Entity-Aware Abstractive Multi-Document Summarization

Hao Zhou[1], **Weidong Ren**[1], **Gongshen Liu**[1]*, **Bo Su**[1], **Wei Lu**[2]
[1] School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University
[2] StatNLP Research Group, Singapore University of Technology and Design
{zhou1998,renweidong1997,lgshen,subo}@sjtu.edu.cn
luwei@sutd.edu.sg

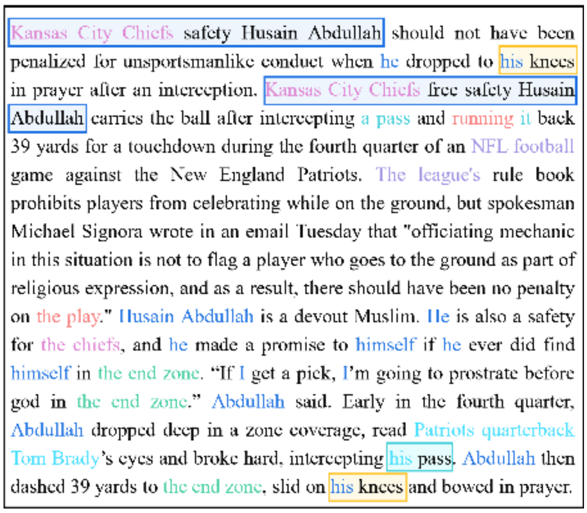
Abstract

Entities and their mentions convey significant semantic information in documents. In multi-document summarization, the same entity may appear across different documents. Capturing such cross-document entity information can be beneficial – intuitively, it allows the system to aggregate diverse useful information around the same entity for better summarization. In this paper, we present EMSum, an entity-aware model for abstractive multi-document summarization. Our model augments the classical Transformer-based encoder-decoder framework with a heterogeneous graph consisting of text units and entities as nodes, which allows rich cross-document information to be captured. In the decoding process, we design a novel two-level attention mechanism, allowing the model to deal with saliency and redundancy issues explicitly. Our model can also be used together with pre-trained language models, arriving at improved performance. We conduct comprehensive experiments on the standard datasets and the results show the effectiveness of our approach.

1 Introduction

Multi-document summarization aims at generating a short and informative summary across a set of topic-related documents. It is a task that can be more challenging than single-document summarization due to the presence of diverse and potentially conflicting information (Ma et al., 2020).

While significant progress has been made in single-document summarization, the mainstream sequence-to-sequence models, which can perform well on single-document summarization, often struggle with extracting salient information and handling redundancy in the presence of multiple, long documents. Thus, simply adopting models that were shown effective for single-document summarization to the multi-document setup may not lead to ideal results (Lebanoff et al., 2018; Zhang et al., 2018; Baumel et al., 2018).



Kansas City Chiefs safety Husain Abdullah should not have been penalized for unsportsmanlike conduct when he dropped to his knees in prayer after an interception. Kansas City Chiefs free safety Husain Abdullah carries the ball after intercepting a pass and running it back 39 yards for a touchdown during the fourth quarter of an NFL football game against the New England Patriots. The league's rule book prohibits players from celebrating while on the ground, but spokesman Michael Signora wrote in an email Tuesday that "officiating mechanic in this situation is not to flag a player who goes to the ground as part of religious expression, and as a result, there should have been no penalty on the play." Husain Abdullah is a devout Muslim. He is also a safety for the chiefs, and he made a promise to himself if he ever did find himself in the end zone. "If I get a pick, I'm going to prostrate before god in the end zone." Abdullah said. Early in the fourth quarter, Abdullah dropped deep in a zone coverage, read Patriots quarterback Tom Brady's eyes and broke hard, intercepting his pass. Abdullah then dashed 39 yards to the end zone, slid on his knees and bowed in prayer.

Figure 1: A sample article from MultiNews. We show the results of co-reference resolution. Mentions of the same entity are highlighted with the same color.

Several previous research efforts have shown that modeling cross-document relations is essential in multi-document summarization (Liu and Lapata, 2019a; Li et al., 2020). Such relations were shown useful in identifying the salient and redundant information from long documents, and can thus guide the summary generation process. However, while effective empirically, such approaches do not focus on explicitly modeling the underlying semantic information across documents.

Entities and their mentions convey rich semantic information, and can be significant in summarization, especially when a specific entity is the topic under discussion for a set of documents. As shown in Figure 1, entity mentions frequently appear in the input article, and are playing unique roles that contribute towards the coherence and conciseness of the text. We believe that entities can be regarded

[*] Corresponding author.

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 351–362
August 1–6, 2021. ©2021 Association for Computational Linguistics

as the indicator of saliency and can be used to reduce redundancy. This motivates us to propose an entity-aware abstractive multi-document summarization model that effectively encodes relations across documents with the help of entities, and explicitly solve the issues of saliency and redundancy.

Inspired by Wang et al. (2020a), we build a heterogeneous graph that consists of nodes that represent documents and entities. The entity nodes can serve as bridges that connect different documents – we can model the relations across documents through entity clusters. We apply the graph attention network (GAT) (Veličković et al., 2017) to enable information flow between nodes and iteratively update the node representations. In the decoding process, we design a novel two-level attention mechanism. The decoder first attends to the entities. Next, the attention weights of entities are incorporated with graph edge weights to guide the attention to the documents. Intuitively, the first stage indentifies the salient content in each decoding step. By considering the global interactions between entities and documents in the graph, the second stage is able to handle the redundancy issue. Experiments show that our model significantly improves the performance on several multi-document datasets. Further improvements can be made when our model is used together with the pre-trained language models.

