GSAC: Improving Multi-Document Summarization with Graph Structure-Aware Encoder

Anonymous ACL submission

Abstract

Sequence-to-sequence neural networks have achieved remarkable success in abstractive text summarization. However, current models may not be directly adaptable to the task of multi-document summarization (MDS). In this paper, we propose a neural summarization framework that can effectively process lengthy texts and multiple input documents. We propose a method to seamlessly integrate graph representations into the encoder-decoder model. Additionally, we introduce an extra training objective aimed at maximizing the similarity between the compressed graph text and the ground-truth summary at the node level. Our approach utilizes an innovative method for constructing text graphs to tackle the challenges of applying graph structures in multi-document scenarios. With a base PRIMERA model, our method shows superior performance compared to previous state-of-the-art models on the Multi-news, Multi-XScience and Wikisum datasets.

1 Introduction

In recent years, the advancement of natural language processing (NLP) has generated significant interest in the processing of lengthy texts. Long texts play a significant role in conveying information, including government documents and medical reports. The narratives in lengthy texts often span hundreds or thousands of words, covering a wide array of topics. This presents a challenge for neural summarizers in identifying the main themes (Ma et al., 2022), which increases the risk of omitting important information in the generated summaries. To address this issue, graphs have been widely used as they have proven to be effective in capturing the intricate relationships within long texts. The significance of graphs in text summarization was first proposed by (Erkan and Radev, 2004). They emphasized the importance of graphs in the summarization process. Expanding on this concept, (Li et al., 2020b) proposed integrating the graph into the encoding stage of summarization.They utilized the graph structure to enhance the representation of the input text, resulting in improved summarization performance.

However, employing graph-based methods in multi-document summarization poses a series of challenges. The length of multi-document text makes it difficult to extract graph structures. Typically, multi-text consists of multiple parallel documents, such as in Multi-News where several news articles cover the same event. While these articles often have complete narrative structures, more detailed information needs to be supplemented in various ways. Studies have shown that multiple pieces of coarse-grained information can be integrated to create a summary (Zhang et al., 2022). (Liu et al., 2021) offers a method that a summary can be created by consolidating overlapping key information from various documents or paragraphs. Additionally, it has been noted that not all input texts are consumed by the summarization model due to input limitations. Rearranging the order of text combinations can effectively improve the performance of the summary. However, this approach is difficult to apply to complex structures and longer texts. Furthermore, the fixed receptive field size, combined with text recombination, weakens the model's contextual consistency.

In this study, we propose GSAC, an efficient framework for multi-document summarization using graph-text representation. Unlike the methods proposed by (Pasunuru et al., 2021a), we optimize the graph representation of lengthy texts by introducing subgraph pruning and duplicate data removal techniques. By leveraging the search clustering of text graphs, our model can extract essential information from lengthy texts while keeping the input length within an acceptable range. This eliminates the need for context expansion.In contrast to the PRIMERA model, we also propose a

new method for extracting text graph representations. By utilizing a multi-stage search clustering approach, our model can significantly enhance text coverage for multi-document scenarios and eliminate outlier errors in lengthy text files, thereby ensuring the adequacy and effectiveness of information. The applied graph-structure supervision method also demonstrates improved performance in maintaining structural consistency between summaries and source texts.Furthermore, we integrate structural information into the text encoding process, enabling the graph structure to explicitly influence the text weights. Our model includes hierarchical encoding layers, allowing it to process both modalities of information and perform deep fusion encoding during the joint encoding phase without requiring additional encoder structures. This results in a more balanced decoding process and enhanced summarization performance.

2 Related Work

Multi-Document Summarization Abstrictive Multi-Document Summarization aim to generate concise summaries given a set of similar documents related to the same topic has been studied for a long time. Previous approachs mostly focus on extracting salient contents from source documents (Radev, 2004) (Wan and Yang, 2006) (Wan, 2008), or use these extractions for sequence-to-sequence models to generate abstractive summary (Song et al., 2022) (Tu and Nie., 2022).However, these approach essentially treats the MDS task as an excessively long SDS task, and neglects the complex cross-document relations. Many MDS studies consider optimizing attention mechanisms, proposing local attention (Beltagy et al., 2020) that means each token can only attend to information from its neighboring tokens and hierarchical attnetion (Yang et al., 2016) that focus on different level structure of documents. Our method falls into the family of local attention variants and extends existing local attention with cross position global attention to enable better preserving cross-textual information.

PLMs for MDS With the tremendous success of pre-trained language models that follow a Transformer-based (Vaswani et al., 2017) encoder-decoder architecture, fine-tuning/re-training PLMs in smaller-sized or with other domains datasets has become the primary paradigm for abstractive multi-document summarization. However, they use a flat concatenation (Guo et al., 2021) PRIMERA model and face the same limitations of modeling intricate cross-document relations as discussed previously. In order to sufficiently leverage the pre-training knowledge in PLM, others commonly initialize layers of their architecture with the PLM weights. PRIMERA (Xiao et al., 2021) propose pyramid-based entity masked scheme into the pre-training, where the overlapping entities are used to select out salient sentences for pre-training. But re-training PLMs with smaller-sized text summarization corpus may introduce new conflicts and lead to catastrophic destruction of the previously pre-trained knowledge.

