SAPGraph: Structure-aware Extractive Summarization for Scientific Papers with Heterogeneous Graph

Siya Qi[*1] **Lei Li**[†*1] **Yiyang Li**[1] **Jin Jiang**[1] **Dingxin Hu**[1] **Yuze Li**[1]
Yingqi Zhu[1] **Yanquan Zhou**[1] **Marina Litvak** [2] **Natalia Vanetik**[2]
[1]Beijing University of Posts and Telecommunications
[2]Shamoon College of Engineering
{qsy,leili,kenlee,jiangjin}@bupt.edu.cn
{hudingxin,lyzbupt,zhuyq,zhouyanquan}@bupt.edu.cn
{marinal,natalyav}@sce.ac.il

Abstract

Scientific paper summarization is always challenging in Natural Language Processing (NLP) since it is hard to collect summaries from such long and complicated text. We observe that previous works tend to extract summaries from the head of the paper, resulting in information incompleteness. In this work, we present SAPGraph[1] to utilize paper structure for solving this problem. SAPGraph is a scientific paper extractive summarization framework based on a structure-aware heterogeneous graph, which models the document into a graph with three kinds of nodes and edges based on structure information of facets and knowledge. Additionally, we provide a large-scale dataset of COVID-19-related papers, CORD-SUM. Experiments on CORD-SUM and ArXiv datasets show that SAPGraph generates more comprehensive and valuable summaries compared to previous works.

1 Introduction

In recent years, scientific papers represented by COVID-19-related papers have shown an expanding growth in a short period, which produces information overload and makes it difficult for researchers to follow. Automatic summarization can help researchers quickly focus on valuable information in the article and be updated about the latest research progress. The goal of automatic summarization is to condense a long text into a concise summary while retaining essential information. It evolves mainly in two directions: abstractive and extractive methods. Abstractive summarization generates summaries which are rewritten and refined (Lewis et al., 2020; Zhang et al., 2020), while the extractive one selects text segments as summaries (Liu and Lapata, 2019; Nallapati et al., 2017; Zhong et al., 2020; S et al., 2021), which is easier to be applied practically and keep grammar correct. In this work, we study the extractive summarization of scientific papers, which are much longer than news articles (see Table 1). Scientific papers also contain different facets of sections, which are usually composed of **Introduction**, **Method**, **Result**, and **Conclusion** (Hartley, 2014), assisting readers in constructing a coherent chain of idea.

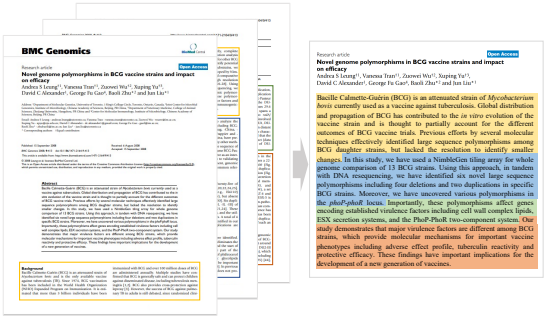


Figure 1: An example in our CORD-SUM dataset. Texts highlighted with different colors denote different facets of the summary.

For scientific paper summarization, it is difficult to generate summaries from professional texts like COVID-19-related papers, due to their long texts with complicated structures. To deal with the long text, classical deep learning methods simply truncate documents and may therefore discard useful information. Other methods propose a better data structure, such as graph-based models (Wang et al., 2020a; Dong et al., 2021; Zheng and Lapata, 2019) or sliding window in sequence models (Beltagy et al., 2020; Cui and Hu, 2021). Some scientific paper summarization studies have noticed the importance of writing structure in papers, to better deal with long text (Meng et al., 2021). These works consider the paper structure and try to manually pick sections as input (Cachola et al., 2020), or they consider hierarchical features of a document (Cao and Wang, 2022; Cohan et al., 2018).

[*]The first two authors contributed equally.
[†]Corresponding author.
[1]Available at: https://github.com/cece00/SAPGraph

Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 575–586
November 20–23, 2022.

Among the extractive methods, we notice that these works are still insufficient at dealing with papers and are prone to obtain summaries with *head distribution problems*, which means that systems tend to extract summaries from the beginning of the document (see Figure 5). The reasons might be that sequence-based extractive summarization models are weak at establishing potential associations of distant sentences, despite the sliding window mechanism. And furthermore, the structure of long papers is not well-utilized because long documents always possess several facets with certain logical relations, as in Figure 1. Hence, the extracted summaries are incomplete and cannot cover all the critical information that researchers need.

To improve this problem, we propose a **S**tructure-**A**ware **P**aper Heterogeneous **Graph** Network (SAPGraph) for scientific paper summarization. Inspired by Meng et al. (2021) and Hartley et al. (1996), facet structure is deeply considered in SAPGraph. And the domain knowledge is also crucial for papers, which can be seen as a latent structure. Based on these structures, SAPGraph models an entire paper as a heterogeneous graph with three node types: section, sentence, and entity, and is trained with the Graph Neural Network (GNN) (Kipf and Welling, 2016; Veličković et al., 2018). Such a design can effectively aggregate information from different facets and improve the diversity and coverage of summaries. Also, we provide CORD-SUM, a summarization dataset based on COVID-19 Open Research Dataset (CORD-19) (Wang et al., 2020b)[2]. We compare SAPGraph with strong extractive summarization models, and our experiments show that SAPGraph outperforms previous works in terms of ROUGE (Lin and Hovy, 2003) and BERTScore (Zhang et al., 2019) on CORD-SUM and ArXiv (Cohan et al., 2018). In our metrics, ROUGE-N and ROUGE-L can measure the similarity between system summaries and reference summaries by the n-gram co-occurrences and the longest common subsequence, and BERTScore computes this similarity based on cosine similarities between their tokens' embeddings. Ablation studies show our evaluation on different graph structures, suggesting that SAPGraph can surpass other types of graph construction.