Our contributions are as follows:

- We construct a heterogeneous graph network for multi-document summarization. The graph consists of document-level and entity-level nodes. To the best of our knowledge, we are the first to model the relations between documents and entities in one heterogeneous graph. Experiments show that exploiting entity nodes as the intermediary between documents can be more effective than exploiting other semantic units (e.g., words).
- We propose a novel two-level attention mechanism during the decoding process, solving the issues of saliency and redundancy explicitly. The mechanism can also reduce the computational cost, making it easier to process long inputs.
- Our model achieves state-of-the-art results on WikiSum and MultiNews. Extensive analysis including ablation studies show the effectiveness of our model.[1]

2 Related Work

2.1 Abstractive Document Summarization

Abstractive summarization is often regarded as the ultimate goal of document summarization research. Extractive summarization methods produce summaries that are semantically similar to the original documents. Thus, they may be able to achieve relatively high ROUGE scores (Lin, 2004). However, sentence-level extraction lacks flexibility and tends to produce redundant information. By contrast, the process of abstractive summarization is more similar to the human summarization process and requires more sophisticated natural language understanding and generation techniques. Traditional approaches to abstractive summarization can be divided into sentence fusion-based (Barzilay and McKeown, 2005; Filippova and Strube, 2008; Banerjee et al., 2015), paraphrasing-based (Bing et al., 2015; Cohn and Lapata, 2009) and information extraction-based (Li, 2015; Wang and Cardie, 2013; Pighin et al., 2014).

With the development of neural-based methods, abstractive methods achieved promising results on single document summarization (See et al., 2017; Paulus et al., 2018; Gehrmann et al., 2018; Li et al., 2018). More recently, due to the excellent performance on various text generation tasks, transformer-based methods become the mainstream approach for abstractive multi-document summarization, as well as pre-trained language models. Liu and Lapata (2019b) propose BertSUM for both extractive and abstractive summarization. Zhang et al. (2019) build low-level and high-level Berts for sentence and document understanding, respectively. Moreover, several general purpose sequence-to-sequence pre-trained models are proposed, such as T5 (Raffel et al., 2020) and BART (Lewis et al., 2019). They are further fine-tuned for the summarization task. Zhang et al. (2020) propose PEGASUS, in which they design a pre-training objective tailored for abstractive text summarization. Zou et al. (2020) present three sequence-to-sequence pre-training objectives by reinstating source text for abstractive summarization.

2.2 Graph-based Document Summarization

Graph-based methods have long been utilized for extractive summarization. Text units on graphs are ranked and selected as the most salient ones to be included in the summary. LexRank (Erkan and Radev, 2004) computes sentence salience based on

[1] Our code is at https://github.com/Oceandam/EMSum

the eigenvector centrality in the connectivity graph of inter-sentence cosine similarity. Wan (2008) further incorporate the document-level information and the sentence-to-document relationship into the graph-based ranking process. Christensen et al. (2013) build multi-document graphs to approximate the discourse relations across sentences based on indicators including discourse cues, deverbal nouns, co-reference and more.

For recent methods based on graph neural networks, Tan et al. (2017) propose a graph-based attention mechanism to identify salient sentences. Yasunaga et al. (2017) construct an approximate discourse graph based on discourse markers and entity links, then apply graph convolutional networks over the relation graph. Fan et al. (2019) construct a local knowledge graph, which is then linearized into a structured input sequence so that models can encode within the sequence-to-sequence setting. Huang et al. (2020) further design a graph encoder, which improves upon graph attention networks, to maintain the global context and local entities complementing each other. Li et al. (2020) utilize homogeneous graphs to capture cross-document relations and guide the summary generation process. However, Wang et al. (2020a) are the first to introduce different granularity levels of text nodes to construct heterogeneous graphs for extractive summarization. Our work is partly similar to theirs, but we construct heterogeneous graphs composed of text unit nodes and entity nodes for abstractive multi-document summarization.

2.3 Summarization with Additional Features

In addition to the direct application of the general sequence-to-sequence framework, researchers attempted to incorporate various features into summarization. Cao et al. (2018) extract actual fact descriptions from the source text and propose a dual-attention mechanism to force the generation conditioned on both the source text and the extracted fact descriptions. Sharma et al. (2019) take a pipeline method for single-document summarization which is composed of an entity-aware content selection module and a summary generation module. By contrast, our EMSum model is an end-to-end method for multi-document summarization. Gunel et al. (2020) inject structural world knowledge from Wikidata to a transformer-based model, enabling the model to be more fact-aware. Zhu et al. (2020) extract factual relations from the source texts to build a local knowledge graph and integrated it into the transformer-based model.

Apart from entity or fact information, there are several works that incorporate topic information into summarization model. Narayan et al. (2018) recommend an encoder associating each word with a topic vector capturing whether it is representative of the document's content, and a decoder where each word prediction is conditioned on a document topic vector. Zheng et al. (2019) propose to mine cross-document subtopics. In their work, sentence salience is estimated in a hierarchical way with subtopic salience and relative sentence salience. Perez-Beltrachini et al. (2019) explicitly model the topic structure of summaries, and utilize it to guide a structured convolutional decoder. Wang et al. (2020b) rearrange and further explore the semantics of the topic model and develope a friendly topic assistant for transfomer-based abstractive summarization models.

