–98.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner english: The nus corpus of learner english. In *Proceedings of the eighth workshop on innovative use of NLP for building educational applications*, pages 22–31.

Mariano Felice, Christopher Bryant, and Ted Briscoe. 2016. Automatic extraction of learner errors in ESL sentences using linguistically enhanced alignments. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 825–835, Osaka, Japan. The COLING 2016 Organizing Committee.

Mingqi Gao, Jie Ruan, Renliang Sun, Xunjian Yin, Shiping Yang, and Xiaojun Wan. 2023. Human-like summarization evaluation with chatgpt. *arXiv preprint arXiv:2304.02554*.

Tao Ge, Furu Wei, and Ming Zhou. 2018. Fluency boost learning and inference for neural grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1055–1065.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2003. English gigaword. *Linguistic Data Consortium, Philadelphia*, 4(1):34.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. *arXiv preprint arXiv:1805.06266*.

Yin Jou Huang and Sadao Kurohashi. 2021. Extractive summarization considering discourse and coreference relations based on heterogeneous graph. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3046–3052.

Di Jin, Zhijing Jin, Zhiting Hu, Olga Vechtomova, and Rada Mihalcea. 2022. Deep learning for text style transfer: A survey. *Computational Linguistics*, 48(1):155–205.

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

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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. *arXiv preprint arXiv:1910.13461*.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Alpacaeval: An automatic evaluator of instruction-following models. `https://github.com/tatsu-lab/alpaca_eval`.

Zhenwen Li, Wenhao Wu, and Sujian Li. 2020. Composing elementary discourse units in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6191–6196.

Jared Lichtarge, Chris Alberti, Shankar Kumar, Noam Shazeer, Niki Parmar, and Simon Tong. 2019. Corpora generation for grammatical error correction. *arXiv preprint arXiv:1904.05780*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

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

Yang Liu and Mirella Lapata. 2019. Hierarchical transformers for multi-document summarization. *arXiv preprint arXiv:1905.13164*.

Ye Liu, Jian-Guo Zhang, Yao Wan, Congying Xia, Lifang He, and Philip S Yu. 2021a. Hetformer: heterogeneous transformer with sparse attention for long-text extractive summarization. *arXiv preprint arXiv:2110.06388*.

Yi Liu, Guoan Zhang, Puning Yu, Jianlin Su, and Shengfeng Pan. 2021b. Biocopy: A plug-and-play span copy mechanism in seq2seq models. *arXiv preprint arXiv:2109.12533*.

Yixin Liu, Pengfei Liu, Dragomir Radev, and Graham Neubig. 2022. Brio: Bringing order to abstractive summarization. *arXiv preprint arXiv:2203.16804*.

Zhengyuan Liu and Nancy Chen. 2019. Exploiting discourse-level segmentation for extractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 116–121.

Tomoya Mizumoto and Yuji Matsumoto. 2016. Discriminative reranking for grammatical error correction with statistical machine translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1133–1138.

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

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

Feng Nan, Ramesh Nallapati, Zhiguo Wang, Cicero Nogueira dos Santos, Henghui Zhu, Dejiao Zhang, Kathleen McKeown, and Bing Xiang. 2021. Entity-level factual consistency of abstractive text summarization. *arXiv preprint arXiv:2102.09130*.

R OpenAI. 2023. Gpt-4 technical report. *arXiv*, pages 2303–08774.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Qian Ruan, Malte Ostendorff, and Georg Rehm. 2022. Histruct+: Improving extractive text summarization with hierarchical structure information. *arXiv preprint arXiv:2203.09629*.

Evan Sandhaus. 2008a. *The New York Times Annotated Corpus*. LDC corpora. Linguistic Data Consortium.

Evan Sandhaus. 2008b. The new york times annotated corpus LDC2008T19. Web Download.

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

Libin Shen, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative reranking for machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 177–184, Boston, Massachusetts, USA. Association for Computational Linguistics.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

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.

Xiaojun Wan, Ziqiang Cao, Furu Wei, Sujian Li, and Ming Zhou. 2015. Multi-document summarization via discriminative summary reranking. *arXiv preprint arXiv:1507.02062*.

Jiaan Wang, Yunlong Liang, Fandong Meng, Haoxiang Shi, Zhixu Li, Jinan Xu, Jianfeng Qu, and Jie Zhou. 2023a. Is chatgpt a good nlg evaluator? a preliminary study. *arXiv preprint arXiv:2303.04048*.