Our contributions are highlighted as follows: Firstly, we provide CORD-SUM, a summarization dataset compiled of scientific papers about COVID-19, and their summaries. The dataset and construction code are publicly available for researchers to process the updated CORD-19 dataset. Secondly, we propose SAPGraph, a multi-layer heterogeneous graph for structure-aware paper summarization. SAPGraph effectively models an entire paper with much fewer structural nodes and edges than state-of-the-art graphs. The final point is that results on the dataset of CORD-SUM and ArXiv prove the effectiveness of our work. And our experiments show that SAPGraph successfully utilizes the explicit structure of facets and the implicit structure of knowledge to alleviate the head distribution problem in scientific paper summarization.

[2]Weekly updated on Kaggle: https://www.kaggle.com/datasets/allen-institute-for-ai/CORD-19-research-challenge

2 Related work

The study of extractive summarization of scientific papers has always been a hotspot. Just as regular extractive summarization, systems for scientific papers aim to pick informative texts from the source document to form a summary, except that these documents are longer, more professional, and have a clear hierarchical structure.

With the development of sequence neural networks, more RNN and Transformer-based models are used for scientific paper summarization. Sequence models like hierarchical RNN are used to build attention between different layers of the paper on ArXiv and PubMed (Cohan et al., 2018). Global and local contexts are also considered when extracting sentences (Xiao and Carenini, 2019). DANCER (Gidiotis and Tsoumakas, 2020) selects sections and makes multiple source-target pairs to generate summaries respectively. Meng et al. (2021) generate a summary from four aspects of Emerald dataset, including Purpose, Method, Findings, and Value. Subramanian et al. (2020) use an extract-then-abstract model and pick out the Introduction section as one input. For sequence-based methods, papers are too long to process directly. Unlike vanilla sequence models accompanied by truncation of long text, SCITLDR (Cachola et al., 2020) performs extreme summarization from concatenated Introduction and Conclusion, which is more reasonable than treating every section equally. But other than shortening the text, sliding window (Beltagy et al., 2020; Cui and Hu, 2021; Grail et al., 2021) is commonly used. For instance, Longformer (Beltagy et al., 2020) relieves the computational pressure caused by the attention mechanism

with sliding window attention, and can be used on long text summarization as BERT does (Liu and Lapata, 2019). Other pretrained language models such as SCIBERT (Beltagy et al., 2019) and BIOBERT (Lee et al., 2020), which are pretrained on scientific literature or medical papers, are more adaptable to scientific document processing tasks.

Although some of the above works value the function of facet structure, the majority of them rely on manual selection, which lacks universality and may also result in the loss of supporting information. In contrast, graph-based models are more flexible and can build connections between long-span texts.

Early works like LexRank (Erkan and Radev, 2004) and TextRank (Mihalcea and Tarau, 2004) predict sentence centrality of a document graph. Recently, more well-designed graph-based methods consider the structure information, such as PACSUM (Zheng and Lapata, 2019), HIPORANK (Dong et al., 2021), FAR (Liang et al., 2021), etc. To rank sentences, they fuse together such information as hierarchical structure, sentence position, and sentence similarity. GNN (Kipf and Welling, 2016; Veličković et al., 2018) can learn nodes representation with neural networks. Heterogeneous graph methods (Huang and Kurohashi, 2021; Wang et al., 2020a; Yasunaga et al., 2017) can consider more diverse information with multi-type nodes and edges. In graph-based works, HETERSUMGRAPH (HSG) (Wang et al., 2020a) is comparable to our SAPGraph, but SAPGraph takes into account the structure of facets and knowledge in the paper, making it a better graph prior to paper summarizing.

3 Approach

Here we describe three main stages of SAPGraph: the facet alignment between summaries and source documents, the graph construction, and the learning method applied to the constructed graph. Figure 2 shows the overall framework of SAPGraph.

3.1 Facet Alignment

To better guide our model, we first investigate the distribution of gold summary sentences on paper facets. And we use the author-written abstracts as gold summaries in our experiments. For the most part, however, summaries have no clear segmentation facets. But papers do have section facets, usually named, Introduction, Method, Result and Conclusion. So we divide papers into the above four facet categories by keyword matching (Meng et al., 2021) on section names (see Appendix A). The mismatched section names are classified into Others.

Based on the classification results, we count the number of article sentences in category i having the highest ROUGE scores with summary sentences as C_i. The proportion of each category in a summary is measured by $C_i / \sum_i (C_i)$. Here, we sample 100 articles illustrated as a heat map (Figure 3). It is noticeable that Introduction and Conclusion account for a high percentage of a summary (Cachola et al., 2020), but the other three categories also cannot be discounted. We calculate the average percentage of each category in our data as follows: $FacetWeight = [0.35, 0.1, 0.15, 0.35, 0.05]$, respectively. We also infuse this structure information into our graph.

3.2 Graph Construction

3.2.1 Node Embedding

Sentence embedding, which represents the local information inside one sentence, is crucial to the initialization of the graph model. We implement a local encoder to embed entities and sentences, the same graph initializer as HSG (Wang et al., 2020a) to verify the function of our graph, which consists of a CNN (LeCun et al., 1998) and a BiLSTM (Hochreiter and Schmidhuber, 1997) encoder. The output of the local encoder is the initial representation of the sentence node. As for entity nodes, we set entity embedding to be the mean pooling of its words. The representation of a section node is the mean pooling of all sentences belonging to it, for the purpose of gathering comprehensive information.