Graph Text MDS Previous graph text MDS approaches are extractive, which extract salient discourse units from documents based on representations of phrases. ATTOrderNet (Yin et al., 2019) propose a graph text neural model to sort sentences by exploiting entity linking graph to measure the cross-relations between sentences. However, there are only a handful of proposed models using graphs to encode documents in abstractive MDS (Li and Zhuge, 2021) (Cui and Hu, 2021). Most of these models only leverage homogeneous graphs as they do not consider different edge types of graphs. For example, GraphSum (Li et al., 2020a) introduces similarity graphs over the documents. Graphs constructed in these models are indeed homogeneous. Unlike to these works only consider the graph structure of source documents, our work focus on the impact of graph spatial position structure on semantic consistency level.

3 Method

In this section, we initially introduce pre-trained Transformer Encoder-Decoder architecture models as our baseline model. To better adapt to MDS, we introduce Graph Text to capture the textual structure of multi-documents and combine a non-pretrained graph encoder to effectively contextualize semantic information in the input context. Subsequently, we incorporate a document-level attention mechanism in the model's decoder to focus on the differences existing among the documents. Specifically, given a set of source documents $\mathcal{D} = \{d_1, d_2, \cdots, d_n\}$, where n is the number of documents.We aim to generate a summary S of the document cluster.

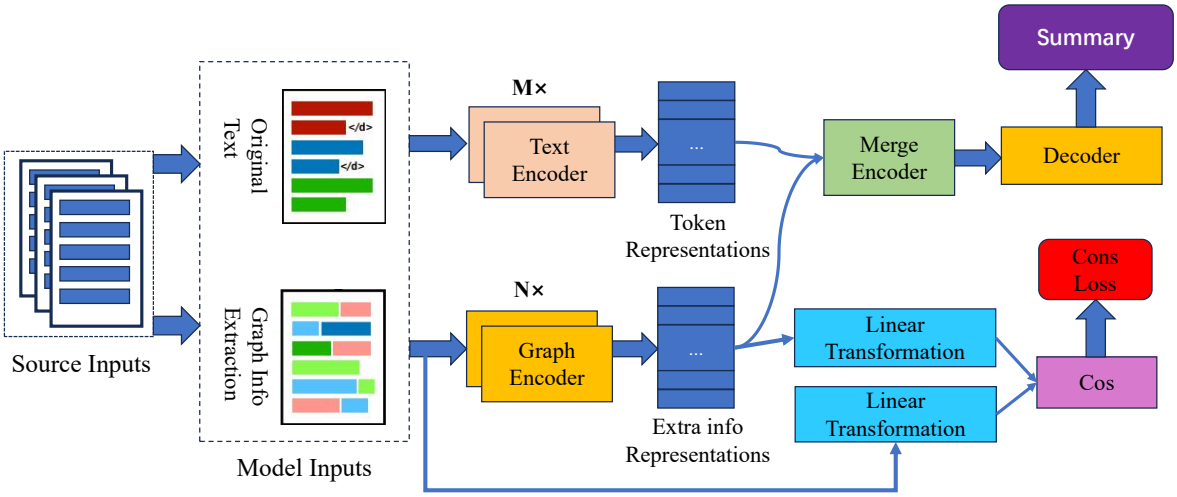

Figure 1: The architecture of the GSAC model illustrates its input, which comprises a set of texts. The model then splits the graph text within it, processing it separately with the Graph Encoder and the Text Encoder. In the model's architecture, the total number of layers in the Graph Encoder and the Text Encoder remains constant. After the independent encoding stage, the outputs of both encoders are combined and enter the joint encoding phase. Finally, the summary is extracted, and the generated text is subjected to a graph structure consistency loss.

3.1 Graph Content Extraction

Graph Text To effectively capture the textual structure of multi-documents, we introduce a groundbreaking data structure called Graph Text. Graph Text is designed as a non-connected graph that integrates multi-documents, with each node representing a document at a macroscopic level. The connections within the graph illustrate the relationships between textual segments. The nodes in the graph contain extracted factual information, consisting of unique facts distilled from the documents. By condensing essential information from the text, Graph Text enables a more profound understanding of the source text information during subsequent processing by models.

Graph Text Generation To generate Graph Text that accurately represents the textual structure information, we refer to the generation method proposed by (Koncel-Kedziorski et al., 2019) to extract and obtain information with redundant facts from multiple documents. Specifically, our extraction method follows the following strategies:

Open Information Extraction(OIE) : We employ an open information extraction system (Gardner et al., 2017), to extract the base nodes of the graph text using open information extraction triplets. We concatenate all the documents into a single document, perform OIE, and add a token before each document. Given that multi-document summaries often contain longer texts, this process ensures that we obtain enough text to serve as graph nodes.

Building Coarse-Grained Text Graph : We then cluster the extracted discrete text segments to create an overall connectivity graph that is distributed by documents and encompasses the resolution, disambiguation, and reference relationships of the entities within them. In our approach, we utilize the graph structure representation of special tokens and the linear graph model from (Pasunuru et al., 2021a) to ensure the consistency of graph information.

