Heterogeneous Graph Neural Networks for Extractive Document Summarization

Danqing Wang[*], **Pengfei Liu**[*], **Yining Zheng, Xipeng Qiu**[†], **Xuanjing Huang**
Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
School of Computer Science, Fudan University
825 Zhangheng Road, Shanghai, China
{dqwang18,pfliu14,ynzheng19,xpqiu,xjhuang}@fudan.edu.cn

Abstract

As a crucial step in extractive document summarization, learning cross-sentence relations has been explored by a plethora of approaches. An intuitive way is to put them in the graph-based neural network, which has a more complex structure for capturing inter-sentence relationships. In this paper, we present a *heterogeneous graph-based neural network* for extractive summarization (HETERSUMGRAPH), which contains semantic nodes of different granularity levels apart from sentences. These additional nodes act as the intermediary between sentences and enrich the cross-sentence relations. Besides, our graph structure is flexible in natural extension from a single-document setting to multi-document via introducing document nodes. To our knowledge, we are the first one to introduce different types of nodes into graph-based neural networks for extractive document summarization and perform a comprehensive qualitative analysis to investigate their benefits. The code will be released on Github[1].

1 Introduction

Extractive document summarization aims to extract relevant sentences from the original documents and reorganize them as the summary. Recent years have seen a resounding success in the use of deep neural networks on this task (Cheng and Lapata, 2016; Narayan et al., 2018; Arumae and Liu, 2018; Zhong et al., 2019a; Liu and Lapata, 2019b). These existing models mainly follow the encoder-decoder framework in which each sentence will be encoded by neural components with different forms.

To effectively extract the summary-worthy sentences from a document, a core step is to model the cross-sentence relations. Most current models capture cross-sentence relations with recurrent neural networks (RNNs) (Cheng and Lapata, 2016; Nallapati et al., 2017; Zhou et al., 2018). However, RNNs-based models are usually hard to capture sentence-level long-distance dependency, especially in the case of the long document or multi-documents. One more intuitive way is to model the relations of sentences using the graph structure. Nevertheless, it is challenging to find an effective graph structure for summarization. Efforts have been made in various ways. Early traditional work makes use of inter-sentence cosine similarity to build the connectivity graph like LexRank (Erkan and Radev, 2004) and TextRank (Mihalcea and Tarau, 2004). Recently, some works account for discourse inter-sentential relationships when building summarization graphs, such as the Approximate Discourse Graph (ADG) with sentence personalization features (Yasunaga et al., 2017) and Rhetorical Structure Theory (RST) graph (Xu et al., 2019). However, they usually rely on external tools and need to take account of the error propagation problem. A more straightforward way is to create a sentence-level fully-connected graph. To some extent, the Transformer encoder (Vaswani et al., 2017) used in recent work(Zhong et al., 2019a; Liu and Lapata, 2019b) can be classified into this type, which learns the pairwise interaction between sentences. Despite their success, how to construct an effective graph structure for summarization remains an open question.

In this paper, we propose a heterogeneous graph network for extractive summarization. Instead of solely building graphs on sentence-level nodes, we introduce more semantic units as additional nodes in the graph to enrich the relationships between sentences. These additional nodes act as the intermediary that connects sentences. Namely, each additional node can be viewed as a special rela-

[*] These two authors contributed equally.
[†] Corresponding author.
[1] https://github.com/brxx122/HeterSUMGraph

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 6209–6219
July 5 - 10, 2020.

tionship between sentences containing it. During the massage passing over the heterogeneous graph, these additional nodes will be iteratively updated as well as sentence nodes.

Although more advanced features can be used (e.g., entities or topics), for simplicity, we use words as the semantic units in this paper. Each sentence is connected to its contained words. There are no direct edges for all the sentence pairs and word pairs. The constructed heterogeneous *word-sentence* graph has the following advantages: (a) Different sentences can interact with each other in consideration of the explicit overlapping word information. (b) The word nodes can also aggregate information from sentences and get updated. Unlike ours, existing models usually keep the words unchanged as the embedding layer. (c) Different granularities of information can be fully used through multiple message passing processes. (d) Our heterogeneous graph network is expandable for more types of nodes. For example, we can introduce document nodes for multi-document summarization.

We highlight our contributions as follows:

(1) To our knowledge, we are the first one to construct a heterogeneous graph network for extractive document summarization to model the relations between sentences, which contains not only sentence nodes but also other semantic units. Although we just use word nodes in this paper, more superior semantic units (e.g. entities) can be incorporated.

(2) Our proposed framework is very flexible in extension that can be easily adapt from single-document to multi-document summarization tasks.

(3) Our model can outperform all existing competitors on three benchmark datasets without the pre-trained language models[2]. Ablation studies and qualitative analysis show the effectiveness of our models.

2 Related Work

Extractive Document Summarization With the development of neural networks, great progress has been made in extractive document summarization. Most of them focus on the encoder-decoder framework and use recurrent neural networks (Cheng and Lapata, 2016; Nallapati et al., 2017; Zhou et al., 2018) or Transformer encoders (Zhong et al., 2019b; Wang et al., 2019a) for the sentential encoding. Recently, pre-trained language models are also applied in summarization for contextual word representations (Zhong et al., 2019a; Liu and Lapata, 2019b; Xu et al., 2019; Zhong et al., 2020).