3.2.2 Heterogeneous Graph

Given a document, $D = \{sec_1, sec_2, \cdots, sec_n\}$, with n sections, we model each section as a relatively independent subgraph and connect them according to the original structure of the paper. In every subgraph, sentences are connected to each other with edges that consider similarity, as in TextRank (Mihalcea and Tarau, 2004). Local information inside a sentence is emphasized by entities, while global information across sentences and sections is leveraged by inter-sentence and inter-section connections.

For each section, we implement a subgraph as shown in Figure 2 (top). The subgraph contains

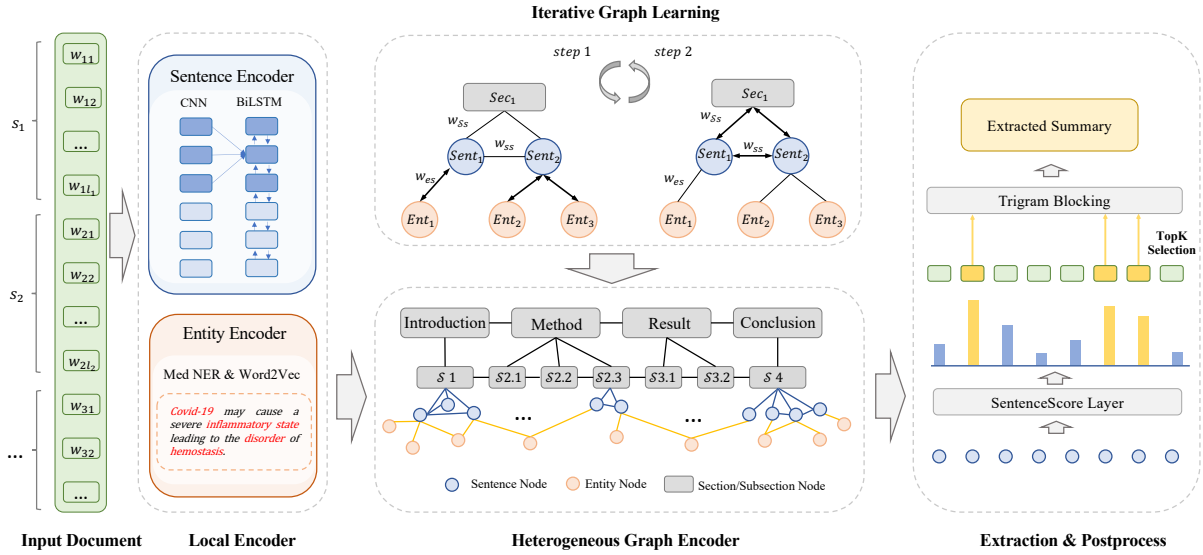


Figure 2: The model contains three main modules: 1) **Local Encoder**: is composed of an Entity Encoder and a Sentence Encoder, the embeddings of entities and sentences are the initial features of graph nodes; 2) **Heterogeneous Graph Encoder**: an iteratively computed graph with $FacetWeight$; and 3) **Extraction & Postprocess**: ranks sentences while minimizing redundancy with Trigram Blocking.

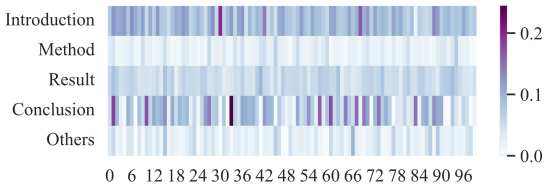


Figure 3: Heat map of five section categories.

four types of learnable edges to link the nodes. To further assess the importance of edges, we infuse both frequency values, such as TF-IDF, and discourse values, such as position and facet importance. To be more specific, we build the following edge types:

Ent-Sent Construct an edge if an entity occurs in a sentence. For an entity node $v_i = \{w_{i0}, \cdots, w_{im}\}$ and a sentence node $v_j = \{w_{j0}, \cdots, w_{jl}\}$, the weight of edge is $e_{ij} = \sum_{k=0}^{m} tfidf_{ik}/m$, where $tfidf_{ik}$ is the product of term frequency (TF), which is the term count of w_{ik} in v_j, and inverse document frequency (IDF), which measures how uncommon w_{ik} is.

Sent-Sent For two sentence nodes v_j and v_s, the edge weight $w_{js} = f(sim(v_j, v_s))$, (e.g., the cosine distance between their distributed representations).

Sec-Sent For a section node $v_c = \{s_{c0}, \cdots, s_{cn}\}$ and a sentence node v_j, the weight of edge is $w_{cj} = FacetWeight_c \cdot Pos_{cj}$, where $Pos_{cj} = min(pos_{cj}, n - pos_{cj})$ and pos_{cj} denotes the position of sentence j in section c, which follows the idea of the sentence boundary function (Dong et al., 2021), (i.e., sentences closer to the section's boundaries are more important).

Sec-Sec We distinguish two levels of sections to form a finer structure, connecting section nodes hierarchically with edge weights initialized with 1.