3 Model

Our model is illustrated in Figure 2, which follows the transformer-based encoder-decoder architecture (Vaswani et al., 2017). We modify the encoder with graph neural networks, so we can incorporate entity information and graph representations at the same time. We design a novel two-level decoding process to explicitly deal with the problem of saliency and redundancy.

3.1 Entity Cluster Extraction

Wang et al. (2020a) use words as semantic units in addition to sentence nodes, acting as the intermediary to enrich the relationships between sentences. However, we argue that word-level semantic units are too fine and will bring huge computational costs. For multi-document summarization, models are usually required to process tens of documents. The total number of words will be vast, which further causes a hindrance for the graph construction and message passing process. Therefore, we use entity clusters as more advanced semantic units. We utilize the co-reference resolution tool (Lee et al., 2017) from AllenNLP (Gardner et al., 2018) to extract entity clusters. Note that we perform extraction globally, which means we concatenate all the documents into one long document. We denote the extracted entity clusters as $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$, where $C_i = \{mention_1, mention_2, \ldots, mention_l\}$, and l is

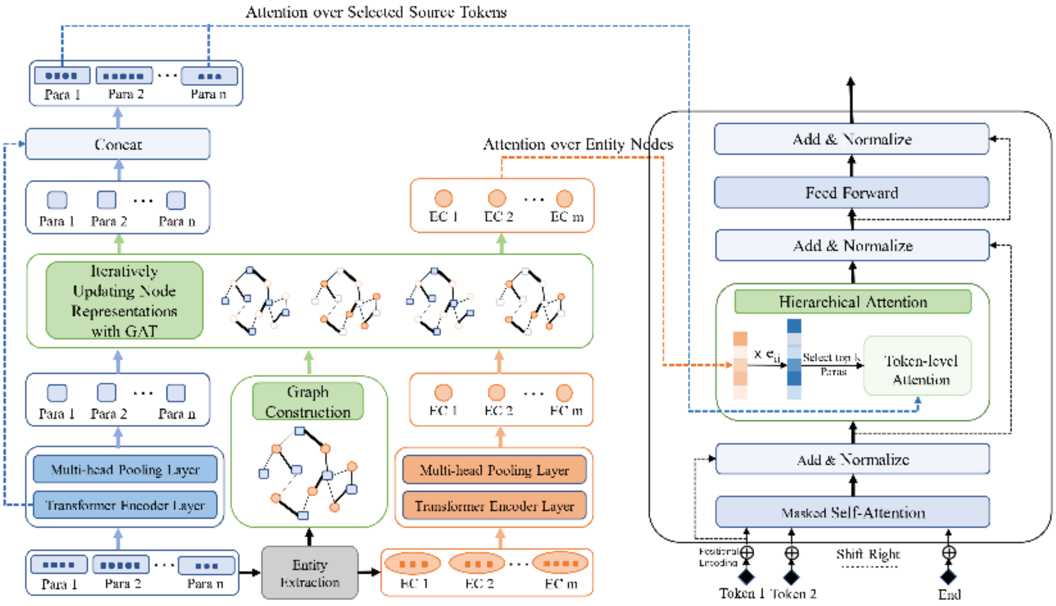


Figure 2: Overall architecture of our model.

the number of entity mentions in cluster C_i.

3.2 Graph Construction

Given a source document cluster $\mathcal{D}$, we firstly divide them into smaller semantic units $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$, such as paragraphs and sentences, depending on the characteristics of datasets. We then construct a heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. $\mathcal{V}$ includes paragraph nodes $\mathcal{V}_p$ and entity cluster nodes $\mathcal{V}_c$. $\mathcal{E}$ represents undirected edges between nodes. There exists no edge inside paragraph nodes or entity cluster nodes, but only between them. An edge which connects P_i and C_j means paragraph P_i contains an entity mention in C_j.

We would like to include more information in the graph. We get an occurrence matrix $\mathbf{E} \in \mathbb{R}^{m \times n}$ after extraction, where $e_{ij} \neq 0$ indicates P_i contains entity mentions in C_j for e_{ij} times. Based on $\mathbf{E}$, we further calculate the TF-IDF value matrix $\tilde{\mathbf{E}} \in \mathbb{R}^{m \times n}$ to model the importance of relationships between entity clusters and paragraphs.

3.3 Document Encoder

Paragraph Encoder Several token-level transformer encoding layers are stacked to encode contextual information within each paragraph. The transformer layer is the same as the vanilla transformer layer (Vaswani et al., 2017). Let x_w^0 be the input token vector. For the l-th transformer layer, the input is x_w^{l-1}, the hidden state is h_w^l, and the output is x_w^l.

$$h_w^l = \text{LayerNorm}(x_w^{l-1} + \text{MHAttn}(x_w^{l-1})) \quad (1)$$

$$x_w^l = \text{LayerNorm}(h_w^l + \text{FFN}(h_w^l)) \quad (2)$$

LayerNorm is the layer normalization operation (Ba et al., 2016). MHAttn is multi-head attention from Vaswani et al. (2017). FFN is a feed-forward network with ReLU as activation function. We take the output of last layer as token-level features. We use $\mathbf{H_{p_w}} \in \mathbb{R}^{n_w \times d_w}$ to denote the token-level feature matrix, where n_w is the total number of tokens in all paragraphs and d_w is the dimension of token embedding.