Another intuitive structure for extractive summarization is the graph, which can better utilize the statistical or linguistic information between sentences. Early works focus on document graphs constructed with the content similarity among sentences, like LexRank (Erkan and Radev, 2004) and TextRank (Mihalcea and Tarau, 2004). Some recent works aim to incorporate a relational priori into the encoder by graph neural networks (GNNs) (Yasunaga et al., 2017; Xu et al., 2019). Methodologically, these works only use one type of nodes, which formulate each document as a homogeneous graph.

Heterogeneous Graph for NLP Graph neural networks and their associated learning methods (i.e. message passing (Gilmer et al., 2017), self-attention (Velickovic et al., 2017)) are originally designed for the homogeneous graph where the whole graph shares the same type of nodes. However, the graph in the real-world application usually comes with multiple types of nodes (Shi et al., 2016), namely the heterogeneous graph. To model these structures, recent works have made preliminary exploration. Tu et al. (2019) introduced a heterogeneous graph neural network to encode documents, entities and candidates together for multi-hop reading comprehension. Linmei et al. (2019) focused on semi-supervised short text classification and constructed a topic-entity heterogeneous neural graph.

For summarization, Wei (2012) proposes a heterogeneous graph consisting of topic, word and sentence nodes and uses the markov chain model for the iterative update. Wang et al. (2019b) modify TextRank for their graph with keywords and sentences and thus put forward HeteroRank. Inspired by the success of the heterogeneous graph-based neural network on other NLP tasks, we introduce it to extractive text summarization to learn a better node representation.

3 Methodology

Given a document $D = \{s_1, \cdots, s_n\}$ with n sentences, we can formulate extractive summarization as a sequence labeling task as (Narayan et al., 2018;

[2]Since our proposed model is orthogonal to the methods that using pre-trained models, we believe our model can be further boosted by taking the pre-trained models to initialize the node representations, which we reserve for the future.

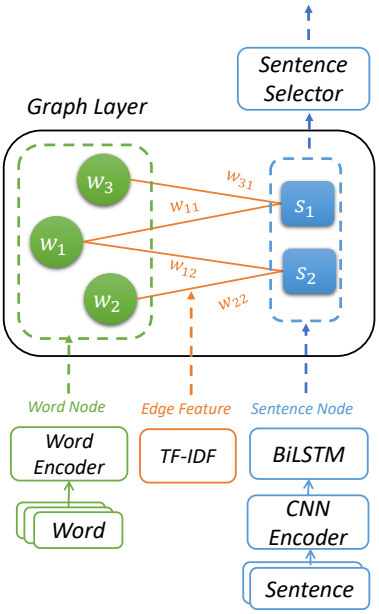

Figure 1: Model Overview. The framework consists of three major modules: *graph initializers*, the *heterogeneous graph layer* and the *sentence selector*. Green circles and blue boxes represent word and sentence nodes respectively. Orange solid lines denote the edge feature (TF-IDF) between word and sentence nodes and the thicknesses indicate the weight. The representations of sentence nodes will be finally used for summary selection.

Liu and Lapata, 2019b). Our goal is to predict a sequence of labels $y_1, \cdots, y_n$ ($y_i \in \{0, 1\}$) for sentences, where $y_i = 1$ represents the i-th sentence should be included in the summaries. The ground truth labels, which we call ORACLE, is extracted using the greedy approach introduced by Nallapati et al. (2016) with the automatic metrics ROUGE (Lin and Hovy, 2003).

Generally speaking, our heterogeneous summarization graph consists of two types of nodes: basic semantic nodes (e.g. words, concepts, etc.) as relay nodes and other units of discourse (e.g. phrases, sentences, documents, etc.) as supernodes. Each supernode connects with basic nodes contained in it and takes the importance of the relation as their edge feature. Thus, high-level discourse nodes can establish relationships between each other via basic nodes.

In this paper, we use words as the basic semantic nodes for simplicity. HETERSUMGRAPH in Section 3.1 is a special case which only contains one type of supernodes (sentences) for classification, while HETERDOCSUMGRAPH in Section 3.5 use two (documents and sentences). Based on our framework, other types of supernodes (such as paragraphs) can also be introduced and the only difference lies in the graph structure.

3.1 Document as a Heterogeneous Graph

Given a graph $G = \{V, E\}$, where V stands for a node set and E represents edges between nodes, our undirected heterogeneous graph can be formally defined as $V = V_w \cup V_s$ and $E = \{e_{11}, \cdots, e_{mn}\}$. Here, $V_w = \{w_1, \cdots, w_m\}$ denotes m unique words of the document and $V_s = \{s_1, \cdots, s_n\}$ corresponds to the n sentences in the document. E is a real-value edge weight matrix and $e_{ij} \neq 0$ ($i \in \{1, \cdots, m\}, j \in \{1, \cdots, n\}$) indicates the j-th sentence contains the i-th word.