Fine-Grained Extraction of Graph Text : As mentioned, the coarse-grained graph text is further refined to obtain a detailed information graph, filtering out erroneous and duplicate information. We use Sentence Transformers (Reimers and Gurevych, 2020) to calculate the similarity between sentences, taking into account both entity similarity and contextual sentence similarity. If both the similarity of the entities and the similarity of the connection sentences exceed the matching threshold, we deem

the information redundant and eliminate the redundant nodes. We use breadth-first search to traverse all the subgraphs of the document graphs. The resulting graph text is considered to be fine-grained and contains minimal redundant information when all the nodes have been computed.

Algorithm 1 Graph Text Generation

Input: Document cluster
Input: List of Documents Ds
Output: Lineared Graph Text
  for $D \in DS$ **do**
    $T \leftarrow$ AllenNLP(D) $Ts.append(T)$
  end for
  $\tau' \leftarrow$ selected threshold
  for $T \in Ts$ **do**
    for $T' \in Ts$ *w/o* T **do**
      $\tau \leftarrow \text{SentenceTransformers}_1(T, \quad T')$
      if $\tau >= \tau'$ **then**
        Delete T' in Ts
      end if
    end for
  end for
  return Ts

3.2 Document-Level Attention

Compared to single-text documents, multi-text documents have longer length and more complex content structures. Therefore, it is necessary to enhance the model's attention to text information differences during the decoding process. To address this, we further design a text-level hierarchical decoder to leverage the document hierarchy already captured by the extended merge encoder.

The document-level hierarchical decoder follows the same architecture as a Transformer decoder (which first carries out mask self-attention with the previously generated tokens to prevent attention to future words, followed by cross-attention with the input tokens), is initialized with pretrained weights. We do not make any modifications to the mask self-attention mechanism of the decoder, as it operates independently of the original text input. For cross-attention, we leverage special tokens (Sec.3.1) that indicate document boundaries as document-level representation to scale the attention weights of tokens within the respective documents.

Given N documents, we denote the cross-attention scores in the decoder toward each document token as $\{a_{n,0}, a_{n,1}, \cdots, a_{n,k_n}\}$, where k_n is the number of tokens in the n document. Next, we normalize the attention scores in each document as follows. We first obtain the scaling weights S for all documents as:

$$\mathbf{S} = [s_0, \cdots, s_n] = Softmax(a_{0,0}, \cdots, a_{n,0}) \tag{1}$$

where $a_{n,0}$ is the attention score for the special token in the n document representing the boundary of document.Then, we acquire the normalized attention scores for tokens in each document as:

$$\mathbf{A_k} = s_n \times Softmax(a_{n,0}, \cdots, a_{n,k}) \tag{2}$$

where A_k represents the document-level attention scores of kth document.

The intuition behind normalization to cross-attention score is to ensure that decoder can recognize the relative degree of difference between documents without changing the relative attention weights within each document, fitting to the process PLMs experienced by during the pre-training stage.

3.3 Integration Encoding

After encoding the graph and text separately using the graph encoder and text encoder, it is required to integrate the information from both sources. In order to facilitate effective interaction between the two modalities, our method construct merge encoder input by concatenating the outputs and feed them into a joint encoder. The implementation of the joint encoder is similar to the text encoder, with the difference being that the encoding states undergo a unified node extraction process to facilitate the calculation of graph loss.

According to the conclusion of GraphLong (Pasunuru et al., 2021b), we seperately encode the documents text and linearized graph text via text encoder and graph encoder initialized with the pretrained encoder to achieve a significant improvement. Let denote token representations and graph lineared representations as $\{w_1, w_2, \cdots\}$ and $\{g_1, g_2, \cdots\}$. We truncate each document to the size of prefixed maximum length of pretrained model divided by N (where N is the number of documents in the cluster), and concatenate all documents with a special token. Then, the output of text encoder and graph encoder are:

$$[w_1, w_2, \cdots, w_n] = TextEnc(D_1, D_2 \cdots, D_N) \tag{3}$$

$$[g_1, g_2, \cdots, g_m] = GraphEnc(G_1, G_2 \cdots, G_N) \quad (4)$$

where n and m respectively represents the text length of the source document and the corresponding Graph Text.

Let w and g represent text representations corresponding to source documents text and its linearized graph text. In addition to directly concatenating w and g as the input of decoder we combine the both text encoder and graph encoder outputs and give them as a single input to the merge encoder. The combined input to merge encoder is defined as:

$$\mathbf{E} = [e_1, e_2, \cdots, e_{n+m}] = [w; g] \quad (5)$$

where [;] represents the concatenation and $\mathbf{E}$ represents the final input to merge encoder (total number of inputs is equal to the sum of documents text and graph text tokens).

Finally, the final encoder output $\hat{\mathbf{E}}$ is obtained through merge encoder encoding:

$$\hat{\mathbf{E}} = MergeEnc(\mathbf{E}) \quad (6)$$

Compared to the work of (Xiao et al., 2021), we direct global attention towards different special tokens based on the pretrained model. This approach ensures that the model recognizes document boundaries and improves interactions between documents. Additionally, we incorporate global attention to tokens that contain special semantic information from graph nodes (Sec.3.1). These special tokens facilitate cross-document token interactions and preserve keywords in the texts. However, our utilization of global tokens differs from Longformer (Beltagy et al., 2020) in terms of attention patterns. This enables each global token to encode information from different documents during the PLM's pre-training process, interacting with other tokens to exchange information within the same document as well as across documents.