Multi-Head Pooling To obtain fixed length paragraph representations, we follow Liu and Lapata (2019a) to apply a weighted-pooling operation. The multi-head pooling mechanism calculates the weight distributions over tokens, allowing the model to flexibly encode paragraphs in different representational subspace by different head.

$$\mathbf{h_p} = \text{MHPool}(h_{w1}, h_{w2}, \ldots) \quad (3)$$

We use $\mathbf{H_p} \in \mathbb{R}^{n \times d_h}$ to denote the paragraph level feature matrix, where n is the number of paragraphs, d_h is the hidden size.

Entity Cluster Encoder We perform the same encoding process as the paragraph encoder to get entity clusters' representation, but without sharing parameters between the two encoders. We choose this method rather than additional entity embedding methods because we seek to model the relationship between paragraphs and entities in a unified semantic space. Note that we firstly remove pronouns and stopwords in entity mention clusters, which are common in co-reference resolution results but render little benefit for our semantic modeling. We use $\mathbf{H_{c_w}} \in \mathbb{R}^{m_w \times d_w}$ and $\mathbf{H_c} \in \mathbb{R}^{m \times d_h}$ to denote the token level feature matrix and cluster level feature matrix, respectively.

3.4 Graph Encoder

We use graph attention networks (GAT) (Veličković et al., 2017) to update the representations of semantic nodes. We use $i, j \in \{1, ..., (m+n)\}$ to denote an arbitrary node in graph, use $\mathbf{h}_i, \mathbf{h}_j \in \mathbb{R}^{d_h}$ to denote the node representations, and use $\mathcal{N}_i$ to denote the set of neighboring nodes of node i. The GAT layer is designed as follows:

$$z_{ij} = \text{LeakyReLU}(\mathbf{W}_a[\mathbf{W}_q\mathbf{h}_i; \mathbf{W}_k\mathbf{h}_j]) \quad (4)$$

$$\tilde{z}_{ij} = \tilde{e}_{ij} \times z_{ij} \quad (5)$$

$$\alpha_{ij} = \frac{\exp(\tilde{z}_{ij})}{\sum_{l \in \mathcal{N}_i} \exp(\tilde{z}_{il})} \quad (6)$$

$$\mathbf{u_i} = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}\mathbf{W}_v\mathbf{h}_j\right) \quad (7)$$

where $\mathbf{W}_a$, $\mathbf{W}_q$, $\mathbf{W}_k$, $\mathbf{W}_v$ are trainable weights, σ is the sigmoid function, $\tilde{e}_{ij}$ is the edge weight derived from TF-IDF value matrix $\tilde{\mathbf{E}}$.

We basically follow Wang et al. (2020a) to iteratively update node representations. They infuse the scalar edge weight $\tilde{e}_{ij}$ by simply discretizing the real values into integers, and then learn embeddings for such integers. That is how they map the weights to the multi-dimensional embedding space $\mathbf{e_{ij}} \in \mathbb{R}^{d_e}$. In this way, the information contained in the values needs to be learned by an additional embedding matrix. However, we argue that TF-IDF values themselves indicate the closeness between an entity cluster and a paragraph. Therefore, we directly incorporate the raw TF-IDF information into the GAT mechanism by modifying the attention weights using Equation 5.

We combine GAT with multi-head operation. We also add a residual connection to avoid gradient vanishing after several iterations:

$$\tilde{\mathbf{h}}_i = \mathbf{h}_i + \mathbf{u}_i \quad (8)$$

We use the above GAT layer and position-wise feed-forward layer to iteratively update the node representations. Each iteration contains a paragraph-to-entity and a entity-to-paragraph updating process. After iterating for t times, we concatenate $\tilde{\mathbf{H}}_\mathbf{p}$ to each corresponding input token vector, arriving at $\tilde{\mathbf{H}}_\mathbf{p_w} \in \mathbb{R}^{n_w \times (d_w + d_h)}$.

3.5 Entity-Aware Decoder with Two-level Attention

Under the setting of multi-document summarization, the input source documents may involve an extremely large number of word tokens. If the decoder needs to compute attention weights over all tokens, the cost would be very high and the attention could be dispersed. Our two-level decoding process firstly focuses on several centering entity cluster nodes, which can be regarded as indicators of saliency. The indicator restricts the token-level attention only to some of the paragraphs, which can further reduce redundancy than naively attending to all tokens. Different from Section 3.4, we use i and j to denote the entity node and paragraph node, respectively.