Figure 1 presents the overview of our model, which mainly consists of three parts: *graph initializers* for nodes and edges, the *heterogeneous graph layer* and the *sentence selector*. The initializers first create nodes and edges and encode them for the document graph. Then the heterogeneous graph updates these node representations by iteratively passing messages between word and sentence nodes via Graph Attention Network (GAT) (Velickovic et al., 2017). Finally, the representations of sentence nodes are extracted to predict labels for summaries.

3.2 Graph Initializers

Let $\mathbf{X}_w \in \mathbb{R}^{m \times d_w}$ and $\mathbf{X}_s \in \mathbb{R}^{n \times d_s}$ represent the input feature matrix of word and sentence nodes respectively, where d_w is the dimension of the word embedding and d_s is the dimension of each sentence representation vector. Specifically, we first use Convolutional Neural Networks (CNN) (LeCun et al., 1998) with different kernel sizes to capture the local n-gram feature for each sentence l_j and then use the bidirectional Long Short-Term Memory (BiLSTM) (Hochreiter and Schmidhuber, 1997) layer to get the sentence-level feature g_j. The concatenation of the CNN local feature and the BiLSTM global feature is used as the sentence node feature $X_{s_j} = [l_j; g_j]$.

To further include information about the importance of relationships between word and sentence nodes, we infuse TF-IDF values in the edge weights. The term frequency (TF) is the number of times w_i occurs in s_j and the inverse document frequency (IDF) is made as the inverse function of the out-degree of w_i.

3.3 Heterogeneous Graph Layer

Given a constructed graph G with node features $\mathbf{X}_w \cup \mathbf{X}_s$ and edge features $\mathbf{E}$, we use graph attention networks (Velickovic et al., 2017) to update the representations of our semantic nodes.

We refer to $\boldsymbol{h}_i \in \mathbb{R}^{d_h}$, $i \in \{1, \cdots, (m+n)\}$ as the hidden states of input nodes and the graph attention (GAT) layer is designed as follows:

$$z_{ij} = \text{LeakyReLU}\left(\mathbf{W}_a[\mathbf{W}_q\boldsymbol{h}_i; \mathbf{W}_k\boldsymbol{h}_j]\right), \quad (1)$$

$$\alpha_{ij} = \frac{\exp(z_{ij})}{\sum_{l\in\mathcal{N}_i}\exp(z_{il})}, \quad (2)$$

$$\boldsymbol{u}_i = \sigma\Big(\sum_{j\in\mathcal{N}_i}\alpha_{ij}\mathbf{W}_v\boldsymbol{h}_j\Big), \quad (3)$$

where $\mathbf{W}_a$, $\mathbf{W}_q$, $\mathbf{W}_k$, $\mathbf{W}_v$ are trainable weights and α_{ij} is the attention weight between $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$. The multi-head attention can be denoted as:

$$\boldsymbol{u}_i = \Big\|_{k=1}^{K}\sigma\left(\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k\mathbf{W}^k\boldsymbol{h}_i\right). \quad (4)$$

Besides, we also add a residual connection to avoid gradient vanishing after several iterations. Therefore, the final output can be represented as:

$$\boldsymbol{h}'_i = \boldsymbol{u}_i + \boldsymbol{h}_i. \quad (5)$$

We further modify the GAT layer to infuse the scalar edge weights e_{ij}, which are mapped to the multi-dimensional embedding space $\boldsymbol{e}_{ij} \in \mathbb{R}^{mn\times d_e}$. Thus, Equal 1 is modified as follows:

$$z_{ij} = \text{LeakyReLU}\left(\mathbf{W}_a[\mathbf{W}_q\boldsymbol{h}_i; \mathbf{W}_k\boldsymbol{h}_j; \boldsymbol{e}_{ij}]\right). \quad (6)$$

After each graph attention layer, we introduce a position-wise feed-forward (FFN) layer consisting of two linear transformations just as Transformer (Vaswani et al., 2017).

Iterative updating To pass messages between word and sentence nodes, we define the information propagation as Figure 2. Specifically, after the initialization, we update sentence nodes with their neighbor word nodes via the above GAT and FFN layer:

$$\mathbf{U}^1_{s\leftarrow w} = \text{GAT}(\mathbf{H}^0_s, \mathbf{H}^0_w, \mathbf{H}^0_w), \quad (7)$$

$$\mathbf{H}^1_s = \text{FFN}\left(\mathbf{U}^1_{s\leftarrow w} + \mathbf{H}^0_s\right), \quad (8)$$

where $\mathbf{H}^1_w = \mathbf{H}^0_w = \mathbf{X}_w$, $\mathbf{H}^0_s = \mathbf{X}_s$ and $\mathbf{U}^1_{s\leftarrow w} \in \mathbb{R}^{m\times d_h}$. $\text{GAT}(\mathbf{H}^0_s, \mathbf{H}^0_w, \mathbf{H}^0_w)$ denotes that $\mathbf{H}^0_s$ is used as the attention query and $\mathbf{H}^0_w$ is used as the key and value.