3.4 Graph Structure-Aware Loss

Due to the significant resemblance between the discourse structure of multiple texts and the graph representation proposed in our work, once the graph representation is constructed, we train the model to comprehend and enhance the summarization of the internal structure of multi-texts. We achieve this by training the model using a position-based optimization approach on the graph structure. Similar to the previously mentioned method for extracting the Golden Summary, we permit the presence of redundant data in the summary extraction process to ensure that the graph representation has an adequate number of edges for regularization, particularly in cases with limited source texts. In our additional optimization method, the graph representation of the original text is denoted as $G_{D_{Src}}$, and the graph representation of the summary text is denoted as G_S.

We define the loss as the cosine similarity between the node embeddings of the summary graph S_S and the source graph $S_{D_{Src}}$. We believe that the two generalized graph structures are similar, allowing us to extract the embeddings $S^*_{D_{Src}}$ and S^*_S of the key node structures in each graph and calculate the similarity between the graph structures. The similarity measured on the graph structures is defined as:

$$\mathcal{L} = \cos(S^*_{D_{Src}}, S^*_S) \quad (7)$$

4 Experiments

We evaluate our proposed GSAC and compare it against SOTA abstractive MDS models over several datasets and different backbones. We also report the results of an ablation study to show the effectiveness of the components of GSAC.

4.1 Datasets and Metrics

We conduct experiments on a various range of text summarization datasets as follows. In this work, we use the same splits provided by Huggingface Dataset for training/validation/test, respectively. Following (Fabbri et al., 2019), we truncate N source documents and its graph content to a total length of L tokens such that we choose L/N tokens from each document and graph content and concatenate the truncated documents and contents as input.

Multi-News A large-scale English dataset (Fabbri et al., 2019) containing various topics in news domain. Each documents set consists of 2 to 10 documents describing the same topic.

WikiSum This dataset (Liu et al., 2018) provides how-to articles from wikihow.com and contains the article, the summary and the wikihow url, written as a coherent paragraph.

Multi-X-Science A large English dataset (Lu et al., 2020) containing document summaries collected from scientific articles. The task of the Multi-XScience dataset is to generate the related work section of a target scientific paper.

Metrics Like most previous works, we also use the F1 variants of the ROUGE-N scores for performance evaluation. Following previous work (Fabbri et al., 2019), we report the F1 scores of ROUGE-1, ROUGE-2, and ROUGE-L on datasets.

Dataset	Example	SrcL	SumL
arXiv	214K	6021	272
Wikisum	1.5M	2238	113
Multi-XScience	40K	700	105
Multi-News	56K	1793	217

Table 1: The statistics of all datasets explored in experiments.

4.2 Baselines

We compare our proposed framework on two baselines.

- PRIMERA The overall architecture is the same as LED (Beltagy et al., 2020). Unlike most pretrained models, PRIMERA is specifically pretrained for multi-document summarization, which gives it a strong performance in text summarization tasks. Similarly, we used the public version parameters from Huggingface.

- LED Longformer Encoder-Decoder, is another baseline model that has shown competitive performance in long document summarization. We initialized it directly with the "BART-large" checkpoint, but it contains slightly more parameters (435M) owing to the inclusion of extra global token attention projections. We use the pretrained checkpoints hosted on Huggingface as the baseline models to be directly fine-tuned with our datasets.

4.3 Training Details

Following (Xiao et al., 2021), we use source and target truncation of 4096 and 1024 respectively for all experiments.For the PLMs, we use the large version of the models. We start with the pre-trained model as mentioned in table and fine-tune on document summarization datasets. We tune all our models based on the validation performance. By default, we use Adam optimizer with a learning rate of 3e-6 with 1000 warm-up steps.We apply dropout of 0.1 and a label smoothing of 0.1. We perform standard tokenization following previous work and lowercase both source and target.For our pretrained model with Longformer attention, we use a default attention context window size of 256.All experiments are run on 4xNVIDIA RTX 3090.

5 Result and Analysis

5.1 Overall results

In this section, we show the Rouge results of our proposed method GSAC against serveral strong baselines over all datasets and list the comparison results in Table 3. GSAC outperforms most of the benchmark models on ROUGE score, demonstrating the effectiveness of GSAC for MDS. To give a fair comparison, we rerun all the baseline models.As shown in Table 3, our GSAC consistently improves from the corresponding "PRIMERA", despite that they are both initialized from the same weights.

Models	Mul-news	Mul-XSc	Wikisum
PEGASUS	36.5	-	-
LED	17.3	14.6	10.5
PRIMERA	46.6	**31.9**	28.0
MGSum	45.6	-	23.2
GraphSum	45.0	18.8	**42.6**
BART-large	**47.4**	31.5	-
GASC	**49.4**	**34.3**	**42.8**

Table 2: Model performance on summarizing MULTI-NEWS, MULTI-XSCIENCE, and WIKISUM in terms of ROUGE scores.

5.2 Graph Text Input Strategy

Due to the varying input lengths of different models, we have developed two strategies for integrating graph text with the source text for experimentation. Detailed examples can be found in Appendix Table 7.

Truncation In this approach, we directly combine the source text with the graph text and then trim the surplus text input according to the model's maximum input length. However, this truncation leads to information loss, which particularly affects the integrity of the graph information.