3.3 Graph Learning and Predicting

We upgrade node features through a layer of Graph Attention Model (GAT) (Veličković et al., 2018) and Feed-Forward Network (FFN) (Vaswani et al., 2017). When a node v_i aggregates information from its neighbours, attention coefficient α_{ij} with node v_j is calculated as follows:

$$z_{ij} = LeakyReLU(W_a[W_q h_i; W_k h_j]; e_{ij}) \quad (1)$$

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

where W_a, W_q, W_k are trainable weights. And we infuse e_{ij} into original GAT with four multi-dimensional embedding spaces for four types of edges. The multi-head attention and FFN layer can be denoted as:

$$u_i = \|_{k=1}^{K} \sigma(\sum_{j \in \mathcal{N}} \alpha_{ij}^k W^k h_i) \quad (3)$$

$$u_i^{'} = max(0, u_i W_{f1} + b_1) W_{f2} + b_2 \quad (4)$$

At the end of aggregation, node v_i is updated as $h_i^{'} = u_i^{'} + h_i$. The nodes are upgraded iteratively as shown at the top of Figure 2. The outputs from the sentence nodes H_s are then forwarded to a classification layer to receive scores.

Eventually, we get all the predicted scores of sentences. Following the previous work (Liu and Lapata, 2019), trigram blocking is used to reduce redundancy. We rank sentences by their scores, and a sentence can only be extracted if there are no trigram overlaps between it and other sentences that have already been extracted.

4 Experiment Setup

4.1 Dataset

CORD-SUM is reorganized from CORD-19 (Wang et al., 2020b) (by September, 2021). Data cleaning included removing papers with no titles, abstracts, or section breaks, or written in languages other than English. Useless information such as authors and publication dates are also removed. Each item is a pair of a paper and its corresponding author-written abstract. The dataset has 122726 articles that we split for training, validation, and testing, in respective percentages of 70%, 15%, and 15%.

We explored the document length distribution in existing summarization datasets as Table 1, including news datatsets (CNN/Dailymail (Hermann et al., 2015), NYTimes (Sandhaus, 2008), XSUM (Narayan et al., 2018)) and scientific datasets (PubMed, ArXiv (Cohan et al., 2018), SciSummNet (Yasunaga et al., 2019), SciTldr (Cachola et al., 2020), FacetSum (Meng et al., 2021)). The document length and abstract length of scientific papers are both much longer than news articles. We evaluate SAPGraph on CORD-SUM as well as on ArXiv to measure the performance on both medical domain papers and general papers.

Type	Dataset	#Pairs	Avg W/D	Avg W/A
News	NYTimes	655K	549	40
	CNN	92K	656	43
	DailyMail	219K	693	52
	XSUM	226K	431	23
Scientific Papers	PubMed	133K	3016	203
	ArXiv	215K	4938	220
	SciSummNet	1.0K	4720	151
	SciTldr	3.2K	4983	21
	FacetSum	5.8K	6827	290
	CORD-SUM	**123K**	**3806**	**223**

Table 1: News and Scientific Papers datasets statistics of size and text length. W/D and W/A denote words per document and words per abstract, respectively.

4.2 Toplines

We obtain sentences greedily from documents by maximizing the similarity between the gold summary and the whole oracle sentence set, following the work of Nallapati et al. (2017), denoted as Oracle-D. Additionally, we attempt to select the most similar sentence from the document for every sentence in the gold summary. We denote a summary generated from these sentences by Oracle-S. And the above similarity is calculated by ROUGE-1+ROUGE-2 scores. The oracles can be seen as the toplines. In our experiments, we choose Oracle-S as the target to supervise all models, because of its better performance on ROUGE and BERTScore.

4.3 Baselines

We choose from heuristics, unsupervised and supervised state-of-the-art summarization models for extractive summarization.

4.3.1 Heuristics Models

We randomly select 10 sentences from the source text and concatenate them as a summary, denoted as **Random-10**. We also select the first 10 sentences as **Lead-10**. To prove the effectiveness of section information in summarization task, we also implement **SecLead-3-10** to select the first 3 sentences from each section and overall limit to 10 sentences.

4.3.2 Unsupervised Models

We choose three graph-based ranking algorithms: **TextRank** (Mihalcea and Tarau, 2004) is to build a classical inter-sentence graph to measure a sentence node centrality. Unlike TextRank, **PacSum** (Zheng and Lapata, 2019) uses BERT to initialize node embedding and value sentence position in the document as a decent feature. **HipoRank** (Dong et al., 2021) presents a two-level hierarchical graph of the document introducing section-level information, and extends the model into scientific papers.

4.3.3 Supervised Models

We explore the supervised summarizing systems as pretrained models and graph models. For pretrained models, **BERTSUMEXT** is a strong baseline for extractive summarization. Its sentence classifier is built on top of a Transformer stack. To alleviate the weakness of the length constraint of BERT, we also use **Longformer** with sliding window attention mechanism, to suit Transformer to

Type	Models	CORD-SUM				ArXiv			
		R-1	R-2	R-L	BS	R-1	R-2	R-L	BS
Oracle	Oracle-D	59.36	32.63	27.71	84.49	38.90	13.28	34.51	85.41
	Oracle-S	59.31	32.31	35.83	88.44	54.96	27.37	49.89	87.17
Heuristics	Random-10	37.62	9.83	17.00	83.41	34.39	8.95	30.90	82.04
	Lead-10	37.57	11.14	18.12	83.17	34.88	10.45	31.52	82.99
	SecLead-3-10	38.50	11.45	18.94	83.33	34.99	11.37	31.76	82.82
Unsupervised	TextRank (Mihalcea and Tarau, 2004)	42.54	14.67	21.37	84.51	38.17	11.80	32.73	82.49
	PacSum (Zheng and Lapata, 2019)	39.55	11.70	18.40	83.73	38.42	11.17	34.70	83.37
	HipoRank (Dong et al., 2021)	44.09	15.52	20.41	84.84	38.72	12.29	34.94	83.02
Supervised	BertSumExt (Liu and Lapata, 2019)	40.20	13.43	20.81	84.11	34.66	11.36	31.45	83.15
	LongformerSumExt	42.34	13.28	20.72	83.70	35.93	12.37	32.66	83.46
	HSG (Wang et al., 2020a)	44.01	16.23	20.95	84.86	39.68	**14.64**	35.90	84.27
Ours	SAPGraph-Longformer	45.43	16.64	20.95	85.28	35.24	10.25	31.69	82.70
	SAPGraph	**47.10**	**18.53**	**22.30**	**85.74**	**41.22**	14.43	**37.30**	**84.48**

Table 2: Limited-length summaries scores on CORD-SUM and ArXiv, where R-1,2,L denote ROUGE-1,2,L and BS denotes BERTScore. **Bold** denotes the best score and underline indicates the second best score.

long text. To better study the head distribution problem, we set the input length as 4096 tokens, which can cover most of the source documents.