Attending the Entity Cluster Nodes At each decoding step, the state of decoder is $\mathbf{s}$, we compute attention scores over entity cluster nodes $\mathbf{c}_i$.

$$z_i = \mathbf{u}_0^T \text{LeakyReLU}([\mathbf{W}_{z_1}\mathbf{s}; \mathbf{W}_e\mathbf{c}_i]) \quad (9)$$

The entity nodes act as the intermediary between paragraph nodes. We incorporate z_i with edge weights $\tilde{e}_{ij}$ to enable the information flow between entity nodes and paragraph nodes by:

$$\tilde{z}_j = \sum_{i=1}^{m} z_i \times \tilde{e}_{ij} \quad (10)$$

$$\beta_j = \frac{\exp(\tilde{z}_j)}{\sum\limits_{l=1}^{m} \exp(\tilde{z}_l)} \quad (11)$$

Attending the Paragraph Tokens We select the top-k paragraph nodes with the highest attention score β_j. Then we apply the attention mechanism over the T_w tokens in the selected paragraphs.

$$z_{w_i} = \mathbf{u}_1^T \text{LeakyReLU}([\mathbf{W}_{z_2}\mathbf{s}_t; \mathbf{W}_w\tilde{h}_{w_i}]) \quad (12)$$

$$\gamma_{w_i} = \frac{\exp(z_{w_i})}{\sum\limits_{l=1}^{T_w} \exp(z_{w_l})} \quad (13)$$

For token w_i in paragraph P_j, we further modify γ_{w_i} by

$$\hat{\gamma}_{w_i} = \beta_j \times \gamma_{w_i} \tag{14}$$

Then the context vector $\mathbf{v}_t$ can be computed by:

$$\mathbf{v}_t = \sum_i \hat{\gamma}_{w_i} \tilde{h}_{w_i} \tag{15}$$

Token Prediction Context vectors, treated as salient contents summarized from sources, are concatenated with the decoder hidden state $\mathbf{s}_t$ to produce the vocabulary distribution:

$$P_{vocab} = \mathrm{Softmax}(\mathbf{W}_o[\mathbf{s}_t; \mathbf{v}_t]) \tag{16}$$

We use the weight-sharing strategy between the input embedding matrix and the matrix $\mathbf{W}_o$ to reuse linguistic knowledge (Paulus et al., 2018). We further add a copy mechanism as proposed by See et al. (2017).

3.6 Training

Our training process follows that of the traditional sequence-to-sequence modeling, with maximum likelihood estimation that minimizes:

$$\mathcal{L}_{seq} = -\frac{1}{|D|} \sum_{(\mathbf{y},\mathbf{x}) \in D} \log p(\mathbf{y}|\mathbf{x}; \theta) \tag{17}$$

where $\mathbf{x}$ and $\mathbf{y}$ are document-summary pairs from training set D, and θ are parameters to be learned.

3.7 Pre-trained LMs as Document Encoder

Our document encoder illustrated in section 3.3 can be replaced by a pre-trained language model such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). Pre-trained language models can be more effective on short inputs than training stacked transformer layers from scratch. We feed input tokens to a pre-trained language model and take the last layer output as token embeddings. Then a single-layer bidirectional LSTM is employed over token embeddings, producing token features. Finally, we perform the same multi-head pooling strategy to obtain paragraph representations.

4 Experiments

We use ROUGE scores to evaluate summarization quality automatically (Lin, 2004). We report different versions of the metric, based on overlaps of unigrams (ROUGE-1, R-1), bigrams (ROUGE-2, R-2) and the longest common subsequences (ROUGE-L, R-L).

4.1 Experimental Setup

We conduct experiments on two major datasets used in the literature of multi-document summarization, namely WikiSum (Liu et al., 2018) and MultiNews (Fabbri et al., 2019).

WikiSum Dataset Liu et al. (2018) treat the generation of Wikipedia section titles as a supervised multi-document summarization task. Liu and Lapata (2019a) crawled Wikipedia articles and source reference documents through the provided urls. They further split the long and messy source documents into multiple paragraphs by line-breaks and select the top-40 paragraphs as input for summarization systems. However, the top-40 dataset is quite heavy for entity extraction and co-reference resolution. Experiment shows that the ROUGE-L recall of top-20 paragraphs against the gold target text is 53.84, and top-40 is 60.42. So we choose to use the top-20 version of WikiSum dataset in order to find a balance between computational cost and the coverage of input content. We get 300,000 instances for training, 38,144 for validation and 38,205 for test. On average, each paragraph has 70.1 tokens, and target sumamry has 139.4 tokens. We then perform entity cluster extraction on the top-20 WikiSum dataset. For each instance, we get 23.7 clusters on average and each cluster has 10.2 tokens on average.

MultiNews Dataset Introduced by Fabbri et al. (2019), MultiNews consists of news articles and hand-written summaries. The source articles come from a diverse set of news sources, over 1,500 sites. Following their experimental settings, we get 44,972 instances for training, 5,622 for validation and 5,622 for test. Different from the WikiSum dataset, each source article only contains 2.8 paragraphs and 21.6 sentences on average, thus we choose to build graph on sentence level rather than paragraph level for this dataset. For each instance, we get 13.3 clusters on average and each cluster has 9.9 tokens on average.

Hyperparameters We set the number of our vanilla Transformer encoding layers as 6, the hidden size as 256 and the number of heads as 8, while the hidden size of feed-forward layers is 1,024. We truncate the length of input paragraphs and entity clusters to 100 and 50 tokens, respectively. In the multi-head pooling layer, the number of heads is 8. In the graph encoding process, each layer has 8 heads and the hidden size is 256. We select the

	Model	R-1	R-2	R-L
Ext	Lead†	38.22	16.85	26.89
	LexRank†	36.12	11.67	22.52
Abs	FT†	40.56	25.35	34.73
	FT+R†	42.05	27.00	36.56
	T-DMCA†	40.77	25.60	34.90
	HT†	41.53	26.52	35.76
	GraphSum†	42.63	27.70	36.97
	GraphSum+R†	**42.99**	27.83	37.36
	FT+R-20*	39.79	24.39	33.87
	T-DMCA-20*	38.64	21.25	29.77
	HT-20*	37.46	24.71	33.36
	EMSum	42.40	27.97	37.28
	EMSum+R	42.93	**28.11**	**38.19**

Table 1: Evaluation Results on WikiSum using ROUGE scores. The results of models with '†' are taken from Li et al. (2020). '∗' indicates the results are obtained by running the released code. Model name with suffix '+R' means RoBERTa is used. 'Ext' means extractive methods, 'Abs' means abstractive methods.