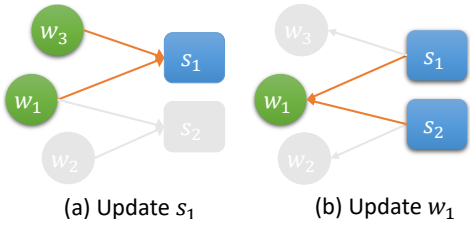

Figure 2: The detailed update process of word and sentence nodes in *Heterogeneous Graph Layer*. Green and blue nodes are word and sentence nodes involved in this turn. Orange edges indicate the current information flow direction. First, for sentence s_1, word w_1 and w_3 are used to aggregate word-level information in (a). Next, w_1 is updated by the new representation of s_1 and s_2 in (b), which are the sentences it occurs. See Section 3.3 for details on the notation.

After that, we obtain new representations for word nodes using the updated sentence nods and further update sentence nodes iteratively. Each iteration contains a *sentence-to-word* and a *word-to-sentence* update process. For the t-th iteration, the process can be represented as:

$$\mathbf{U}^{t+1}_{w\leftarrow s} = \text{GAT}(\mathbf{H}^t_w, \mathbf{H}^t_s, \mathbf{H}^t_s), \quad (9)$$

$$\mathbf{H}^{t+1}_w = \text{FFN}\left(\mathbf{U}^{t+1}_{w\leftarrow s} + \mathbf{H}^t_w\right), \quad (10)$$

$$\mathbf{U}^{t+1}_{s\leftarrow w} = \text{GAT}(\mathbf{H}^t_s, \mathbf{H}^{t+1}_w, \mathbf{H}^{t+1}_w), \quad (11)$$

$$\mathbf{H}^{t+1}_s = \text{FFN}\left(\mathbf{U}^{t+1}_{s\leftarrow w} + \mathbf{H}^t_s\right). \quad (12)$$

As Figure 2 shows, word nodes can aggregate the document-level information from sentences. For example, the high degree of a word node indicates the word occurs in many sentences and is likely to be the keyword of the document. Regarding sentence nodes, the one with more important words tends to be selected as the summary.

3.4 Sentence Selector

Finally, we need to extract sentence nodes included in the summary from the heterogeneous graph. Therefore, we do node classification for sentences and cross-entropy loss is used as the training objective for the whole system.

Trigram blocking Following Paulus et al. (2017) and Liu and Lapata (2019b), we use Trigram Blocking for decoding, which is simple but powerful version of Maximal Marginal Relevance (Carbonell and Goldstein, 1998). Specifically, we rank sentences by their scores and discard those which have trigram overlappings with their predecessors.

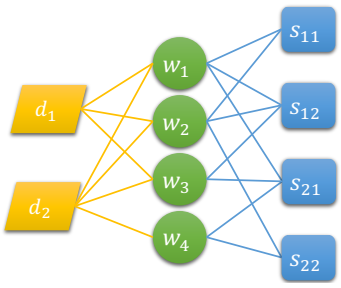

Figure 3: Graph structure of HETERDOCSUMGRAPH for multi-document summarization (corresponding to the *Graph Layer* part of Figure 1). Green, blue and orange boxes represent word, sentence and document nodes respectively. d_1 consists of s_{11} and s_{12} while d_2 contains s_{21} and s_{22}. As a relay node, the relation of *document-document*, *sentence-sentence*, and *sentence-document* can be built through the common word nodes. For example, sentence s_{11}, s_{12} and s_{21} share the same word w_1, which connects them across documents.

3.5 Multi-document Summarization

For multi-document summarization, the document-level relation is crucial for better understanding the core topic and most important content of this cluster. However, most existing neural models ignore this hierarchical structure and concatenate documents to a single flat sequence(Liu et al., 2018; Fabbri et al., 2019). Others try to model this relation by attention-based full-connected graph or take advantage of similarity or discourse relations(Liu and Lapata, 2019a).

Our framework can establish the document-level relationship in the same way as the sentence-level by just adding supernodes for documents(as Figure 3), which means it can be easily adapted from single-document to multi-document summarization. The heterogeneous graph is then extended to three types of nodes: $V = V_w \cup V_s \cup V_d$ and $V_d = \{d_1, \cdots, d_l\}$ and l is the number of source documents. We name it as HETERDOCSUMGRAPH.

As we can see in Figure 3, word nodes become the bridges between sentences and documents. Sentences containing the same words connect with each other regardless of their distance across documents, while documents establish relationships based on their similar contents.

Document nodes can be viewed as a special type of sentence nodes: a document node connects with contained word nodes and the TF-IDF value is used as the edge weight. Besides, document nodes also share the same update process as sentence nodes. The differences lie in the initialization, where the document node takes the mean-pooling of its sentence node features as its initial state. During the sentence selection, the sentence nodes are concatenated with the corresponding document representations to obtain the final scores for multi-document summarization.

4 Experiment

We evaluate our models both on single- and multi-document summarization tasks. Below, we start our experiment with the description of the datasets.

4.1 Datasets

CNN/DailyMail The CNN/DailyMail question answering dataset (Hermann et al., 2015; Nallapati et al., 2016) is the most widely used benchmark dataset for single-document summarization. The standard dataset split contains 287,227/13,368/11,490 examples for training, validation, and test. For the data prepossessing, we follow Liu and Lapata (2019b), which use the non-anonymized version as See et al. (2017), to get ground-truth labels.