Proportional Concatenation We attempted to achieve a balance between the source text and the graph text by proportionally combining them. In our experiments, we used a ratio of 3:7, with the source text accounting for 30 % and the graph text accounting for 70 % of the combined input.

5.3 Structure of Graph Text

We define graph text as a multi-connected graph text on multi documents. It consists of four components: narrative, reference, subject, and object. The narrative component represents factual text extracted using the OpenIE tool, while the remaining three components are represented using special tokens. The corresponding special tokens are <pred>, <obj>, <cat> and <sub>.

For example, as shown below, although the extracted graph text may have different lengths compared to the original text and the golden summary, our proposed method for graph text node extraction ensures that their embedding presentations are in the same dimension.

5.4 Extraction Evaluation

One of the crucial aspects of the extracted graph text that we need to focus on is its graph structure characteristics. We have analyzed the specific features of the graph text extracted from different datasets. As shown in Table3, it can be observed that the number of subjects extracted from different datasets has a relatively narrow range, and compared to the length of the corresponding text, the graph labels show an even smaller variation in length. This indicates that our proposed method for extracting text from graphs effectively reduces information redundancy through techniques such as search pruning. The graph displayed demonstrates that our extraction method maintains a stable proportion of different Graph Special Tokens between the source and target texts for both abstracts and source texts. This consistency in the structure of the extracted graph text across different sources ensures that the optimization of the summary's structure can be correctly performed while accounting for the loss of the graph structure.

5.5 Results on Multi Backbone

To demonstrate the wide-ranging applicability of our proposed method across various Seq2Seq models, we conducted experiments not only on PRIMERA but also on LED and Pegasus-Xsum. The experimental results indicate that our graph text approach performs well for models that handle long inputs. Although we observed a performance decrease of approximately 1 % in terms of the ROUGE-1 score on Pegasus, the reason behind it is evident. The shorter model is unable to accommodate the entire length of our proposed joint encoding, resulting in a loss of information on the graph encoding side. However, we also observed that our method effectively enhances the quality of summarization for models handling lengthy inputs. Taking LED as an example, we achieved an improvement of approximately 1.87 % in ROUGE-1 score compared to the baseline.

5.6 Ablations

To figure out the effectiveness of all components for GSAC performance, we conduct an ablation study on Multi-news dataset and compare our full method with various model variants which are composed of different components of GSAC. In addition, we also considered the impact of the number of graph encoder layers on the performance of the GSAC in table 5.

As compared to the baseline PRIMERA in table 6, simply adding any additional component all can gain improvements in Rouge results. Firstly, by introducing the intergration encoding and graph structure-aware loss, we get great performance gains. Besides, by adding the document-level attention during decoding, we gain slight improvements. However, when used together, these components lead to further performance improvements, eventually arriving at our full model GSAC. It is interesting to see that two graph encoder layers is better than that of four graph encoder layers. However, when there are six graph encoder layers, GSAC model reaches optimal performance.

6 Limitations

Time In our GSAC framework, we specifically focus on ensuring consistency in the graph structure. Since each set of texts requires a corresponding pair of graph-based representations, the dataset needs to undergo separate graph-based text extraction in our approach. Due to the extensive searching and matching required by the extraction algorithm, the data preparation phase of model training will take longer.

Increased Inference Cost The architecture of the GSAC using graph-based encoders involves layering and chunking, which results in increased

Model	<sub> Source/Target	<obj> Source/Target	<pred> Source/Target
Multi-News	Train<13.24/3.29> Val<13.03/3.30> Test<13.15/3.24>	Train<34.11/10.02> Val<33.63/9.96> Test<33.93/9.92>	Same as <obj>
Multi-XScience	Train<6.98/4.44> Val<21.84/4.42> Test<21.54/4.42>	Train<11.52/5.71> Val<36.22/5.67> Test<35.92/5.68>	Same as <obj>
WikiSum	Train<21.56/21.56> Val<21.59/21.59> Test<21.60/2.90>	Train<32.33/32.33> Val<32.29/32.29> Test<32.35/4.18>	Same as <obj>

Table 3: The statistical analysis of the Special Token's proportion in the graphs extracted from our dataset reveals that the ratio of tokens in the Source Text and Golden Summary remains consistent across different splits of the given dataset. This observation suggests that the graph structure in the Golden Summary demonstrates a certain degree of stability.

		R-1	R-2	R-L	Input
PRIMERA	Backbone	47.61	18.66	23.24	4096
	ours	**49.92**	**20.25**	**25.34**	4096
Pegasus-Xsum	Backbone	**47.21**	18.06	25.07	512
	ours	46.38	**18.63**	**25.44**	512
LED-large-16384	Backbone	43.37	16.21	23.60	16384
	ours	**45.24**	**16.68**	**23.86**	16384

Table 4: ROUGE scores of different backbone models on Multi-News. For all backbone models with various maximum input lengths, ROUGE scores increase with the help of proposed framework. Input indicates the maximum number of tokens the model can take.

Models	R-1	R-2	R-L
PRIMERA	46.6	18.8	23.2
GSAC w/t 2 Layers	47.8	19.1	**25.2**
GSAC w/t 4 Layers	48.3	18.9	24.9
GSAC w/t 6 Layers	**48.6**	**19.5**	24.8

Table 5: Summarization results of GSAC with different encoder layers on Multi-News.

overall inference time. While a substantial amount of graph-based text may not be necessary during the inference stage, the extraction of inference samples still proves to be time-consuming and adds to the processing stage. Therefore, to achieve a balance between computational cost and effectiveness during inference, the appropriate numbers of graphs and graph branches should be chosen.