Peiyi Wang, Lei Li, Liang Chen, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. 2023b. Large language models are not fair evaluators. *arXiv preprint arXiv:2305.17926*.

Yuping Wu, Ching-Hsun Tseng, Jiayu Shang, Shengzhong Mao, Goran Nenadic, and Xiao-Jun Zeng. 2022. Edu-level extractive summarization with varying summary lengths. *arXiv preprint arXiv:2210.04029*.

Wen Xiao and Giuseppe Carenini. 2022. Entity-based spancopy for abstractive summarization to improve the factual consistency. *arXiv preprint arXiv:2209.03479*.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Discourse-aware neural extractive text summarization. *arXiv preprint arXiv:1910.14142*.

Divakar Yadav, Jalpa Desai, and Arun Kumar Yadav. 2022. Automatic text summarization methods: A comprehensive review. *arXiv preprint arXiv:2204.01849*.

Xianjun Yang, Yan Li, Xinlu Zhang, Haifeng Chen, and Wei Cheng. 2023. Exploring the limits of chatgpt for query or aspect-based text summarization. *arXiv preprint arXiv:2302.08081*.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading esol texts. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 180–189.

Yasuhisa Yoshida, Jun Suzuki, Tsutomu Hirao, and Masaaki Nagata. 2014. Dependency-based discourse parser for single-document summarization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1834–1839.

Nan Yu, Meishan Zhang, Guohong Fu, and Min Zhang. 2022. Rst discourse parsing with second-stage edu-level pre-training. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4269–4280.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B Hashimoto. 2023. Benchmarking large language models for news summarization. *arXiv preprint arXiv:2301.13848*.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. *arXiv preprint arXiv:2004.08795*.

Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. 2018. Sequential copying networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

A. Experiment Details

we show the templates we used for GPT4 evaluation in Tables 12, 13 and 14.

[System]
You are a helpful assistant.
[Question 1]
Please determine which of the following two sentences is free of grammatical errors, and select the more perfect one.
This is the first sentence:{S1}
This is the second sentence:{S2}
please analyze whether there are any grammatical errors among them.
[Start of LLM's first answer]
{R1}
[End of LLM's first answer]
[Question 2]
Now please tell me which sentence is more perfect. If it's the first one, please answer 0; if it's the second one, please answer 1. all you need to do now is reply with a number.
[Start of LLM's second answer]
{R2}
[End of LLM's second answer]

Table 12: We provided a template for assessing the grammatical quality of sentences. **S1**, **S2** serve as our inputs. **R1** and **R2** are responses from the LLM.

[System]
You are a helpful assistant.
[Question 1]
Determine which of the following two summaries is more in line with the description of the long text. I will give you a long text and two summaries.
This is the long text: {DOCUMENT}
This is the first summary:{SUM1}
This is the second summary:{SUM2}
Please analyze which parts of these two summaries are inconsistent with the description in the long text.
[Start of LLM's first answer]
{R1}
[End of LLM's first answer]
[Question 2]
Now you need to determine which summary is more factual. If the first summary is better, you can reply to me with "0". If the second summary is better, you can reply to me with "1". Of course, if you think these two summaries are similar, please reply to "2". all you need to do now is reply with a number.
[Start of LLM's second answer]
{R2}
[End of LLM's second answer]

Table 13: We provide a template for evaluating the factual consistency of summaries on LLM. **DOCUMENT**, **SUM1**, and **SUM2** serve as our inputs. **R1** is the basis for LLM's evaluation of summary quality, and **R2** is the conclusion.

[System]
You are a helpful assistant.
[Question 1]
Please be a summary evaluation expert. I will provide you with a long text and three summaries and let you judge the ranking of the quality of these three summaries. This is the long text: {DOCUMENT}
This is the first summary: {REFERENCE}
This is the second summary:{SUM1}
This is the third summary:{SUM2}
Please briefly analyze the advantages and disadvantages of these three.
[Start of LLM's first answer]
{R1}
[End of LLM's first answer]
[Question 2]
Now you need to tell me their ranking and reply with a list. If the first summary is the best, the second summary is the second, and the third summary is the worst, please reply to me [2,1,0]. Similarly, if the second summary is the best, the first summary is the second, and the third summary is the worst, you need to reply to me [1,2,0].
[Start of LLM's second answer]
{R2}
[End of LLM's second answer]

Table 14: We have provided a template for evaluating the quality of summaries on LLM. **DOCUMENT**, **REFERENCE**, **SUM1**, and **SUM2** serve as our inputs. **R1** is the basis for LLM's evaluation of summary quality, and **R2** is the conclusion.