Efficiently Summarizing Text and Graph Encodings of Multi-Document Clusters

Ramakanth Pasunuru[1] **Mengwen Liu**[2] **Mohit Bansal**[1]
Sujith Ravi[2] **Markus Dreyer**[2]
[1]UNC Chapel Hill
[2]Amazon Alexa
{ram,mbansal}@cs.unc.edu, {mengwliu,sujithai,mddreyer}@amazon.com

Abstract

This paper presents an efficient graph-enhanced approach to multi-document summarization (MDS) with an encoder-decoder Transformer model. This model is based on recent advances in pre-training both encoder *and* decoder on very large text data (Lewis et al., 2019), and it incorporates an efficient encoding mechanism (Beltagy et al., 2020) that avoids the quadratic memory growth typical for traditional Transformers. We show that this powerful combination not only scales to large input documents commonly found when summarizing news clusters; it also enables us to process additional input in the form of auxiliary graph representations, which we derive from the multi-document clusters. We present a mechanism to incorporate such graph information into the encoder-decoder model that was pre-trained on text only. Our approach leads to significant improvements on the Multi-News dataset, overall leading to an average 1.8 ROUGE score improvement over previous work (Li et al., 2020). We also show improvements in a transfer-only setup on the DUC-2004 dataset. The graph encodings lead to summaries that are more abstractive. Human evaluation shows that they are also more informative and factually more consistent with their input documents.[1]

1 Introduction

Abstractive Multi-Document Summarization (MDS), the task of writing a consolidated summary of the main information from multiple documents, has seen advancements with the introduction of large-scale datasets and powerful Transformer-based models (Liu et al., 2018; Liu and Lapata, 2019; Fabbri et al., 2019). However, some of the key challenges of MDS include lack of proper inter-document context-aware information, improper logical flow of information, and need for external deep context representations. Liu and Lapata (2019) and Li et al. (2020) have addressed the inter-document context modeling to some extent with local and global attention, and document-level similarity graphs. Further, Li et al. (2020) have addressed the later part of using external contextual information (large pre-trained language models, e.g., RoBERTa (Liu et al., 2019)) to improve the performance of MDS models. However, these pre-trained language models are (1) not scalable for long documents because of their encoding length limit and quadratic memory growth; and (2) they do not jointly explore alternate auxiliary information, e.g., semantic graphs derived from multi-document clusters.

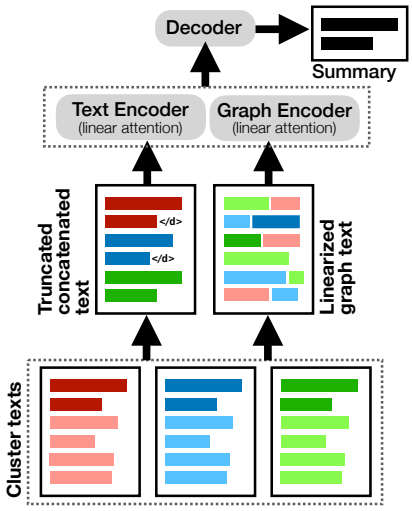


Figure 1: Illustration of our dual-encoder approach to summarizing multi-document clusters with graph encodings. The truncated concatenated text contains the beginnings of each cluster document; the graphs contain information from the full documents.

Addressing these issues, we present an efficient graph-enhanced approach to multi-document summarization using a pre-trained encoder-decoder Transformer model (Lewis et al., 2019), depicted in Fig. 1, along with an efficient encoding mech-

[1]All our code publicly available at: https://github.com/amazon-research/BartGraphSumm.

Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 4768–4779
June 6–11, 2021.

anism to encode longer input texts. To this end, we first provide a strong baseline for MDS on the Multi-News dataset (Fabbri et al., 2019) using a pre-trained encoder-decoder model, called BART (Lewis et al., 2019). Next, we incorporate a Longformer-based approach (Beltagy et al., 2020) into the pre-trained BART model, replacing the quadratic memory growth of the full self-attention mechanism with an efficient context window-based attention mechanism that scales the memory linearly w.r.t. the input length. This enables us to encode longer documents than previous work. This efficient encoding mechanism comprises local and global attention mechanisms that address the challenge of modeling inter-document context.

Further, we build consolidated semantic graph representations of the multiple input documents and explore ways to incorporate them into the encoder-decoder model. The semantic graph for a given multi-document cluster is a compact representation of subject-predicate-object triplets (Stanovsky et al., 2018) extracted from the text of the documents; see Fig. 3 for an example. We propose a dual encoding mechanism that separately encodes the regular text of a multi-document cluster and a text representation of its graph. The regular text is encoded by the pre-trained BART encoder, while the graph text is encoded by a transformer encoder that is not pre-trained.

Empirically, we show that our approach (including the ability to use longer parts of the input documents and add auxiliary graph encodings) leads to significant improvements on the Multi-News dataset (achieving state-of-the-art), overall leading to an average 1.8 ROUGE score improvement over previous work (Li et al., 2020). Based on various automatic evaluation metrics, we show that adding graph encodings can help the model abstract away from the specific lexical content of the input and generate summaries that are more *abstractive*. Further human evaluation shows that they are also more informative and factually more consistent with their input documents. We also test our model with auxiliary graph encodings on the DUC-2004 dataset (Over and Yen, 2004) in a test-only transfer setup, and show that it improves the generalization performance better than a non-graph baseline model. Finally, we present ablations, such as analyzing the effect of input document length on the performance, qualitative analysis of the output summaries, and effect of various graph encoding approaches on the performance of the MDS system.