	Model	R-1	R-2	R-L
Ext	LexRank†	41.77	s13.81	37.87
	HeterSumGraph†	46.05	16.35	**42.08**
Abs	FT*	43.28	14.59	20.39
	FT+R*	43.10	15.32	21.66
	HT*	42.03	15.18	22.79
	GraphSum†	45.02	16.69	22.50
	GraphSum+R†	46.07	17.42	23.21
	EMSum	45.57	17.71	26.43
	EMSum+R	**46.89**	**18.26**	27.55

Table 2: Evaluation Results on MultiNews using ROUGE scores. The results of models with '†' are taken from Li et al. (2020) or Wang et al. (2020a). '*' indicates the results are obtained by running the released code.

number of iterations $t = 2$ based on the performance. We use dropout with probability 0.1 before all linear layers and label smoothing (Szegedy et al., 2016) with smoothing factor 0.1. We train our model for 200,000 steps with gradient accumulation every four steps. During decoding we apply beam search with beam size 5 and length penalty (Wu et al., 2016) with factor 0.4.

For models with pre-trained LMs, we choose the base version of RoBERTa. We follow Liu and Lapata (2019b), employing two Adam optimizers (Kingma and Ba, 2015) for the pre-trained part and other parts, with $\beta_1 = 0.9$, $\beta_2 = 0.998$. For the pre-trained part, the learning rate and warmup steps are set as 0.002 and 20,000, while for other parts are 0.2 and 8,000, respectively.

4.2 Baseline Models

We choose a series of Transformer-based models for comparison due to their excellent performance. Flat Transformer (**FT**) is a 6-layer encoder-decoder model. The title and ranked paragraphs were concatenated and truncated to 800 tokens. Transformer Decoder with Memory Compressed Attention model (**T-DMCA**) is proposed by Liu et al. (2018) with the WikiSum dataset. They use a Transformer decoder but apply a convolutional layer to compress the key and value in self-attention. Moreover, we choose Hierarchical Transformer (**HT**) proposed by Liu and Lapata (2019a), **GraphSum** proposed by Li et al. (2020), and **HeterSumGraph** proposed by Wang et al. (2020a) for comparisons. We have introduced them in Section 2.

4.3 Results

Results on WikiSum Table 1 summarizes the evaluation results on the WikiSum dataset. The first block shows the baseline model Lead and LexRank (Erkan and Radev, 2004), which are extractive methods. The second block shows the results of abstractive models introduced in Section 4.2. We report their results following Li et al. (2020). The last block shows the results of some abstractive models and our model, but such models are fed with 20 top-ranked paragraphs as input.

The results show that if we limit the number of input paragraphs to 20, ROUGE score of all models will drop by about 2 points. We believe this is because the lower-ranked paragraphs can still provide information anyway.

Our model EMSum performs the best under the top-20 setting. Compared to the reported results of GraphSum (which used top 40 documents), EMSum achieves improvements on ROUGE-2 and ROUGE-L, even though EMSum takes shorter source documents as input. The gap between EMSum and GraphSum on ROUGE-1 score is 0.23 (42.40 vs 42.63). Considering all these three metrics together, the results show the effectiveness of our model.

For models combined with pre-trained LMs, the results show that EMSum+RoBERTa further improves the summarization performance on all metrics over EMSum. The improvements over GraphSum+RoBERTa are 0.28 on ROUGE-2 and 0.83 on ROUGE-L, also showing the effectiveness of our model even in the presence of pre-trained LMs.

Results on MultiNews Table 2 summarizes the evaluation results on the MultiNews dataset. Similarly, the first block shows two extractive baselines LexRank, and HeterSumGraph. The second block shows the abstractive methods. We report the results of FT, HT and GraphSum following Li et al. (2020). The last block shows the results of our models. We can see that EMSum outperforms GraphSum and EMSum+RoBERTa outperforms GraphSum+RoBERTa. HeterSumGraph is a extractive method so it achieves better ROUGE-L score. However, our model still achieves higher ROUGE-1 and ROUGE-2 scores than HeterSumGraph. Overall, the results demonstrate the effectiveness of our model on different types of corpora.

4.4 Analysis

We further conduct experiments to analyze the effects of the number of iterations and the number of paragraphs selected for attention. We also conduct ablation studies to validate the effectiveness of different components of our model.

The Number of Iterations We investigate how the number of iterations t influences the performance of our model. To this end, we conduct experiments on WikiSum dataset when $t = 1, 2, 3, 4$. The first block in Table 3 shows the results. Intuitively, the more iterations the graph is updated, the more information is flowed across the nodes. However, the results show us that $t = 3, 4$ outperforms $t = 2$ on ROUGE-L and the overall performance $\tilde{R}$ fluctuates very little. We argue the performance is limited by the number of introduced parameters. Therefore we choose $t = 2$ finally.