Instability of Graph Structure In our proposed method, we always assume that the graph itself has significance. However, this assumption only holds true for inference on our multi-document dataset. When the text is brief and lacks clear hierarchies, GSAC may struggle to be effective. The extracted graph structure may inadequately represent or even disregard the structural information of the text, leading to a decline in model performance.

Models	R-1	R-2	R-L
PRIMERA	46.6	18.8	23.2
GSAC	**49.4**	**19.5**	**25.3**
w/o Document Attn	48.6	**19.5**	24.8
w/o Graph Loss	48.0	19.1	23.9

Table 6: Results of ablation study on Multi-News.

7 Conclusion

In this paper, we focus on the task of multi-document summarization (MDS). We propose a framework for MDS based on graph text representation, along with a novel approach for constructing text graphs. Furthermore, our approach is compatible with any encoder-decoder architecture of pre-trained models, enabling efficient fine-tuning of pre-trained language models (PLMs) on specific MDS datasets without the addition of new parameters. The proposed method introduces innovative document-level interactions by incorporating global tokens in both the encoder and decoder, leveraging the generalization capabilities of pre-trained language models (PLMs) across diverse domains.

Acknowledgements

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Peng Cui and Le Hu. 2021. Topic-guided abstractive multi-document summarization.

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

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun Hsuan Sung, and Yinfei Yang. 2021. Longt5: Efficient text-to-text transformer for long sequences. *arXiv e-prints*.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text Generation from Knowledge Graphs with Graph Transformers. 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 2284–2293, Minneapolis, Minnesota. Association for Computational Linguistics.

Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, and Junping Du. 2020a. Leveraging graph to improve abstractive multi-document summarization.

Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, Haifeng Wang, and Junping Du. 2020b. Leveraging graph to improve abstractive multi-document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6232–6243, Online. Association for Computational Linguistics.

Wei Li and Hai Zhuge. 2021. Abstractive multi-document summarization based on semantic link network. *IEEE Transactions on Knowledge and Data Engineering*, PP(99):1–1.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences.

Shuaiqi Liu, Jiannong Cao, Ruosong Yang, and Zhiyuan Wen. 2021. Highlight-transformer: Leveraging key phrase aware attention to improve abstractive multi-document summarization. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 5021–5027, Online. Association for Computational Linguistics.

Yao Lu, Yue Dong, and Laurent Charlin. 2020. Multi-xscience: A large-scale dataset for extreme multi-document summarization of scientific articles.

Congbo Ma, Wei Emma Zhang, Mingyu Guo, Hu Wang, and Quan Z Sheng. 2022. Multi-document summarization via deep learning techniques: A survey. *ACM Computing Surveys*, 55(5):1–37.

Ramakanth Pasunuru, Mengwen Liu, Mohit Bansal, Sujith Ravi, and Markus Dreyer. 2021a. Efficiently summarizing text and graph encodings of multi-document clusters. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4768–4779, Online. Association for Computational Linguistics.

Ramakanth Pasunuru, Mengwen Liu, Mohit Bansal, Sujith Ravi, and Markus Dreyer. 2021b. Efficiently summarizing text and graph encodings of multi-document clusters. In *North American Chapter of the Association for Computational Linguistics*.

Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Qiqihar Junior Teachers College*, 22:2004.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Yun Zhu Song, Yi Syuan Chen, and Hong Han Shuai. 2022. Improving multi-document summarization through referenced flexible extraction with credit-awareness.

Fangwei Zhu Juanzi Li Lei Hou Tu, Jifan Yu and Jian-Yun Nie. 2022. Uper: Boosting multi-document summarization with an unsupervised prompt-based extractor. In *International Committee on Computational Linguistics*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv*.

Xiaojun Wan. 2008. An exploration of document impact on graph-based multi-document summarization. In *Conference on Empirical Methods in Natural Language Processing*.

Xiaojun Wan and Jianwu Yang. 2006. Improved affinity graph based multi-document summarization. In *North American Chapter of the Association for Computational Linguistics*.

Wen Xiao, Iz Beltagy, Giuseppe Carenini, and Arman Cohan. 2021. Primera: Pyramid-based masked sentence pre-training for multi-document summarization.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Yongjing Yin, Linfeng Song, Jinsong Su, Jiali Zeng, Chulun Zhou, and Jiebo Luo. 2019. Graph-based neural sentence ordering.

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.

Graph Text Input Strategy

Truncation: Tucker Carlson Exposes His Own Sexism on Twitter (Updated) Tucker Carlson has done some good work in the past... His site, The Daily Caller, is a frequent stop of mine and many other Conservatives. They were responsible for exposing the Journolist scandal, which highlighted the planning and coordination of many members of the left-wing press.

I will always be grateful to Tucker's team for bringing that story to light. This is also why I am so angered by Tucker's recent actions. I thought he was better than this. If you haven't heard by now, Monday evening, Tucker Carlson posted a disturbing tweet about Governor Palin which said: Palin's popularity falling in Iowa, but maintains lead to become supreme commander of Milfistan Aside from Tucker's sheep-like response to warped poll numbers, he also failed to take ownership of his sexist comment.