2 Related Work

Researchers have been interested in automatically summarizing multiple documents since the late 1990s. First works (Mani and Bloedorn, 1997; Radev and McKeown, 1998) cited the gaining popularity of the World Wide Web (WWW) as a motivation for the task. They modeled multi-document collections as graph structures – perhaps influenced by the link structure of the WWW itself. Mani and Bloedorn (1997) summarized pairs of documents by building a graph representation of each and performing graph matching to find salient regions across both documents. Radev and McKeown (1998) summarized multiple documents by mapping them to abstract template representations, then generating text from the templates.

In the early 2000s, datasets from the Document Understanding Conference (DUC), which included human-written summaries for multi-document clusters, sparked increased research interest. In LexRank, Erkan and Radev (2004) extracted the most salient sentences from a multi-document cluster by constructing a graph representing pairwise sentence similarities and running a PageRank algorithm on the graph. Subsequent approaches followed the same paradigm while improving diversity of the extracted sentences (Wan and Yang, 2006) or adding document-level information into the graph (Wan, 2008). Dasgupta et al. (2013) incorporated dependency graph features into their sentence relation graphs. Baralis et al. (2013) built graphs over sets of terms, rather than sentences. Li et al. (2016) built a graph over event mentions and their relationships, in order to summarize news events using sentence extraction techniques. Liu et al. (2015) and Liao et al. (2018) leveraged AMR formalism to convert source text into AMR graphs and then generate a summary using these graphs.

More recently, the introduction of larger datasets for MDS has enabled researchers to train neural models for multi-document summarization. Liu et al. (2018) introduced a large-scale dataset for MDS called WikiSum, based on Wikipedia articles. Liu and Lapata (2019) introduced a hierarchical Transformer model to better encode global and local aspects in multiple documents and showed improvements on WikiSum. Fabbri et al. (2019) introduced an MDS dataset of human-written abstracts from the newser.com website, along with

the source articles that are cited from these abstracts. Further, they also proposed a hierarchical neural model for MDS with an additional Maximal Marginal Relevance (MMR) module that calculates sentence ranking scores based on relevancy and redundancy. Li et al. (2020) further showed the usefulness of pre-trained language models to improve the performance on MDS. However, this approach lacks a pre-trained decoder, and it also limits the document length that can be encoded by the pre-trained language models. In contrast, our work utilizes the pre-trained seq2seq BART (Lewis et al., 2019) model to improve the performance on MDS. We have also incorporated the Longformer-based attention mechanism (Beltagy et al., 2020) into BART model to encode long documents.

To encode graphs into an MDS neural model, Fan et al. (2019) constructed a semantic graph representing key phrases and entities from the documents, as well as their expressed relationships; they used linearized forms of these graphs as inputs to their Transformer model. In contrast, we use dual encoders for encoding both documents text and linearized graph text information. Recently, Li et al. (2020) constructed a similarity graph, topic graph, and discourse graph between input documents and encoded this information directly, rather than in linearized form, into a Transformer. In our work, we build semantic graphs at the sentence level and create a consolidated graph representation by efficiently removing less useful information.

3 Models

In this section, we first discuss our baseline MDS model utilizing the pre-trained BART sequence-to-sequence model (Lewis et al., 2019). Next, we integrate a Longformer approach (Beltagy et al., 2020) into the BART model for encoding long documents. Finally, we discuss our integration of graph encodings into the BART model.

3.1 BART Baseline

Bidirectional Auto-Regressive Transformer (BART) (Lewis et al., 2019) is a sequence-to-sequence Transformer-based model where the encoder is bi-directional and the decoder is uni-directional. The objective of this model is to reconstruct the actual input from given noisy text input. Input noising strategies include token masking, sentence permutation, document rotation, token deletion, and text infilling. The BART model is pre-trained on large amounts of text.

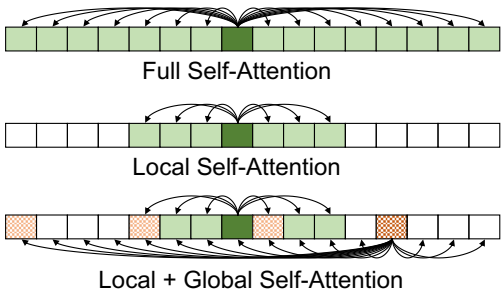


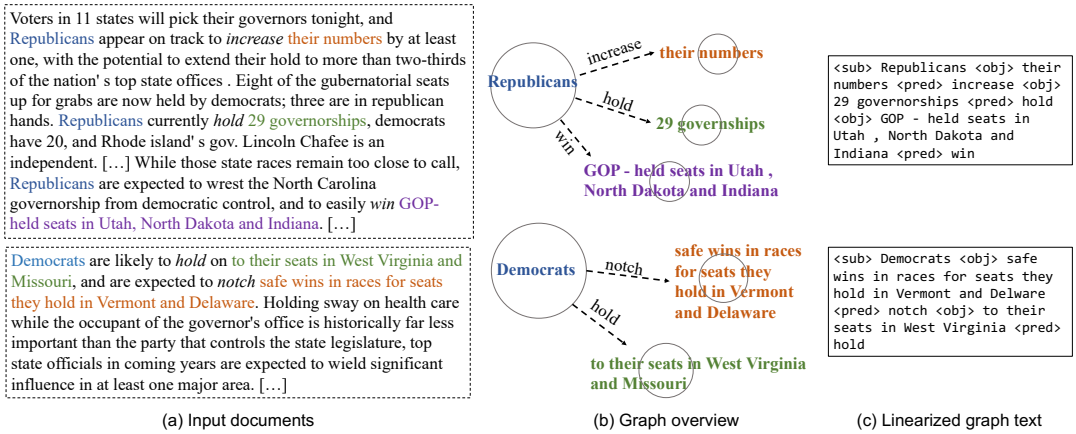
Figure 2: Pictorial overview of various attention mechanisms. Each block represents a token. Texture-filled blocks have global self-attention.