For supervised graph systems, **HSG** models relations between sentences based on their common words, with no direct connection between sentences. It tries to connect every sentence through words in the whole document, but catches no extra structure information of facets and knowledge. We also present a pretrained model + graph model. As we choose Longformer to encode the article and pick [CLS] embedding in front of each sentence as the sentence node embedding. It is challenging and error-prone to train two different models together. Therefore, we adopted modifications such as two-stage learning and residual connection (Lin et al., 2021) from Longformer to SAPGraph consequently in an effort to combine the strength of Transformer with graph representation, encompassing inner-sentence and inter-sentence data.

4.4 SAPGraph Implementation

For graph model initialization, we extract entities with SciSpacy[3]. Especially for our CORD-SUM experiment, we select the extraction package just for medical entities. The vocabulary is limited to 50,000, and we add all words in entities to mitigate out-of-vocabulary (OOV) problem, and then initialize words with 300-dimension GloVe embeddings (Pennington et al., 2014). In our experiment, the vocabulary can cover 87% of all words. For each document graph, we provide 100 sentences with 50 words each as input. BERT and Longformer both tokenized raw text into tokens at the max length of 4096.

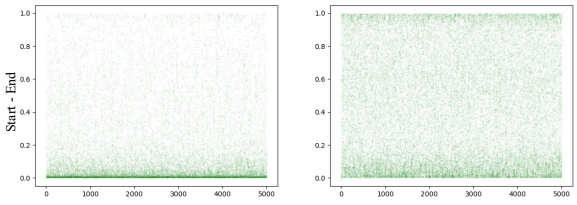
We have 128 dimensions in vectors representing sentences and entity nodes, and 50 dimensions in vectors standing for edges. Each GAT layer has 8 heads and the hidden size is $d_{h-GAT} = 64$. The hidden size for FFN layers is $d_{h-FFN} = 512$.

During training, we set the batch size as 36 within 10 epochs on a single GeForce RTX 3090. We apply Adam optimizer (Kingma and Ba, 2014) with a learning rate of 2e-3 for the graph model, and 5e-5 for the pretrained model. Outputs are limited to ten sentences for consistent comparisons. The training continues until the loss function stops decreasing for three consecutive epochs.

5 Results and Analysis

5.1 Oracle Analysis

We sample 5000 items from CORD-SUM to measure Oracle performance. Figure 4 demonstrates that the sentence positions of the two Oracle distributions show significant variation. Oracle-D is more likely to be head-distributed, while Oracle-S shows a head-to-tail distribution and is more uniformly organized.



(a) Oracle-D (b) Oracle-S

Figure 4: Oracle sentence distributions over a paper.

From Table 2, we also can see that Oracle-S performs better on R-L and BS than Oracle-D, while their R-1 and R-2 scores are close on CORD-SUM. The results on both datasets show Oracle-S is more long-text-friendly. Therefore, we choose labels

[3]https://allenai.github.io/scispacy/

from Oracle-S to train our models to avoid further head distribution problem.

5.2 Models Performance

Through the comparison of Random-10 and Lead-10 results, we have verified the importance of head sentences in a scientific document. We observe that SecLead-3-10 achieves the best performance on ROUGE among the three heuristics models. From the ROUGE scores of SecLead-3-10 and Lead-10, we are able to determine that uniform selection of sentences from different sections can generate better summaries. Once again, this confirms our hypothesis that summarization covering the content of different sections leads to better performance.

The results in Table 2 prove that Transformer's word-level attention is inferior to graph models. Compared with LongformerSUMEXT, our graph model achieves 4.76/5.25/1.58/2.04 improvements of R-1,2,L and BERTScore on CORD-SUM, and 5.29/2.06/4.64/1.02 on ArXiv, respectively. At the same time, SAPGraph outperforms HSG on CORD-SUM, which is also a supervised graph model, with 3.09/2.3/1.35/0.88 on R-1,2,L and BERTScore, respectively. The results indicate that structure information of facets and knowledge can help SAPGraph surpass existing models, especially on medical domain papers.

These results also show that the graph model can pay more attention to sentence semantics and learn more about cross-sentence relationships, so it performs better on the scientific paper summarization task even with much fewer parameters (110M for BERT and 16M for SAPGraph).

From the result of SAPGraph-Longformer, we try to get sentence embedding from Longformer instead of our Local Encoder. But it seems an embedding from document-scale may mislead the training of GNN. So, the integration method of pre-trained models and graph models is still a subject worthy of further exploration.

In conclusion, the results show that structure information is very important for scientific paper summarization, and our graph structure can explicitly and effectively utilize facet structure information, making the summaries more interpretable.

5.3 Discussion

5.3.1 Node Analysis

SAPGraph can demonstrate competitive or even better performance by adding a small number of section nodes and a considerably smaller number of entity nodes than word nodes. The average number of nodes in SAPGraph is 41.5% less than in HSG (448 vs 766). Redundant word nodes are removed with the introducing of structure information.