He deleted the original (which is why I had to link to a retweet) obviously aware that what he had posted was wrong. Unfortunately for him, many people had already seen it and responded. You can't put the toothpaste back in the tube, Tucker. Is this the sort of treatment that Conservative women, who want to get involved in the process, are expected to put up with? Is it okay for male columnists (Conservative or otherwise) to continue objectifying women in the world of politics? No it's not! The best thing Tucker Carlson could do, is admit that what he tweeted was wrong, apologize to Governor Palin, and urge his fellow colleagues to be respectful with their language and written word. What he did was demeaning and offensive, and there is no place for it in Conservative circles. Update: This is a poor attempt at an apology.

Tucker Carlson tries to cover his tracks this morning by repeating the same mistakes he made last night. He wrote: Apparently Charlie Sheen got control of my Twitter account last night while I was at dinner. Apologies for his behavior. He didn't take responsibility for his comment and he fails horribly at humor. Try again, and Tucker... you're not funny. Update II: Almost a day later, he finally apologizes: I'm sorry for last night's tweet. I meant absolutely no offense. Not the first dumb thing I've said. Hopefully the last. Tweet with a location You can add location information to your Tweets, such as your city or precise location, from the web and via third-party applications. You always have the option to delete your Tweet location history. Learn more I am not down with @karlrove @tuckercarlson misogynist mockery of @sarahpalinusa .

Sick of it. On Monday night, while the rest of the world was watching Charlie Sheen flame out live on CNN, Tucker Carlson took to Twitter to make some impolitic statements of his own. "Palin's popularity falling in Iowa, but maintains lead to become supreme commander of Milfistan," he wrote. By the next morning, the tweet was deleted and he had apologized, writing, "Apparently Charlie Sheen got control of my Twitter account last night while I was at dinner. Apologies for his behavior." But that wasn't enough to spare him the ire of conservative women on the blogosphere and Twitter. On Tuesday, before Carlson's first apology, Stacy Drake, writing on Conservatives4Palin, praised Carlson's works at The Daily Caller, particularly the leaks of the Journolist emails, saying that's why his tweet stung so badly.

Aside from Tucker's sheep-like response to warped poll numbers, he also failed to take ownership of his sexist comment. He deleted the original (which is why I had to link to a retweet) obviously aware that what he had posted was wrong. Unfortunately for him, many people had already seen it and responded. You can't put the toothpaste back in the tube, Tucker. Is this the sort of treatment that Conservative women, who want to get involved in the process, are expected to put up with? Is it okay for male columnists (Conservative or otherwise) to continue objectifying women in the world of politics? No it's not! She was unimpressed with his first apology, and called for him to apologize to Palin while continuing to denounce him for sexism on her Twitter account.

Michelle Malkin joined the calls Tuesday, tweeting: "I am not down with @karlrove @tuckercarlson misogynist mockery of @sarahpalinusa. Sick of it." Later Tuesday, Carlson obliged: "I'm sorry for last night's tweet. I meant absolutely no offense. Not the first dumb thing I've said. Hopefully the last." Some bros have come to Carlson's aid. Tuesday, Erick Erickson tweeted, "Maybe my sense of humor needs to be recalibrated, but when I heard @TuckerCarlson's MILFistan comment, I laughed then got out my passport." (Needless to say, Drake was not amused.) But by Wednesday, the thing had escalated into a full-blown war of the sexes within the conservative blogosphere, with Whitney Pitcher taking Carlson's tweet as inspiration for her post on Conservatives4Palin: "MILF–Misogynists (and Elites) I'd Like to Fulminate." Perhaps an additional reason that Governor Palin does not win the respect of the Elite and Establishment is that you cannot be praised for your "perfectly creased pants" if you often wear a skirt, right David Brooks? The continued line of attack from the Establishment and Elite men in the GOP have come as a result of Governor Palin's genetic makeup. This post has been updated to correct the spelling of Stacy Drake's first name.

<sub> I <obj> he was better than this <pred> thought <obj> to a retweet) obviously aware that what he had posted was wrong <pred> link <obj> absolutely no offense <pred> meant <obj> Not the first dumb thing <pred> 've <obj> @TuckerCarlson 's MILFistan comment <pred> heard <obj> my passport <pred> got <sub> Not the first dumb thing <obj> I 've <pred> said

<sub> Tucker Carlson <obj> a disturbing tweet about Governor Palin which said : Palin 's popularity falling in Iowa <pred> posted <obj> to cover his tracks this morning by repeating the same mistakes he made last night <pred> tries <obj> his tracks <pred> cover <obj> the same mistakes he made last night <pred> repeating <obj> some impolitic statements of his own <pred> make <sub> he <obj> ownership of his sexist

Concatenation Tucker Carlson Exposes His Own Sexism on Twitter (Updated) Tucker Carlson has done some good work in the past... His site, The Daily Caller, is a frequent stop of mine and many other Conservatives. They were responsible for exposing the Journolist scandal, which highlighted the planning and coordination of many members of the left-wing press. I will always be grateful to Tucker's team for bringing that story to light. This is also why I am so angered by Tucker's recent actions. I thought he was better than this.