NYT50 NYT50 is also a single-document summarization dataset, which was collected from New York Times Annotated Corpus (Sandhaus, 2008) and preprocessed by Durrett et al. (2016). It contains 110,540 articles with summaries and is split into 100,834 and 9706 for training and test. Following Durrett et al. (2016), we use the last 4,000 examples from the training set as validation and filter test examples to 3,452.

Multi-News The Multi-News dataset is a large-scale multi-document summarization introduced by Fabbri et al. (2019). It contains 56,216 articles-summary pairs and each example consists of 2-10 source documents and a human-written summary. Following their experimental settings, we split the dataset into 44,972/5,622/5,622 for training, validation and test examples and truncate input articles to 500 tokens.

4.2 Settings and Hyper-parameters

For both single-document and multi-document summarization, we limit the vocabulary to 50,000 and initialize tokens with 300-dimensional GloVe embeddings (Pennington et al., 2014). We filter stop words and punctuations when creating word

nodes and truncate the input document to a maximum length of 50 sentences. To get rid of the noisy common words, we further remove 10% of the vocabulary with low TF-IDF values over the whole dataset. We initialize sentence nodes with $d_s = 128$ and edge features $\boldsymbol{e}_{ij}$ in GAT$_e$ with $d_e = 50$. Each GAT layer is 8 heads and the hidden size is $d_h = 64$, while the inner hidden size of FFN layers is 512.

During training, we use a batch size of 32 and apply Adam optimizer (Kingma and Ba, 2014) with a learning rate 5e-4. An early stop is performed when valid loss does not descent for three continuous epochs. We select the number of iterations $t = 1$ based on the performance on the validation set.[3] For decoding, we select top-3 sentences for CNN/DailyMail and NYT50 datasets and top-9 for Multi-New according to the average length of their human-written summaries.

4.3 Models for Comparison

Ext-BiLSTM Extractive summarizer with BiLSTM encoder learns the cross-sentence relation by regarding a document as a sequence of sentences. For simplification, we directly take out the initialization of sentence nodes for classification, which includes a CNN encoder for the word level and 2-layer BiLSTM for sentence level. This model can also be viewed as an ablation study of our HETERSUMGRAPH on the updating of sentence nodes.

Ext-Transformer Extractive summarizers with Transformer encoder learn the pairwise interaction (Vaswani et al., 2017) between sentences in a purely data-driven way with a fully connected priori. Following (Liu and Lapata, 2019b), we implement a Transformer-based extractor as a baseline, which contains the same encoder for words followed by 12 Transformer encoder layers for sentences. Ext-Transformer can be regarded as the sentence-level fully connected graph.

HETERSUMGRAPH Our heterogeneous summarization graph model relations between sentences based on their common words, which can be denoted as *sentence-word-sentence* relationships. HETERSUMGRAPH directly selects sentences for the summary by node classification, while HETERSUMGRAPH with trigram blocking further utilizes the n-gram blocking to reduce redundancy.

[3]The detailed experimental results are attached in the Appendix Section.

Model	R-1	R-2	R-L
LEAD-3 (See et al., 2017)	40.34	17.70	36.57
ORACLE (Liu and Lapata, 2019b)	52.59	31.24	48.87
REFRESH (Narayan et al., 2018)	40.00	18.20	36.60
LATENT (Zhang et al., 2018)	41.05	18.77	37.54
BanditSum (Dong et al., 2018)	41.50	18.70	37.60
NeuSUM (Zhou et al., 2018)	41.59	19.01	37.98
JECS (Xu and Durrett, 2019)	41.70	18.50	37.90
LSTM+PN (Zhong et al., 2019a)	41.85	18.93	38.13
HER w/o Policy (Luo et al., 2019)	41.70	18.30	37.10
HER w Policy (Luo et al., 2019)	42.30	18.90	37.60
Ext-BiLSTM	41.59	19.03	38.04
Ext-Transformer	41.33	18.83	37.65
HSG	42.31	19.51	38.74
HSG + Tri-Blocking	**42.95**	**19.76**	**39.23**

Table 1: Performance (Rouge) of our proposed models against recently released summarization systems on CNN/DailyMail.

5 Results and Analysis

5.1 Single-document Summarization

We evaluate our single-document model on CNN/DailyMail and NYT50 and report the unigram, bigram and longest common subsequence overlap with reference summaries by R-1, R-2 and R-L. Due to the limited computational resource, we don't apply pre-trained contextualized encoder (i.e. BERT (Devlin et al., 2018)) to our models, which we will regard as our future work. Therefore, here, we only compare with models without BERT for the sake of fairness.

Results on CNN/DailyMail Table 1 shows the results on CNN/DailyMail. The first part is the LEAD-3 baseline and ORACLE upper bound, while the second part includes other summarization models.

We present our models (described in Section 4.3) in the third part. Compared with Ext-BiLSTM, our heterogeneous graphs achieve more than 0.6/0.51/0.7 improvements on R-1, R-2 and R-L, which indicates the cross-sentence relationships learned by our *sentence-word-sentence* structure is more powerful than the sequential structure. Besides, Our models also outperform Ext-Transformer based on fully connected relationships. This demonstrates that our graph structures effectively prune unnecessary connections between sentences and thus improve the performance of sentence node classification.