In our experiments, we also find that the entity nodes with more degrees have a more important role in the graph. They help establish more sentence connections, and can provide more diverse and rich topological information of knowledge, in addition to sentence similarity. The entities of the two datasets vary significantly, due to the differences of each field, which is why entities have a strong ability to represent the content of papers. Example entities are shown in Appendix C.

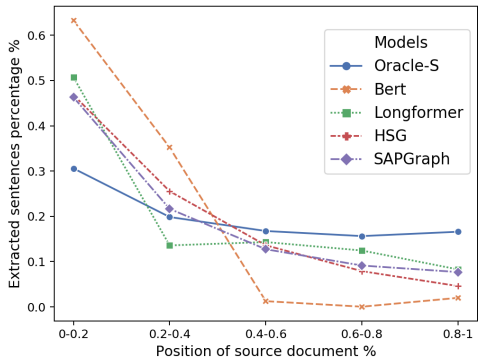


Figure 5: Summary sentences distributions of models.

5.3.2 Summary Distribution

The distribution of the summary's sentence positions in the source document can reflect the coverage of the summary. We calculate the distribution of Oracle-S and the other four models on the CORD-SUM test set.

As shown in Figure 5, the x-coordinate represents the position of the summary sentence in the article and the y-coordinate denotes the proportion of the summary sentence. For example, over 60% of the summary sentences generated by BERTSUMEXT locate in the top quintile of the article,

Models	PCCs	p-value
Bert	0.95174	0.01263
Longformer	0.96890	0.00655
HSG	0.96401	0.00815
SAPGraph	**0.99076**	0.00107

Table 3: Pearson Correlation Coefficients (PCCs) of summary distribution of CORD-SUM test set between models and Oracle-S.

Section	Subsection	Text	Oracle-S	HSG	SAP-Graph
Introduction	-	the pandemic peak of coronavirus disease-19 (covid-19) has put the italian healthcare system into massive stress…		√	
		hospitals were then forced to make room for medical and intensive care wards dedicated to patients with suspect or confirmed infection by severe acute respiratory syndrome coronavirus-2 (sars-cov-2).		√	
		despite the huge efforts, patients admitted with covid-19 experienced a high burden of respiratory failure and high mortality rates.		√	
		covid-19-associated mortality is the highest in older patients, in those with multimorbidity and cardiometabolic diseases.		√	
		furthermore, significant differences in clinical presentation and course of the patients hospitalized for covid-19…	√	√	
		the primary objective of this retrospective single-center study, conducted in the covid-19 hospital hub of an area of …	√	√	√
		the secondary objectives were to describe the prevalence of older age, frailty, and multimorbidity in patients admitted for suspect covid-19, and their association with hospital mortality.		√	√
Method	Study setting & population	the study was conducted at the geriatric-rehabilitation department of parma university-hospital, in the city of parma, emilia-romagna region.		√	√
		inclusion criteria for this retrospective study were age ≥18 years old and presence of symptoms and chest hrct…		√	
	Data collection	information collected on the findings of the chest hrct performed on admission included the presence of ground-glass opacities, , the presence of consolidations, and the covid-19 visual score.			√
	Statistical analysis	linear regression and binary logistic regression were used for age- and sex-adjusted comparisons.			√
Result	Temporal trends	a total number of 1634 patients were admitted to our department from the establishment of the covid-19 care path…		√	
		among them, 1487 clinical records were screened for inclusion.			√
		the final study population was composed of 1264 patients (711 m, 553 f) with clinical and radiological features…	√		√
		patients admitted during the second phase exhibited lower needs of oxygen support (maximum oxygen flow administered during stay 36%, iqr 28–75, vs. 50%, iqr 28–75, age-and sex-adjusted p < 0.001), reduced prescription of non-invasive…	√		
	Role of multimorbidity	the number of participants with multimorbidity (≥2 chronic diseases) was 923 (73%), with a prevalence increasing from…	√		
		patients with multimorbidity were older, mostly of female gender, and disabled.	√		√
	Factors associated with adverse	the clinical and anamnestic factors associated with hospital mortality were tested with binary logistic regression models…	√		
		notably, admission during the second phase of the pandemic peak was inversely associated with mortality in the total population and in positive patients.	√		√
	Clinical presentation…	a total number of 807 patients (339 f, 468 m) tested positive at rt-pcr for sars-cov-2 detection on nasopharyngeal swabs performed the day of admission.			√
Discussion	-	this study provides an overview of the clinical characteristics and outcomes of a large group of patients admitted…		√	
Conclusion	-	in our experience during the first pandemic wave of covid-19 in northern italy, older patients, especially frail, multimorbid, , and of female gender, were more frequently hospitalized during the second phase of the outbreak and …	√		
		multimorbidity and dependency in daily activities were independently associated with in-hospital mortality…	√		

Table 4: HSG and SAPGraph outputs compared with Oracle-S (✓means the sentence is included in the summary).

which exposes an overwhelming head distribution problem. A relatively flat line, similar to the Oracle-S, indicates that the summaries are more comprehensive. In Table 3 we also calculate the Pearson Correlation Coefficient (PCCs) which shows that the summaries obtained by SAPGraph are the closest to the Oracle-S distribution, owing to the introduced structure information. To better demonstrate the high quality of our produced summaries, we also report a case study in Section 5.4.

5.4 Case Study

As can be seen from the case in Table 4, the sentences predicted by both HSG and SAPGraph account for a fraction of the Introduction, including the background and goals of the paper. However, the sentences predicted by HSG tend to be distributed in the first half of the paper, and prominently so in the Introduction. Although the content in Introduction is important, SAPGraph can still pay more attention to the other sections, thus having more sentences hit in Oracle. This is the result of comprehensive consideration of the structure of the full document. It is obvious that such a summary can meet the expectations of a paper abstract. The background, motivation, method, and conclusion are quickly given to readers to determine whether further reading or reference is required.