To perform multi-document summarization (MDS), we use the pre-trained BART model (trained as described above) and fine-tune it on the MDS datasets. Following Fabbri et al. (2019), we feed cluster documents as a single string joined by a special marker to the BART encoder.

3.2 BART-Long

Recently, the Longformer model (Beltagy et al., 2020) was introduced to allow the pre-trained RoBERTa model (Liu et al., 2019) to encode longer documents than its pre-fixed 512 limit. This is achieved by replacing the traditional full self-attention mechanism (top diagram in Fig. 2) in the Transformers (n^2 memory complexity) with a sparse context window-based attention mechanism which has linear memory in complexity w.r.t. the document length. Further, a small number of tokens are selected to attend over all other tokens, thus creating global attention along with the local context window-based attention (bottom diagram in Fig. 2).

Previously, Longformer has only been explored for pre-trained encoder-only based models, e.g, RoBERTa. In our work, we explore this approach to the pre-trained sequence-to-sequence BART model. We integrate the Longformer including both local and global attention mechanisms, into the BART model, named BART-Long, to encode documents much longer than its maximum token limit of 1024. In order to better encode the information from multiple documents, we incorporate global attention after every sentence and explore various context window sizes for local attention.



(a) Input documents (b) Graph overview (c) Linearized graph text

Figure 3: Our graph construction pipeline: (a) Text showing parts of input documents. (b) Overview of graph representation of the information using OIE triplets. (c) Conversion of graph information into text form.

3.3 BART with Graph Encodings

Recently, Fan et al. (2019) converted each multi-document input of the MDS into a graph and then pass the linearized form of this graph as input to a non-pre-trained sequence-to-sequence model, replacing the original text input. In contrast, our work explores the integration of graph encodings into a pre-trained BART model with a separate graph encoder. It is important and also challenging to encode graph representations into the pre-trained model while leveraging the pre-existing knowledge from pre-trained models. Moreover, we utilize the BART-Long model described in Sec. 3.2 to avoid the limitation in the input length for encoding both graph and textual information. Next, we describe how we convert multiple input documents into a consolidated graph representation and later describe how we encode this information into an extended BART architecture.

Graph Construction. Following Fan et al. (2019), we perform three steps for constructing a consolidated graph from multiple input documents. First, we do *co-reference resolution* within each document and extract *open information extraction triplets* (OIE) at the sentence level from all input documents.[2] Each OIE triplet consists of a subject, a predicate, and an object. Once we have all the triplets, in the second step, we build a graph with subjects and objects as nodes and the predicates as the edge relationship between the nodes. We also calculate the TF-IDF scores for each word in a document. This is useful in identifying similar phrases and merging their corresponding nodes in the graph.[3] Once we build the graph, we remove the clusters (sub-graphs) with only two nodes, thereby creating a *consolidated graph*. In the third step, we convert the graph into a *linearized form*. For this, we traverse sub-graphs in the order of their size, and within each sub-graph we simply start from a node with the highest centrality and move down the sub-graph in a breadth-first search approach to generate linearized text. We concatenate these texts together to form the linearized graph text. Fig. 3 gives an overview of our graph construction approach with examples of linearized graph. Here, we use special tokens like <sub> for subject, <pred> for predicate, <obj> for object, and <cat> for concatenating multiple predicates between a pair of a subject and an object.

Linear Graph Model. Our initial experiments combining both the documents text and linearized graph text into one single input for the BART model gave a slight improvement. To further enable better encoding, we used two encoders: (1) encoding the documents' original text via the pre-trained BART encoder; and (2) encoding the linearized graph text via a new graph encoder, as shown in Fig. 4. Let x_i and g_i represent the tokens at position i corresponding to the documents text and linearized graph text, respectively. Also, let the corresponding token embeddings be e_i^x and e_i^g, and the positional embeddings be p_i^x and p_i^g. Then, the input to the BART encoder (x_i^0) and graph encoder

[2]We use AllenNLP (https://allennlp.org/) library for co-reference resolution and extracting OIE triplets.

[3]We define one representative unique string (as a node) from the pool of all matched strings. We manually set the TF-IDF matching threshold to 0.5 based on graph size.

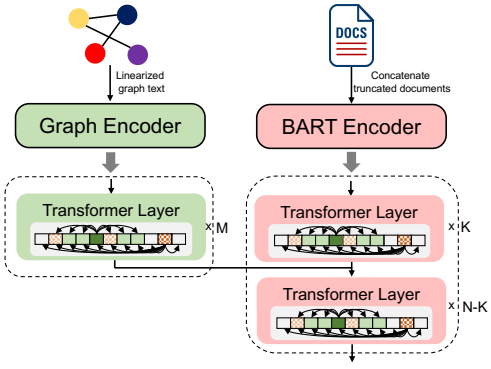


Figure 4: Overview of our approach with BART encoder and a graph encoder. All the Transformer layers use Longformer attention. We use pre-trained representations for BART encoder.