Compared with the second block of Figure 1, we observe that HETERSUMGRAPH outperforms all previous non-BERT-based summarization systems

and trigram blocking leads to a great improvement on all ROUGE metrics. Among them, HER (Luo et al., 2019) is a comparable competitor to our HETERSUMGRAPH, which formulated the extractive summarization task as a contextual-bandit problem and solved it with reinforcement learning. Since the reinforcement learning and our trigram blocking plays a similar role in reorganizing sentences into a summary (Zhong et al., 2019a), we additionally compare HER without policy gradient with HETERSUMGRAPH. Our HETERSUMGRAPH achieve 0.61 improvements on R-1 over HER without policy for sentence scoring, and HETERSUMGRAPH with trigram blocking outperforms by 0.65 over HER for the reorganized summaries.

Model	R-1	R-2	R-L
First sentence (Durrett et al., 2016)	28.60	17.30	-
First k words (Durrett et al., 2016)	35.70	21.60	-
LEAD-3	38.99	18.74	35.35
ORACLE	60.54	40.75	57.22
COMPRESS (Durrett et al., 2016)	42.20	24.90	-
SUMO (Liu et al., 2019)	42.30	22.70	38.60
PG* (See et al., 2017)	43.71	26.40	-
DRM (Paulus et al., 2017)	42.94	26.02	-
Ext-BiLSTM	46.32	25.84	42.16
Ext-Transformer	45.07	24.72	40.85
HSG	**46.89**	**26.26**	**42.58**
HSG + Tri-Blocking	46.57	25.94	42.25

Table 2: Limited-length ROUGE Recall on NYT50 test set. The results of models with * are copied from Liu and Lapata (2019b) and '-' means that the original paper did not report the result.

Results on NYT50 Results on NYT50 are summarized in Table 2. Note that we use limited-length ROUGE recall as Durrett et al. (2016), where the selected sentences are truncated to the length of the human-written summaries and the recall scores are used instead of F1. The first two lines are baselines given by Durrett et al. (2016) and the next two lines are our baselines for extractive summarization. The second and third part report the performance of other non-BERT-based works and our models respectively.

Again, we observe that our cross-sentence relationship modeling performs better than BiLSTM and Transformer. Our models also have strong advantages over other non-BERT-based approaches on NYT50. Meanwhile, we find trigram block doesn't work as well as shown on CNN/DailyMail, and we attribute the reason to the special formation of summaries of CNN/DailyMail dataset. [4]

Ablation on CNN/DailyMail In order to better understand the contribution of different modules to the performance, we conduct ablation study using our proposed HETERSUMGRAPH model on CNN/DailyMail dataset. First, we remove the filtering mechanism for low TF-IDF words and the edge weights respectively. We also remove residual connections between GAT layers. As a compensation, we concatenate the initial sentence feature after updating messages from nearby word nodes in Equal 8:

$$\mathbf{H}_s^1 = \mathrm{FFN}\left([\mathbf{U}_{s\leftarrow w}^1; \mathbf{H}_s^0]\right). \tag{13}$$

Furthermore, we make iteration number $t = 0$, which deletes the word updating and use the sentence representation $\mathbf{H}_s^1$ for classification. Finally, we remove the BiLSTM layer in the initialization of sentence nodes.

As Table 3 shows, the removal of low TF-IDF words leads to increases on R-1 and R-L but drops on R-2. We suspect that filtering noisy words enable the model to better focus on useful word nodes, at the cost of losing some bigram information. The residual connection plays an important role in the combination of the original representation and the updating message from another type of nodes, which cannot be replaced by the concatenation. Besides, the introduction of edge features, word update and BiLSTM initialization for sentences also show their effectiveness.

5.2 Multi-document Summarization

We first take the concatenation of the First-k sentences from each source document as the baseline and use the codes and model outputs[5] released by Fabbri et al. (2019) for other models.

To explore the adaptability of our model to multi-document summarization, we concatenate multi-source documents to a single mega-document and apply HETERSUMGRAPH as the baseline. For comparison, we extend HETERSUMGRAPH to multi-document settings HETERDOCSUMGRAPH

[4] Nallapati et al. (2016) concatenate summary bullets, which are written for different parts of the article and have few overlaps with each other, as a multi-sentence summary. However, when human write summaries for the whole article (such as NYT50 and Multi-News), they will use key phrases repeatedly. This means roughly removing sentences by n-gram overlaps will lead to loss of important information.

[5] https://github.com/Alex-Fabbri/ Multi-News

Model	R-1	R-2	R-L
HSG	42.31	19.51	38.74
- filter words	42.24	19.56	38.68
- edge feature	42.14	19.41	38.60
- residual connection	41.59	19.08	38.05
- sentence update	41.59	19.03	38.04
- word update	41.70	19.16	38.15
- BiLSTM	41.70	19.09	38.13

Table 3: Ablation studies on CNN/DailyMail test set. We remove various modules and explore their influence on our model. '-' means we remove the module from the original HETERSUMGRAPH. Note that HETERSUMGRAPH without the updating of sentence nodes is actually the Ext-BiLSTM model described in Section 4.3.