The Number of Paragraphs Selected for Attention At each decoding step, our two-level attention mechanism firstly computes weights over entity nodes to identify the most salient parts of source documents. The attention weights over the entire long token sequence may be sparse. So we need to figure out how much salient information is enough for our model, namely the proper value of k. We conduct experiments on WikiSum dataset when $k = 5, 10, 15, 20$. As the results in the second block of Table 3 show, when $k = 5$, the number of attended paragraphs is relatively small thereby degrading performance heavily. When $k = 20$, that means we do not perform any cut-off but only modify the paragraph attention weights with the entity attention weights, so the performance is also reduced. When $k = 10, 15$, the cut-off strategy works and boosts the performance. Finally, we choose $k = 10$ because it performs the best.

k	t	R-1	R-2	R-L	$\tilde{\text{R}}$
10	1	41.61	27.74	37.59	35.65
	2	42.93	28.11	38.19	**36.20**
	3	42.58	27.29	38.50	36.12
	4	42.51	27.18	38.76	36.15
5	2	37.75	21.87	31.44	30.35
10		42.93	28.11	38.19	**36.20**
15		42.26	27.36	38.28	35.97
20		40.33	26.17	36.40	34.30

Table 3: Results on different number of iterations t and different number of paragraphs k for attention on WikiSum dataset. $\tilde{\text{R}}$ is the mean of R-1, R-2 and R-L.

Model	R-1	R-2	R-L
EMSum	42.40	27.97	37.28
w/o graph enc	39.47	25.18	29.93
w/o two-level attn	40.51	25.31	31.21

Table 4: Ablation study of our model on WikiSum.

Ablation Study To validate the effectiveness of individual components such as graph encoder module and two-level attention module, we conduct experiments of ablation studies. For experiments without graph encoder module, we simply fix the entity cluster representation and paragraph representation after the multi-head pooling layer. For experiments without two-level attention, we apply token-level attention directly, but attend to the entity cluster representation additionally, which is a naive way to incorporate entity information. Table 4 shows the results. The results show the effectiveness of our new introduced module. Incorporating entity information to construct a heterogeneous graph network enables better information flowing between text nodes, and our design of the novel two-level attention mechanism in this task is indeed playing an important role towards the overall effectiveness of our approach.

4.5 Human Evaluation

We further employ human evaluation to assess model performance. We randomly sampled 20 documents-summary pairs from the WikiSum test set and 20 from the MultiNews test set, and invited 3 participants to assess the outputs of different models independently. Following criteria used by previous work (Liu and Lapata, 2019a), the evaluation score takes three aspects into account: (1) Informativeness: does the summary include salient

Dataset	Model	Rating
WikiSum	FT	-0.517
	T-DMCA	-0.117
	HT	0.250
	EMSum	0.383
MultiNews	FT	-0.650
	T-DMCA	-0.033
	HS	0.317
	EMSum	0.367

Table 5: Human evaluation results on summary quality rating. FT, T-DMCA, HT, HS are baseline models explained in Section 4.2.

parts of the input? (2) Fluency: Is the summary fluent and grammatical? (3) Succinctness: does redundancy occur in the summary? We used Best-Worst Scaling (Louviere et al., 2015) because it has been shown to produce more reliable results than rating scales (Kiritchenko and Mohammad, 2017). Annotators are presented with the gold summary and summaries generated from 3 out of 4 systems and decide which summary is the best and which is the worst based on the criteria mentioned above. The rating of each system was computed as the percentage of times it was chosen as best minus the times it was selected as worst. Ratings range from -1 (worst) to 1 (best).

On the WikiSum dataset, we choose FT, T-DMCA, HT, EMSum and conduct human evaluation to compare their performance. On the MultiNews dataset, we choose FT, T-DMCA, HS, together with EMSum. The results are shown in Table 5. These results show that our EMSum model is able to generate summaries of higher quality than other models and further show the effectiveness of our proposed approach.

5 Conclusion

In this paper, we propose an entity-aware multi-document summarization model. We introduce entity nodes in addition to text unit nodes to construct a heterogeneous graph, helping our model capture complicated relations between text units. We also introduce a decoder with a two-level attention mechanism, which firstly attends to the entity nodes, where the attention weights are then subsequently utilized to guide the attention to the text units. With such a novel design, our model is able to deal with the problems of saliency and redundancy explicitly. Moreover, like other Transformer-based models, our model can be easily integrated with pre-trained language models for improved results. Experiments on standard datasets show the effectiveness of our model.

In the future, we would like to explore other approaches such as reinforcement learning based methods (Sharma et al., 2019) to further improve the summary quality in the context of multi-document summarization. We would also like to apply our method to other tasks such as multi-document question answering (Joshi et al., 2017).

Acknowledgments

We thank all the anonymous reviewers for their valuable suggestions. This research work was supported by the National Natural Science Foundation of China (Grant No.61772337, U1736207). Part of this work was done when Wei Lu was a visiting Professor at SJTU.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Siddhartha Banerjee, Prasenjit Mitra, and Kazunari Sugiyama. 2015. Multi-document abstractive summarization using ilp based multi-sentence compression. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 1208–1214.