(g_i^0) are:

$$x_i^0 = e_i^x + p_i^x; \quad g_i^0 = e_i^g + p_i^g; \tag{1}$$

Let the final outputs of the graph encoder with M Transformer layers be g^M. Let the outputs of the BART encoder after K Transformer layers be x^K.[4] Now, we combine these outputs and give it as a single input to the $(K+1)^{th}$ layer of the BART encoder (as shown in Fig. 4). The combined input to $(K+1)^{th}$ Transformer layer is defined as:

$$\hat{x}^K = [x^K; g^M] \tag{2}$$

where $[;]$ represents the concatenation and $\hat{x}^K$ represents the input to $(K+1)^{th}$ layer (total number of inputs at this layer is equal to the sum of documents text and graph text tokens). Our approach of having separate encoders for graph information could bring the linearized graph text representations closer to that of the pre-trained BART representations.

4 Experimental Setup

Multi-News Dataset. The Multi-News dataset (Fabbri et al., 2019) consists of English news articles and the corresponding summaries written by professionals on the newser.com website. The articles in this dataset are curated from a diverse set of news sources (over $1,500$ sites). In this work, we use the same splits provided by Fabbri et al. (2019), i.e., $44,972/5,622/5,622$ examples for training/validation/test, respectively. Following Fabbri et al. (2019), we truncate N documents to a total length of L tokens such that we choose L/N tokens from each document and concatenate the truncated documents as input.

DUC-2004 Dataset. The DUC-2004 dataset (Over and Yen, 2004) consists of 50 topics with 10 English documents per topic.[5] Each topic has 4 human-written summaries. In our work, we use this dataset as test-only setup to analyze the transfer skills of our models.

Evaluation Metrics. We evaluate our models via automatic evaluation metrics using ROUGE[6] (Lin, 2004), as well as human evaluations of informativeness, coherence, and factual consistency. Following previous work (Fabbri et al., 2019), we report the F1 scores of ROUGE-1, ROUGE-2, and ROUGE-L on Multi-News dataset, and report the F1 scores of ROUGE-1, ROUGE-2, and ROUGE-SU on DUC-2004 dataset with a 100 word limit. In order to have a fair comparison with previous work, we report summary-level ROUGE-L scores.

Training Details. We tune all our models based on the validation performance. We start with the pre-trained BART large model and fine-tune on the Multi-News dataset.[7] All our new methods are implemented on top of fairseq library.[8] We train each model on 4 Nvidia V100 GPUs. By default, we use Adam optimizer with a learning rate of 2×10^5 and manually tune in the range: $[1 \times 10^5, 4 \times 10^5]$ with 500 warm-up steps. We apply dropout of 0.1 and a label smoothing of 0.1. We perform standard tokenization following previous work (Fabbri et al., 2019) and lowercase both source and target. During inference, we use a minimum decoding length of 50 and a maximum decoding length of 500. For our BART model with Longformer attention, we use a default attention context window size of 128. We train our BART-Long model for 5 epochs which approximately takes 6 hours. For BART-Long-Graph model we train for 8 epochs which approximately takes 8 hours. In terms of total number of trainable parameters, BART-Long has 447 million parameters and BART-Long-Graph has 463 million parameters.[9]

[4] We set $K = 1$ and $M = 1$ in all our experiments.

[5] https://duc.nist.gov/duc2004/

[6] https://pypi.org/project/pyrouge/

[7] The BART-Large model has 12 Transformer encoders and decoders.

[8] https://github.com/pytorch/fairseq

[9] Note that models based on pre-trained RoBERTa also have a similar number of parameters given that RoBERTa has 24 Transformer layers with 1024 hidden size, whereas BART-based models have 12 Transformer layers each on the encoder and decoder sides with 1024 hidden size.

Model	ROUGE-1	ROUGE-2	ROUGE-L	Average
PREVIOUS WORK				
PG-BRNN (Gehrmann et al., 2018)	43.77	15.38	20.84	26.66
HiMAP (Fabbri et al., 2019)	44.17	16.05	21.38	27.20
Flat Transformer	44.32	15.11	20.50	26.64
Hierarchical Transformer (Liu and Lapata, 2019)	42.36	15.27	22.08	26.57
RoBERTa + Transformer Decoder (Li et al., 2020)	44.26	16.22	22.37	27.62
GraphSum (Li et al., 2020)	45.02	16.69	22.50	28.07
GraphSum + RoBERTa (Li et al., 2020)	45.87	17.56	23.39	28.94
OUR MODELS				
BART-Long	48.54	18.56	23.78	30.29
BART-Long-Graph (500 tokens graph text)	49.03	**19.04**	**24.04**	30.70
BART-Long-Graph (1000 tokens graph text)	**49.24**	18.99	23.97	**30.73**

Table 1: Performance of various models on the Multi-News test set. We report the reproduced results of previous works provided by Li et al. (2020). We report 'summary-level' ROUGE-L scores following Fabbri et al. (2019).