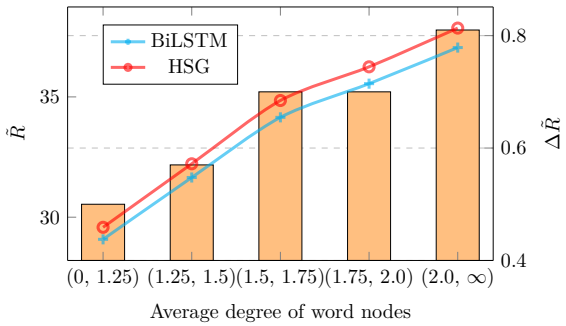

Figure 4: Relationships between the average degree of word nodes of the document (x-axis) and $\tilde{R}$, which is the mean of R-1, R-2 and R-L (lines for left y-axis), and between $\Delta\tilde{R}$, which is the delta $\tilde{R}$ of HETERSUMGRAPH and Ext-BiLSTM (histograms for right y-axis).

as described in Section 3.5. Our results are presented in Table 4.

Specifically, we observe that both of our HETERSUMGRAPH and HETERDOCSUMGRAPH outperform previous methods while HETERDOCSUMGRAPH achieves better performance improvements. This demonstrates the introduction of document nodes can better model the document-document relationships and is beneficial for multi-document summarization. As mentioned above, trigram blocking does not work for the Multi-News dataset, since summaries are written as a whole instead of the concatenations of summary bullets for each source document.

Model	R-1	R-2	R-L
First-1	25.44	7.06	22.12
First-2	35.70	10.28	31.71
First-3	40.21	12.13	37.13
ORACLE	52.32	22.23	47.93
LexRank* (Erkan and Radev, 2004)	41.77	13.81	37.87
TextRank* (Mihalcea and Tarau, 2004)	41.95	13.86	38.07
MMR* (Carbonell and Goldstein, 1998)	44.72	14.92	40.77
PG† (Lebanoff et al., 2018)	44.55	15.54	40.75
BottomUp† (Gehrmann et al., 2018)	45.27	15.32	41.38
Hi-MAP† (Fabbri et al., 2019)	45.21	16.29	41.39
HSG	45.66	16.22	41.80
HSG + Tri-Blocking	44.92	15.59	40.89
HDSG	**46.05**	**16.35**	**42.08**
HDSG + Tri-Blocking	45.55	15.78	41.29

Table 4: Results on the test set of Multi-News. We reproduce models with '*' via the released code and directly use the outputs of † provided by Fabbri et al. (2019) for evaluation.

5.3 Qualitative Analysis

We further design several experiments to probe into how our HETERSUMGRAPH and HETERDOCSUMGRAPH help the single- and multi-document summarization.

Degree of word nodes In HETERSUMGRAPH, the degree of a word node indicates its occurrence across sentences and thus can measure the redundancy of the document to some extent. Meanwhile, words with a high degree can aggregate information from multiple sentences, which means that they can benefit more from the iteration process. Therefore, it is important to explore the influence of the node degree of words on the summarization performance.

We first calculate the average degree of word nodes for each example based on the constructed graph. Then the test set of CNN/DailyMail is divided into 5 intervals based on it (x-axis in Figure 4). We evaluate the performance of HETERSUMGRAPH and Ext-BiLSTM in various parts and the mean score of R-1, R-2, R-L is drawn as lines (left y-axis $\tilde{R}$). The ROUGE increases with the increasing of the average degree of word nodes in the document, which means that articles with a high redundancy are easier for neural models to summarize.

To make $\Delta\tilde{R}$ between models more obvious, we draw it with histograms (right y-axis). From Figure 4, we can observe that HETERSUMGRAPH performs much better for documents with a higher average word node degree. This proves that the benefit brought by word nodes lies in the aggregation of information from sentences and the propagation of their global representations.

Number of source documents We also investigate how the number of source documents influences the performance of our model. To this end,

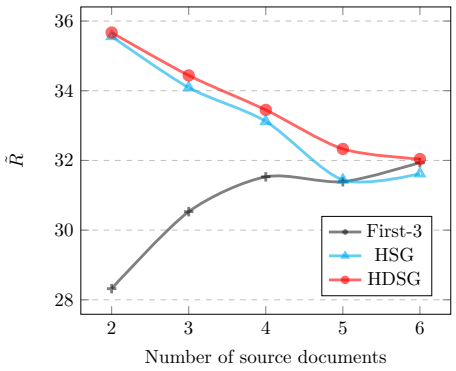

Figure 5: Relationship between number of source documents (x-axis) and $\bar{R}$ (y-axis).

we divide the test set of Multi-News into different parts by the number of source documents and discard parts with less than 100 examples. Then, we take First-3 as the baseline, which concatenates the top-3 sentences of each source document as the summary.

In Figure 5, we can observe that the lead baseline raises while both of our model performance degrade and finally they converge to the baseline. This is because it is more challenging for models to extract limited-number sentences that can cover the main idea of all source documents with the increasing number of documents. However, the First-3 baseline is forced to take sentences from each document which can ensure the coverage. Besides, the increase of document number enlarges the performance gap between HETERSUMGRAPH and HETERDOCSUMGRAPH. This indicates the benefit of document nodes will become more significant for more complex document-document relationships.