Regina Barzilay and Kathleen R McKeown. 2005. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

Tal Baumel, Matan Eyal, and Michael Elhadad. 2018. Query focused abstractive summarization: Incorporating query relevance, multi-document coverage, and summary length constraints into seq2seq models. *arXiv preprint arXiv:1801.07704*.

Lidong Bing, Piji Li, Yi Liao, Wai Lam, Weiwei Guo, and Rebecca Passonneau. 2015. Abstractive multi-document summarization via phrase selection and merging. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1587–1597, Beijing, China. Association for Computational Linguistics.

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

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *Proceedings of the 2013 Conference of the North American Chapter of* *the Association for Computational Linguistics: Human Language Technologies*, pages 1163–1173, Atlanta, Georgia. Association for Computational Linguistics.

Trevor Anthony Cohn and Mirella Lapata. 2009. Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, 34:637–674.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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 Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084, Florence, Italy. Association for Computational Linguistics.

Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. 2019. Using local knowledge graph construction to scale Seq2Seq models to multi-document inputs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4186–4196, Hong Kong, China. Association for Computational Linguistics.

Katja Filippova and Michael Strube. 2008. Sentence fusion via dependency graph compression. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 177–185.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109, Brussels, Belgium. Association for Computational Linguistics.

Beliz Gunel, Chenguang Zhu, Michael Zeng, and Xuedong Huang. 2020. Mind the facts: Knowledge-boosted coherent abstractive text summarization. *arXiv preprint arXiv:2006.15435*.

Luyang Huang, Lingfei Wu, and Lu Wang. 2020. Knowledge graph-augmented abstractive summarization with semantic-driven cloze reward. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5094–5107, Online. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic gradient descent. In *ICLR: International Conference on Learning Representations*.

Svetlana Kiritchenko and Saif Mohammad. 2017. Best-worst scaling more reliable than rating scales: A case study on sentiment intensity annotation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 465–470, Vancouver, Canada. Association for Computational Linguistics.

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, Brussels, Belgium. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Wei Li. 2015. Abstractive multi-document summarization with semantic information extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1908–1913, Lisbon, Portugal. Association for Computational Linguistics.

Wei Li, Xinyan Xiao, Jiachen Liu, Hua Wu, Haifeng Wang, and Junping Du. 2020. 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, Xinyan Xiao, Yajuan Lyu, and Yuanzhuo Wang. 2018. Improving neural abstractive document summarization with structural regularization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4078–4087, Brussels, Belgium. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

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

Yang Liu and Mirella Lapata. 2019a. Hierarchical transformers for multi-document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5070–5081, Florence, Italy. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019b. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

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*.

Jordan J Louviere, Terry N Flynn, and Anthony Alfred John Marley. 2015. *Best-worst scaling: Theory, methods and applications*. Cambridge University Press.

Congbo Ma, Wei Emma Zhang, Mingyu Guo, Hu Wang, and Quan Z Sheng. 2020. Multi-document summarization via deep learning techniques: A survey. *arXiv preprint arXiv:2011.04843*.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*.

Laura Perez-Beltrachini, Yang Liu, and Mirella Lapata. 2019. Generating summaries with topic templates and structured convolutional decoders. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5107–5116, Florence, Italy. Association for Computational Linguistics.

Daniele Pighin, Marco Cornolti, Enrique Alfonseca, and Katja Filippova. 2014. Modelling events through memory-based, open-IE patterns for abstractive summarization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 892–901, Baltimore, Maryland. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Eva Sharma, Luyang Huang, Zhe Hu, and Lu Wang. 2019. An entity-driven framework for abstractive summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3280–3291, Hong Kong, China. Association for Computational Linguistics.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1181, Vancouver, Canada. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30:5998–6008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

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, Honolulu, Hawaii. Association for Computational Linguistics.

Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuanjing Huang. 2020a. Heterogeneous graph neural networks for extractive document summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6209–6219, Online. Association for Computational Linguistics.

Lu Wang and Claire Cardie. 2013. Domain-independent abstract generation for focused meeting summarization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1395–1405, Sofia, Bulgaria. Association for Computational Linguistics.

Zhengjue Wang, Zhibin Duan, Hao Zhang, Chaojie Wang, Long Tian, Bo Chen, and Mingyuan Zhou. 2020b. Friendly topic assistant for transformer based abstractive summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 485–497, Online. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. 2017. Graph-based neural multi-document summarization. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 452–462, Vancouver, Canada. Association for Computational Linguistics.

Jianmin Zhang, Jiwei Tan, and Xiaojun Wan. 2018. Towards a neural network approach to abstractive multi-document summarization. *arXiv preprint arXiv:1804.09010*.

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

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069, Florence, Italy. Association for Computational Linguistics.

Xin Zheng, Aixin Sun, Jing Li, and Karthik Muthuswamy. 2019. Subtopic-driven multi-document summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3153–3162, Hong Kong, China. Association for Computational Linguistics.

Chenguang Zhu, William Hinthorn, Ruochen Xu, Qingkai Zeng, Michael Zeng, Xuedong Huang, and Meng Jiang. 2020. Boosting factual correctness of abstractive summarization with knowledge graph. *arXiv preprint arXiv:2003.08612*.

Yanyan Zou, Xingxing Zhang, Wei Lu, Furu Wei, and Ming Zhou. 2020. Pre-training for abstractive document summarization by reinstating source text. In *Proc. of EMNLP*.