5 Results

5.1 Main Results

In this section, we discuss the performance of various previous works on Multi-News and DUC-2004 datasets, and compare it with our proposed models.

Baseline Results. Table 1 presents the performance of various previous works. First, we report the results of PG-BRNN, HiMAP, and Flat Transformer following Fabbri et al. (2019). Next, we report the results of Hierarchical Transformers, RoBERTa+Transformer Decoder, and the variants of GraphSum models following Li et al. (2020). Some of these previous works use RoBERTa based encoder representations while training MDS models, hence achieving strong results. Note that all these models use 500 tokens for the source input.

BART-Long Results. Table 1 also presents the results on our BART-Long model as described in Sec. 3.2. Our BART-Long model is better than all previous works by a large margin, achieving a new state-of-the-art. This is because of two reasons: (1) The BART model has pre-trained encoder and decoder representations, whereas the previous works have pre-trained encoder-only models such as RoBERTa+Transformer Decoder and GraphSum + RoBERTa; (2) BART model has more number of parameters.[10] Apart from the performance, our BART-Long model has the advantage to encode longer parts of the input documents more efficiently than the traditional Transformer models or RoBERTa style pre-trained models (more results on this in Sec. 5.4; Table 4). This is because BART-Long model uses linear memory complexity via its local and global attention mechanism.

BART-Long-Graph Results. The results of our novel graph-based encodings into the BART model are shown in the last two rows of Table 1. Both these models perform statistically significantly better than our strong BART-Long baseline, where the main difference between these two models is the number of tokens used in the graph encoder.[11] Note that we construct our graph using $2,000$ tokens of input documents and use 500 or $1,000$ tokens of linearized graph text as input along with 500 tokens of input documents text.[12] We further calculated BERTScore (Zhang et al., 2020) for our models, and the F1 scores are 44.06, 44.52, and 44.64 for BART-Long, BART-Long-Graph with 500 tokens of graph, and BART-Long-Graph with $1,000$ tokens of graph, respectively. We have also tried pre-training the BART-Long-Graph with the criteria of decoding the original documents' text using noisy input, i.e., by removing some sentences randomly from linearized graph text and documents text. However, we do not see any significant improvement with this approach.

[10] Note that RoBERTa+Transformer decoder (Li et al., 2020) also has a similar number of parameters (see training details in Sec. 4).

[11] Our BART-Long-Graph with 500 tokens of graph text is significantly better than BART-Long baseline on all ROUGE metrics with $p<0.05$ based on ROUGE script's 95% confidence interval. Whereas, our BART-Long-Graph with $1,000$ tokens of graph text is statistically significantly better on ROUGE-1/2 metrics with $p<0.05$, and it achieves the best average ROUGE score.

[12] BART-Long with longer inputs perform on par w.r.t. BART-Long-Graph (see Table 4), however, they suffer from generating more extractive summaries, whereas our graph methods generate more abstractive summaries (see Table 8).

Model	R-1	R-2	R-SU
EXTRACTIVE METHODS			
MMR (Carbonell and Goldstein, 1998)	30.14	4.55	8.16
LexRank (Erkan and Radev, 2004)	35.56	7.87	11.86
TextRank (Mihalcea and Tarau, 2004)	33.16	6.13	10.16
ABSTRACTIVE METHODS TRAINED ON CNN/DAILY MAIL			
Copy-Transfomer (Gehrmann et al., 2018)	28.54	6.38	7.22
PG-BRNN (Gehrmann et al., 2018)	29.47	6.77	7.56
Hi-MAP (Fabbri et al., 2019)	35.78	8.90	11.43
PG-MMR (Lebanoff et al., 2018)	36.42	9.36	13.23
OUR ABSTRACTIVE METHODS TRAINED ON MULTI-NEWS			
BART-Long (500 text tokens)	33.82	8.09	10.53
BART-Long-Graph (500 text tok. + graph)	34.72	7.97	11.04

Table 2: ROUGE scores on DUC-2004 test-only setup.

	Informative	Coherent
BART-Long *is better*	17.0%	24.0%
BART-Long-Graph *is better*	25.5%	20.0%
None	57.5%	56.0%

Table 3: Human evaluation of informativeness and coherence of generated summaries.

5.2 Transfer Results on DUC-2004

We also evaluate our proposed models in a test-only transfer setup using the DUC-2004 multi-document summarization dataset. Table 2 presents the results on this dataset comparing our models with previous works. Our models perform better than some of the extractive summarization methods (TextRank and MMR). However, some of the previous works perform better than our models, but we cannot strictly compare with them since they are trained on CNN/Daily Mail dataset.[13] Comparing our baseline model and our model with graph encodings, we observe that graph encodings help improve the performance by 0.9 on ROUGE-1 and 0.5 on ROUGE-SU. This suggests that graph information is useful in transfer setups as well.

5.3 Human Evaluation

We conduct a human evaluation on Amazon MTurk to analyze the effect of adding graph input to the BART-Long model (setup details in Appendix B).

Informativeness and coherence: To evaluate how graph encodings impact informativeness and coherence of the generated summaries, we show human annotators pairs of summaries from the BART-Long model and the BART-Long-Graph model and ask them to indicate which one is more informative and which one is more coherent; definitions are listed in the Appendix B. There is also an option for choosing *None*. The summaries are labeled as A and B using random permutation; we also show the target summary from the test set for reference. We obtain judgments from two annotators on 200 examples from the Multi-News test set. Table 3 shows the results; *None* represents all cases where either both annotators picked *None* or the two annotators did not give the same answer. We observe that BART-Long-Graph summaries were picked as more informative by both judges 25.5% of the time, compared to 17% for the BART-Long model. The results are closer for coherence, with a slight disadvantage for the BART-Long-Graph model. We hypothesize that using graph information, which has a different structure than natural text, makes the summary less coherent.