5.5 Ablation Study

Models	R-1	R-2	R-L
SAPGraph	47.10	18.53	22.30
w/o sec pooling	46.64	18.04	21.96
w/o $FacetWeight$	46.02	17.72	21.85
w/o sec node	46.20	17.62	21.67
w/o ent node	45.58	17.29	21.34
only sentence node	45.23	16.83	21.34

Table 5: Ablation study on section embedding and node types on CORD-SUM.

We analyze the importance of different nodes for model training (Table 5). Specifically, we focus on verifying the roles of entity and section nodes, and feature embedding methods. We try not to use a pooling method for section embedding, and replace it with section name embedding, since the name can represent the main section information empirically. However, from the result, we speculate that the section name does not contain enough guiding significance for sentence classification. Therefore, section pooling was chosen over section name. $FacetWeight$ can also provide guidance from section nodes to sentence nodes. Further experiments on it can be seen in Appendix D. Because the sentence node is a necessary component of the graph, we removed the entity nodes first and then the section nodes. The results show that both types of nodes are essential in model training.

6 Conclusion

In this paper, we propose SAPGraph, a structure-aware heterogeneous graph model for scientific paper extractive summarization. SAPGraph can generate more comprehensive summaries while operating on much smaller graphs, with the well-designed graph construction considering the explicit structure of facets and implicit structure of knowledge. Along with SAPGraph, we propose CORD-SUM, a large structure-rich medical-domain scientific paper summarization dataset. Detailed experiments and case studies prove the effectiveness of SAPGraph on alleviating the head distribution problem. SAPGraph can generate more comprehensive summaries on CORD-SUM and ArXiv datasets than previous works. In the future, we will explore how to automatically learn graph structure and find a more effective way to integrate pretrained models and SAPGraph.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant No. 62176024); Beijing Municipal Science & Technology Commission [Grant No. Z181100001018035]; Engineering Research Center of Information Networks, Ministry of Education; the Fundamental Research Funds for the Central Universities (2021XD-A01-1).

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. In *Proc. of EMNLP*.

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

Isabel Cachola, Kyle Lo, Arman Cohan, and Daniel S Weld. 2020. Tldr: Extreme summarization of scientific documents. In *Proc. of EMNLP Findings*.

Shuyang Cao and Lu Wang. 2022. Hibrids: Attention with hierarchical biases for structure-aware long document summarization. *arXiv preprint arXiv:2203.10741*.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proc. of NAACL*.

Peng Cui and Le Hu. 2021. Sliding selector network with dynamic memory for extractive summarization of long documents. In *Proc. of NAACL*.

Yue Dong, Andrei Mircea, and Jackie Chi Kit Cheung. 2021. Discourse-aware unsupervised summarization for long scientific documents. In *Proc. of EACL*.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*.

Alexios Gidiotis and Grigorios Tsoumakas. 2020. A divide-and-conquer approach to the summarization of long documents. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Quentin Grail, Julien Perez, and Eric Gaussier. 2021. Globalizing bert-based transformer architectures for long document summarization. In *Proc. of EACL*.

J Hartley. 2014. Current findings from research on structured abstracts: an update. *Journal of the Medical Library Association: JMLA*.

James Hartley, Matthew Sydes, and Anthony Blurton. 1996. Obtaining information accurately and quickly: are structured abstracts more efficient? *Journal of information science*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Proc. of NeurIPS*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*.

Yin Jou Huang and Sadao Kurohashi. 2021. Extractive summarization considering discourse and coreference relations based on heterogeneous graph. In *Proc. of EACL*.

D. Kingma and J. Ba. 2014. Adam: A method for stochastic optimization. *Computer Science*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proc. of ACL*.

Xinnian Liang, Shuangzhi Wu, Mu Li, and Zhoujun Li. 2021. Improving unsupervised extractive summarization with facet-aware modeling. In *Proc. of ACL Findings*.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proc. of NAACL*.

Yuxiao Lin, Yuxian Meng, Xiaofei Sun, Qinghong Han, Kun Kuang, Jiwei Li, and Fei Wu. 2021. Bertgcn: Transductive text classification by combining gnn and bert. In *Proc. of ACL Findings*.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proc. of EMNLP*.

Rui Meng, Khushboo Thaker, Lei Zhang, Yue Dong, Xingdi Yuan, Tong Wang, and Daqing He. 2021. Bringing structure into summaries: a faceted summarization dataset for long scientific documents. In *Proc. of ACL*.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proc. of EMNLP*.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Proc. of AAAI*.

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 *Proc. of EMNLP*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proc. of EMNLP*.

Deepika S, Lakshmi Krishna N, and Shridevi S. 2021. Extractive text summarization for covid-19 medical records. In *2021 Innovations in Power and Advanced Computing Technologies (i-PACT)*.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*.

Sandeep Subramanian, Raymond Li, Jonathan Pilault, and Christopher Pal. 2020. On extractive and abstractive neural document summarization with transformer language models.

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

P Veličković, A Casanova, Pietro Lio, G Cucurull, A Romero, and Y Bengio. 2018. Graph attention networks.

Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuanjing Huang. 2020a. Heterogeneous graph neural networks for extractive document summarization. *arXiv preprint arXiv:2004.12393*.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Michael Kinney, et al. 2020b. Cord-19: The covid-19 open research dataset. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*.