If you haven't heard by now, Monday evening, Tucker Carlson posted a disturbing tweet about Governor Palin which said: Palin's popularity falling in Iowa, but maintains lead to become supreme commander of Milfistan Aside from Tucker's sheep-like response to warped poll numbers, he also failed to take ownership of his sexist comment. He deleted the original (which is why I had to link to a retweet) obviously aware that what he had posted was wrong. Unfortunately for him, many people had already seen it and responded. You can't put the toothpaste back in the tube, Tucker. Is this the sort of treatment that Conservative women, who want to get involved in the process, are expected to put up with? Is it okay for male columnists (Conservative or otherwise) to continue objectifying women in the world of politics? No it's not!

The best thing Tucker Carlson could do, is admit that what he tweeted was wrong, apologize to Governor Palin, and urge his fellow colleagues to be respectful with their language and written word. What he did was demeaning and offensive, and there is no place for it in Conservative circles. Update: This is a poor attempt at an apology. Tucker Carlson tries to cover his tracks this morning by repeating the same mistakes he made last night. He wrote: Apparently Charlie Sheen got control of my Twitter account last night while I was at dinner. Apologies for his behavior. He didn't take responsibility for his comment and he fails horribly at humor. Try again, and Tucker... you're not funny. Update II: Almost a day later, he finally apologizes: I'm sorry for last night's tweet.

I meant absolutely no offense. Not the first dumb thing I've said. Hopefully the last. Tweet with a location You can add location information to your Tweets, such as your city or precise location, from the web and via third-party applications. You always have the option to delete your Tweet location history. Learn more I am not down with @karlrove @tuckercarlson misogynist mockery of @sarahpalinusa . Sick of it. On Monday night, while the rest of the world was watching Charlie Sheen flame out live on CNN, Tucker Carlson took to Twitter to make some impolitic statements of his own. "Palin's popularity falling in Iowa, but maintains lead to become supreme commander of Milfistan," he wrote. By the next morning, the tweet was deleted and he had apologized, writing, "Apparently Charlie Sheen got control of my Twitter account last night while I was at dinner. Apologies for his behavior."

But that wasn't enough to spare him the ire of conservative women on the blogosphere and Twitter. On Tuesday, before Carlson's first apology, Stacy Drake, writing on Conservatives4Palin, praised Carlson's works at The Daily Caller, particularly the leaks of the Journolist emails, saying that's why his tweet stung so badly. Aside from Tucker's sheep-like response to warped poll numbers, he also failed to take ownership of his sexist comment. He deleted the original (which is why I had to link to a retweet) obviously aware that what he had posted was wrong. Unfortunately for him, many people had already seen it and responded. You can't put the toothpaste back in the tube, Tucker. Is this the sort of treatment that Conservative women, who want to get involved in the process, are expected to put up with? Is it okay for male columnists (Conservative or otherwise) to continue objectifying women in the world of politics? No it's not! She was unimpressed with his first apology, and called for him to apologize to Palin while continuing to denounce him for sexism on her Twitter account. Michelle Malkin joined the calls Tuesday, tweeting: "I am not down with @karlrove @tuckercarlson misogynist mockery of @sarahpalinusa. Sick of it."

Later Tuesday, Carlson <sub> I <obj> he was better than this <pred> thought <obj> to a retweet) obviously aware that what he had posted was wrong <pred> link <obj> absolutely no offense <pred> meant <obj> Not the first dumb thing <pred> 've <obj> @TuckerCarlson 's MILFistan comment <pred> heard <obj> my passport <pred> got <sub> Not the first dumb thing <obj> I 've <pred> said

<sub> Tucker Carlson <obj> a disturbing tweet about Governor Palin which said : Palin 's popularity falling in Iowa <pred> posted <obj> to cover his tracks this morning by repeating the same mistakes he made last night <pred> tries <obj> his tracks <pred> cover <obj> the same mistakes he made last night <pred> repeating <obj> some impolitic statements of his own <pred> make <sub> he <obj> ownership of his sexist comment <pred> take <obj> what <pred> posted <cat> tweeted <obj> the same mistakes <pred> made <obj> I 'm sorry for last night 's tweet <pred> apologizes <obj> Palin 's popularity falling in Iowa , but maintains lead to become supreme commander of Milfistan <pred> wrote <sub> He <obj> the original (which is why I had to link to a retweet) obviously aware that what he had posted was wrong <pred> deleted <obj> Apparently Charlie Sheen got control of my Twitter account last night while I was at dinner <pred> wrote <obj> responsibility <pred> take <sub> You <obj> n't put the toothpaste back in the tube , Tucker <pred> ca <obj> your Tweet location history <pred> delete <sub> control of my Twitter account <obj> Apparently Charlie Sheen <pred> got <obj> " Apparently Charlie Sheen <pred> got <sub> Michelle Malkin <obj> the calls <pred> joined <obj> " I am not down with @karlrove @tuckercarlson misogynist mockery of @sarahpalinusa <pred> tweeting

A Example of the input strategy

The table 7 presents an example from the Multi-News dataset. To enable models with different input lengths to benefit from GSAC, we employed the Truncation strategy for long-input models, aiming to accommodate more source text and enhance information capacity while ensuring the correct input of all information. Conversely, for shorter input models, we adopted the Concatenation strategy to ensure sufficient graph information is pre-inserted and to summarize key information effectively.