Factual consistency: We evaluate factual consistency by highlighting single summary sentences and asking the annotators if it is consistent with the input articles. We ask three annotators to judge the factual consistency of the highlighted summary w.r.t. the articles on 200 outputs per model. For the BART-Long-Graph model, 72% of the summaries are judged as factually consistent by two or more annotators, compared to 68% for the BART-Long model. Frequently, news sources are hallucinated, e.g., "as reported by TMZ". This error accounts for 18 of the 136 errors of the BART-Long-Graph model and 19 of the 144 errors of the BART-Long model. More details in Appendix B.

Model	Input Length	R-1	R-2	R-L	Avg.
BART	500	49.22	18.88	23.88	30.66
BART-Long	500	48.54	18.56	23.78	30.29
BART-Long	1000	49.15	19.50	24.47	31.04
BART-Long	1500	48.79	19.14	24.16	30.70
BART-Long	2000	48.96	19.34	24.37	30.89

Table 4: Performance of BART models at various input lengths on the Multi-News dataset.

5.4 Ablations and Analyses

What is the effect of input documents length over the performance? Table 4 presents the performance comparison of BART with Longformer (BART-Long) over different input lengths. At the same input length, BART-Long performance is slightly lower than the BART model without Longformer attention, i.e., using full self-attention. This is expected as we replace the full self-attention with

[13] If we compare these models (e.g., Hi-MAP) on Multi-News dataset, our models perform much better (see Table 1).

Window Size	R-1	R-2	R-L	Avg.
32	48.39	18.75	23.68	30.27
64	48.93	19.28	24.25	30.82
128	48.96	19.34	24.37	30.89
256	49.47	19.94	24.82	31.41
512	49.43	19.86	24.77	31.35

Table 5: Performance of BART-Long model at various attention window sizes. We use Multi-News dataset with 1000 text tokens as input in this comparison to consider longer context window sizes.

Model	R-1	R-2	R-L	Avg.
BART-Long (BL)	48.54	18.56	23.78	30.29
BL-Graph-Only	44.91	14.19	20.46	26.52
BL-Graph-Concat	48.85	18.78	23.79	30.47
BL-Separate-Graph	49.03	19.04	24.04	30.70

Table 6: Performance of various graph encoding methods. We use 500 tokens of graph text.

local and global attention with lower memory footprint. More importantly, BART-Long can encode longer documents and can achieve better results which is evident from the results in Table 4.[14] Overall, we observe that the best results are achieved at a document length of $1,000$ tokens, and no further improvement for any input length greater than that.

What is the effect of attention context window size over the performance? We also compare the effect of various attention context window sizes in the local attention mechanism of BART-Long model over the summarization performance. Table 5 presents such ablation with attention context window sizes of 32, 64, 128, 256, and 512, on the Multi-News dataset with $1,000$ tokens input.[15] Here, we observe that performance linearly improves till certain context window size and then stays more or less similar. Note that in Table 1 we use an attention context window size of 128 to trade-off between memory and performance.

Different approaches of graph encodings. Table 6 presents the results on various graph encoding methods. First, we replace the original input with linearized graph text and we observe a significant drop in the performance ('BL-Graph-Only'; second row in Table 6). This suggests that documents' text as input is very important to achieve good results.

[14]Encoding more than 500 tokens is not feasible with full self-attention BART model due to high memory requirements.

[15]We use $1,000$ tokens instead of 500 tokens in this setup to effectively compare the 512 window size, otherwise it would be same as full self-attention with 500 tokens input.

Target Graph	R-1	R-2	R-L	Avg.
0 %	49.03	19.04	24.04	30.70
25 %	49.99	20.66	24.93	31.86
50 %	54.17	27.30	29.79	37.09
75 %	53.70	28.09	30.28	37.36
100 %	61.32	39.15	38.66	46.38

Table 7: Ablation of the performance of BART-Long-Graph model with 500 tokens input graph text over additionally using a varying percentage of target summary graph information.

Model	Density	LCS(%)	4-gr (%)
BL (2,000 text tokens)	15.5	68.8	55.0
BL (1,000 text tokens)	13.6	66.8	50.7
BL (500 text tokens)	11.6	62.5	44.4
BL (500 text tok. + graph)	10.0	59.3	41.3
Reference summaries	5.0	45.9	17.9

Table 8: Abstractiveness: Measuring lexical overlap between summaries and their inputs; lower numbers mean higher abstractiveness. Adding graphs increases abstractiveness.

Next, we concatenate the documents' text with linearized graph text and give it has input to the BART model ('BL-Graph-Concat') which achieves slightly better results over the baseline. However, when we add the linearized graph text as a separate graph encoder ('BL-Separate-Graph'; same as our 'BART-Long-Graph' model in Table 1), we achieve the best results.