6 Conclusion

In this paper, we propose a heterogeneous graph-based neural network for extractive summarization. The introduction of more fine-grained semantic units in the summarization graph helps our model to build more complex relationships between sentences . It is also convenient to adapt our single-document graph to multi-document with document nodes. Furthermore, our models have achieved the best results on CNN/DailyMail compared with non-BERT-based models, and we will take the pre-trained language models into account for better encoding representations of nodes in the future.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (No. U1936214 and 61672162), Shanghai Municipal Science and Technology Major Project (No. 2018SHZDZX01) and ZJLab.

References

Kristjan Arumae and Fei Liu. 2018. Reinforced extractive summarization with question-focused rewards. In *Proceedings of ACL 2018, Student Research Workshop*, pages 105–111.

Jaime G Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, volume 98, pages 335–336.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 484–494.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Yue Dong, Yikang Shen, Eric Crawford, Herke van Hoof, and Jackie Chi Kit Cheung. 2018. Banditsum: Extractive summarization as a contextual bandit. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3739–3748.

Greg Durrett, Taylor Berg-Kirkpatrick, and Dan Klein. 2016. Learning-based single-document summarization with compression and anaphoricity constraints. *arXiv preprint arXiv:1603.08887*.

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.

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.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *ICML*, pages 1263–1272.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1684–1692.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

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

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.

Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*.

Hu Linmei, Tianchi Yang, Chuan Shi, Houye Ji, and Xiaoli Li. 2019. Heterogeneous graph attention networks for semi-supervised short text classification. 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 4823–4832.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *Proceedings of the 6th 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.

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 3721–3731, Hong Kong, China. Association for Computational Linguistics.

Yang Liu, Ivan Titov, and Mirella Lapata. 2019. Single document summarization as tree induction. 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 1745–1755.

Ling Luo, Xiang Ao, Yan Song, Feiyang Pan, Min Yang, and Qing He. 2019. Reading like her: Human reading inspired extractive 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 3024–3034.

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.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Ça glar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *CoNLL 2016*, page 280.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. 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 1747–1759.

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

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

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)*, volume 1, pages 1073–1083.

Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and S Yu Philip. 2016. A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering*, 29(1):17–37.

Ming Tu, Guangtao Wang, Jing Huang, Yun Tang, Xiaodong He, and Bowen Zhou. 2019. Multi-hop reading comprehension across multiple documents by reasoning over heterogeneous graphs. *arXiv preprint arXiv:1905.07374*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

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

Danqing Wang, Pengfei Liu, Ming Zhong, Jie Fu, Xipeng Qiu, and Xuanjing Huang. 2019a. Exploring domain shift in extractive text summarization. *arXiv preprint arXiv:1908.11664*.

Hsiu-Yi Wang, Jia-Wei Chang, and Jen-Wei Huang. 2019b. User intention-based document summarization on heterogeneous sentence networks. In *International Conference on Database Systems for Advanced Applications*, pages 572–587. Springer.

Yang Wei. 2012. Document summarization method based on heterogeneous graph. In *2012 9th International Conference on Fuzzy Systems and Knowledge Discovery*, pages 1285–1289. IEEE.

Jiacheng Xu and Greg Durrett. 2019. Neural extractive text summarization with syntactic compression. *arXiv preprint arXiv:1902.00863*.

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

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. 2017. Graph-based neural multi-document summarization. *arXiv preprint arXiv:1706.06681*.

Xingxing Zhang, Mirella Lapata, Furu Wei, and Ming Zhou. 2018. Neural latent extractive document summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 779–784.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuan-Jing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Conference of the Association for Computational Linguistics*.

Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuan-Jing Huang. 2019a. Searching for effective neural extractive summarization: What works and what's next. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1049–1058.

Ming Zhong, Danqing Wang, Pengfei Liu, Xipeng Qiu, and Xuan-Jing Huang. 2019b. A closer look at data bias in neural extractive summarization models. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 80–89.

Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, and Tiejun Zhao. 2018. Neural document summarization by jointly learning to score and select sentences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 654–663.

A Appendices

In order to select the best iteration number for HETERSUMGRAPH, we compare performances of different t on the validation set of CNN/DM. All models are trained on a single GeForce RTX 2080 Ti GPU for about 5 epochs. As Table 5 shows, our HETERSUMGRAPH has comparable results for $t = 1$ and $t = 3$. However, when the iteration number goes from 1 to 3, the time for one epoch nearly doubles. Therefore, we take $t = 1$ as a result of the balance of time cost and model performance.

Number	R-1	R-2	R-L	Time
$t = 0$	43.63	19.58	37.39	3.16h
$t = 1$	44.26	19.97	38.03	5.04h
$t = 2$	44.13	19.85	37.87	7.20h
$t = 3$	44.28	19.96	37.98	8.93h

Table 5: Different turns of iterative updating of sentence nodes. The experiments are performed on the validation set of CNN/DM. Time is the average time of one epoch.