How abstractive are the summaries? Abstractive summarizers generate surprisingly *extractive* summaries, copying large fragments unmodified from the input documents into the summaries (Weber et al., 2018; Pilault et al., 2020). We hypothesize that providing graph representations of the input can help the model abstract away from the specific lexical content of the input and generate summaries that are more abstractive. Table 8 shows the lexical overlap between the summaries and their inputs when truncating the input documents to different numbers of words, and when adding a graph representation of the input (truncated to 1k graph tokens). Density measures the expected length of the extractive fragment that any randomly chosen summary word belongs to (Grusky et al., 2018); LCS(%) is the length of the longest common subsequence divided by the length of the summary; and 4-gr(%) is the proportion of 4-grams in the summaries that are extracted from the input. We

BART-Long	Microsoft's acquisition of Nokia 's cell phone business is a big step in the company 's evolution into a devices and services company, but it 's not going to be the Apple of the mobile world. Nokia, which had a 35% market share of the cell phone market in 2003, made an operating profit of 5.48 billion Euros that year , according to the Wall Street Journal, but today 's sale price for the company — which includes 1.65 billion Euros in patents — is just 5.44 billion Euros. The acquisition is part of a larger effort by Microsoft to " move further away from the moribund world of the beige desktop and towards the sunlit world of smartphones and tablets ," writes Chris Cillizza at Forbes. " Owning the desktop (via windows) and building additional services on top, like office or search, has been vital for Microsoft 's strategy until now, " He writes. " As our interest shifts from the desktop to the tablet or smartphone , it'll be essential to Microsoft that it has a presence in the smartphone and tablet market. " Nokia will continue to operate under the Nokia brand , but the company will be renamed Microsoft mobile.
BART-Long-Graph	Microsoft is buying Nokia 's cell phone business for $ 8.5 billion , a price tag that includes $ 1.65 billion in patents, reports the Wall Street Journal. The move is part of Microsoft's plan to shift away from the desktop and toward smartphones and tablets , and the journal sees the move as " the latest acceleration of that strategy — to move further away from a moribund world of the beige desktop and towards the sunlit world of smartphones and tablet ." Nokia has been in trouble for a while now , notes TechCrunch, but the deal is a sign that the company is finally ready to move on from its mobile roots. "The acquisition of Nokia is the right move for Microsoft ," says one analyst . "It 's a step in the right direction." But the journal notes that the move could complicate Apple 's plans to buy Nokia , which it has been working on for some time.

Table 9: Examples of two summaries generated from the same input by the BART-Long and BART-Long-Graph models. Long extractive fragments are marked in red, shorter ones in orange and yellow. Summaries are re-capitalized and de-tokenized for better readability.

observe that longer text inputs make summaries more *extractive*, while adding a graph makes summaries more *abstractive*.[16]

Would better graph leads to better improvements? In order to answer how good is our graph construction approach, we choose to convert the target summary into a graph and use its linearized text as input to the model along with the original input documents' text and its linearized graph text. Table 7 presents such ablation where we linearly increase the amount of target graph information given as input, and we observe that using more target graph information leads to better performance. This suggests that a better way of including more salient information in the graph construction process could lead to a better summarization model.[17]

Qualitative analysis of output summaries. Table 9 presents generated summaries from two models, BART-Long and BART-Long-Graph. Both examples have the misattribution of source error as mentioned in Sec.5.3, motivating the need to improve factual consistency of abstractive summaries. The overlapped n-grams between the summary and the original source articles are highlighted in colors. Yellow and red stand for shorter and longer n-gram overlap, respectively. The visualizations show that BART-Long-Graph produces more abstractive summaries, as shown in Table 8, due to the fact that it incorporates triplet-based information that abstract away from the surface of the source articles.

Extra Ablations. We provide graph visualization of input documents in Appendix A.

6 Conclusion

We presented an efficient graph-enhanced approach to MDS that achieves state-of-the-art results on the Multi-News dataset using the pre-trained encoder-decoder Transformer model along with an efficient encoding mechanism. We also show improvements in a transfer-only setup on the DUC-2004 dataset. The graph encodings lead to summaries that are more abstractive. Human evaluation shows that they are also more informative and factually more consistent with their input documents. Finally, we present extensive ablations to better understand the usefulness of our method.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. This work was partially supported by NSF-CAREER Award 1846185, an Amazon ML Research Award, and a Microsoft PhD Fellowship.

[16] We observe similar trends when we add graph to longer or shorter text inputs.

[17] We 'randomly' choose $x\%$ of the target graph.

References

Elena Baralis, Luca Cagliero, Naeem Mahoto, and Alessandro Fiori. 2013. GraphSum: Discovering correlations among multiple terms for graph-based summarization. *Information Sciences*, 249:96–109.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336.

Anirban Dasgupta, Ravi Kumar, and Sujith Ravi. 2013. Summarization through submodularity and dispersion. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1014–1022.

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

Alexander R Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R Radev. 2019. Multi-News: a large-scale multi-document summarization dataset and abstractive hierarchical model. In *ACL*.

Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. 2019. Using local knowledge graph construction to scale seq2seq models to multi-document inputs. In *EMNLP*.

Sebastian Gehrmann, Yuntian Deng, and Alexander M Rush. 2018. Bottom-up abstractive summarization. In *EMNLP*.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *NAACL*.

Logan Lebanoff, Kaiqiang Song, and Fei Liu. 2018. Adapting the neural encoder-decoder framework from single to multi-document summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4131–4141.

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*.

Wei Li, Lei He, and Hai Zhuge. 2016. Abstractive News Summarization based on Event Semantic Link Network. In *Proc. of COLING*, pages 236–246.

Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, Haifeng Wang, and Junping Du. 2020. Leveraging graph to improve abstractive multi-document summarization. In *ACL*.

Kexin Liao, Logan Lebanoff, and Fei Liu. 2018. Abstract meaning representation for multi-document summarization. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1178–1190.

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

Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A Smith. 2015. Toward abstractive summarization using semantic representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086.

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

Yang Liu and Mirella Lapata. 2019. Hierarchical transformers for multi-document summarization. In *Proc. of ACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Inderjeet Mani and Eric Bloedorn. 1997. Multi-document summarization by graph search and matching. In *Proceedings of the National Conference on Artificial Intelligence*, pages 622–628. AAAI.

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

Paul Over and James Yen. 2004. An introduction to duc-2004. In *Proceedings of the 4th Document Understanding Conference (DUC 2004)*.

Jonathan Pilault, Raymond Li, Sandeep Subramanian, and Chris Pal. 2020. On extractive and abstractive neural document summarization with transformer language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9308–9319, Online. Association for Computational Linguistics.

Dragomir R Radev and Kathleen R McKeown. 1998. Generating natural language summaries from multiple on-line sources. *Computational Linguistics*, 24(3):470–500.

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised Open Information Extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 885–895, Stroudsburg, PA, USA. Association for Computational Linguistics.

Xiaojun Wan. 2008. An exploration of document impact on graph-based multi-document summarization. In *Proceedings of the 2008 conference on empirical methods in natural language processing*, pages 755–762.

Xiaojun Wan and Jianwu Yang. 2006. Improved Affinity Graph Based Multi-Document Summarization. In *Proc. of NAACL (short papers)*, pages 181–184.

Noah Weber, Leena Shekhar, Niranjan Balasubramanian, and Kyunghyun Cho. 2018. Controlling decoding for more abstractive summaries with copy-based networks. *arXiv preprint arXiv:1803.07038*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *ICLR*.

A Extra Ablations

Qualitative Analysis of Graphs. Each graph in Fig. 5 corresponds to an example in the Multi-News dataset with multiple input documents, where we convert them into a graph using our graph construction process as described in Sec. 3.3. We observe that these graphs are highly connected forming only a few clusters. We further remove clusters with only two nodes to create a very consolidated graphs. It is also worth noting that input documents with a total of $2,000$ tokens can be represented with less than 100 nodes (and their corresponding relations).

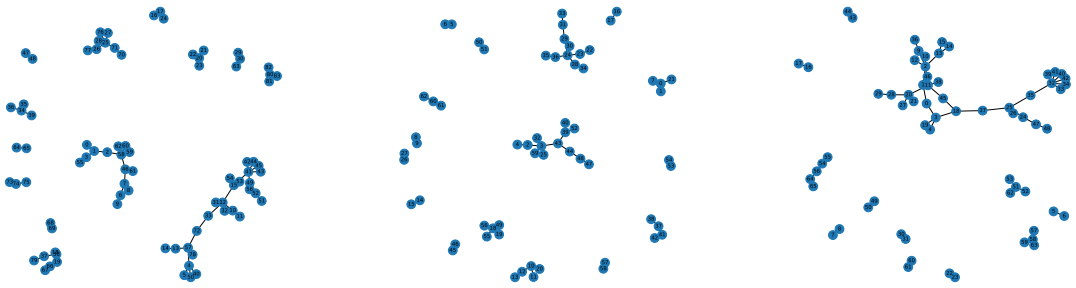
B Details on the Mechanical Turk Setup

For all our evaluations on Mechanical Turk (see Sec. 5.3 of the main paper), we first set up a short qualification test that can be taken by any worker from a country whose main language is English, who has completed 100 or more HITs so far with an acceptance rate of 95% or higher. The qualification test consists of just three questions from our factual consistency setup; two of which must be answered correctly, along with an explanation text (5 words or more) to explain when "not factually consistent" was chosen. 53% of workers who start the test provide answers to all three questions, and 27.6% of these answer at least two correctly and provide a reasonable explanation text, i.e., only 14.6% of the test takers are granted the qualification. The qualification enables workers to work on our factual consistency HITs as well as our HITs judging informativeness and coherence. The rate per HIT differs widely between the two tasks, as the factual consistency task can be done quickly, given the fact that a single summary sentence is evaluated, which is often extractive, and the related sentences in the article are highlighted. The factual consistency task pays \$0.07 per hour with a bonus of \$0.03; the informativeness and coherence task pays \$0.25 per hour with a bonus of \$0.50. Overall, this amounts to an average pay of \$12.50, incl. the bonus. The bonus is paid to workers who spend at least 10 seconds per HIT for the factual consistency task and 60 seconds per HIT for the informativeness and coherence task and who give short explanation texts for their decisions.

We give the following guidelines on deciding which summary is more informative or more coherent, respectively:

- *Informativeness*: The more informative summary is better at expressing the main points of the news story. It contains information that is more relevant and important. It has fewer unimportant details. Its content is more similar to the human-written summary.

- *Coherence*: The more coherent summary has better structure and flow, is easier to follow. The facts are presented in a more logical order.



(a) Example Graph-1 (b) Example Graph-2 (c) Example Graph-3

Figure 5: Visualization of input documents in a graph form, where nodes represent subject or object phrases and edges represent the relationship between them.