Wen Xiao and Giuseppe Carenini. 2019. Extractive summarization of long documents by combining global and local context. In *Proc. of EMNLP*.

Michihiro Yasunaga, Jungo Kasai, Rui Zhang, Alexander R Fabbri, Irene Li, Dan Friedman, and Dragomir R Radev. 2019. Scisummnet: A large annotated corpus and content-impact models for scientific paper summarization with citation networks. In *Proc. of AAAI*.

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*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *Proc. of ICML*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Hao Zheng and Mirella Lapata. 2019. Sentence centrality revisited for unsupervised summarization. In *Proc. of ACL*.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuan-Jing Huang. 2020. Extractive summarization as text matching. In *Proc. of ACL*.

A Keyword List for Section Facet Classification

Category	Keyword
Introduction	intro, purpose, background
Method	design, method, approach
Result	result, find, discuss, analy
Conclusion	conclu, future
Others	case, statement, covid-19, health...

Table 6: Keywords used in section classification for different facets. The words mismatched in the other four categories with the highest frequencies are listed in Others.

From CORD-SUM dataset we randomly sample 80 articles and perform human evaluations. We ask four human evaluators to classify each section in the article by reading the title and content of the section. Each evaluator is responsible for labeling 40 articles. So each article will be labeled by two evaluators. If there exist conflicts, all evaluators will have a discussion until an agreement is achieved. The human-labeled results are treated as the ground truth. The average accuracy of our method can reach 90.3%.

B Full Results

We report full results of ROUGE scores on CORD-SUM and ArXiv, as well as ablation study on CORD-SUM as below in Tables 7, 8 and 9.

C Entity Examples

Figures 6 and 7 show most frequent entities in CORD-SUM and ArXiv respectively.

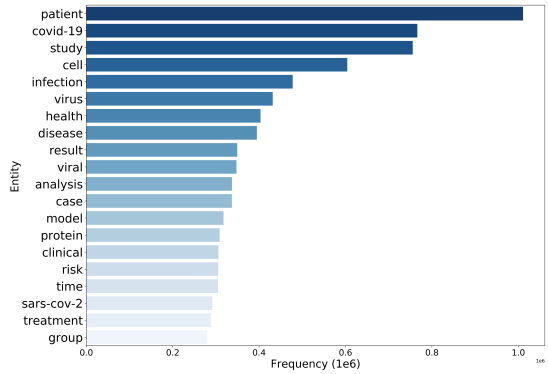


Figure 6: Top 20 frequent entities in CORD-SUM vocabulary.

D FacetWeight Discussion

$FacetWeight$ is a crucial part of our experiment, we get the facet distribution through statistical calculation. Still, we want to discuss the influence of different $FacetWeight$ settings. While searching the best settings, we plus/minus the same proportion to Introduction and Conclusion together, since the two types of sections are almost equally important. Results of Table 10 show that our setting surely is the most reasonable one.

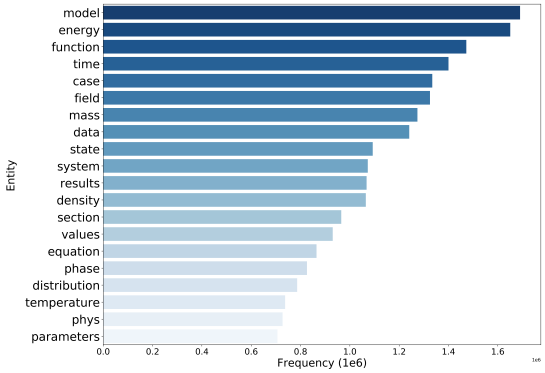


Figure 7: Top 20 frequent entities in ArXiv vocabulary.

Models	R-1			R-2			R-L		
	P	R	F	P	R	F	P	R	F
Oracle-D	61.01	61.92	59.36	34.27	33.44	32.63	29.54	28.52	27.71
Oracle-S	59.77	61.09	59.31	32.48	33.34	32.31	36.14	36.95	35.83
Random-10	38.45	41.06	37.62	10.14	10.67	9.83	17.75	18.56	17.00
Lead-10	43.86	35.35	37.57	13.20	10.40	11.14	21.31	17.06	18.12
SecLead-3-10	43.69	37.13	38.50	13.02	11.07	11.45	21.57	18.32	18.94
TextRank (Mihalcea and Tarau, 2004)	46.25	42.45	42.54	16.20	14.47	14.67	23.50	21.34	21.37
PacSum (Zheng and Lapata, 2019)	41.18	40.32	39.55	12.24	11.92	11.70	19.30	18.72	18.40
HipoRank (Dong et al., 2021)	44.95	45.97	44.09	15.91	16.11	15.52	20.80	21.41	20.41
BertSumExt (Liu and Lapata, 2019)	**48.80**	36.13	40.20	16.40	12.01	13.43	**25.32**	18.74	20.81
LongformerSumExt	44.02	43.53	42.34	13.80	13.69	13.28	21.60	21.37	20.72
HSG (Wang et al., 2020a)	41.16	51.61	44.01	15.19	19.08	16.23	19.68	**24.75**	20.95
SAPGraph-Longformer	44.00	51.08	45.43	16.24	18.63	16.64	22.44	23.61	20.95
SAPGraph	46.30	**52.16**	**47.10**	**18.45**	**20.39**	**18.53**	22.20	24.67	**22.30**

Table 7: Full results of limited-length ROUGE scores on CORD-SUM.

Models	R-1			R-2			R-L		
	P	R	F	P	R	F	P	R	F
Oracle-D	48.35	36.94	38.9	17.26	12.47	13.28	42.98	32.75	34.51
Oracle-S	57.18	54.81	54.96	28.52	27.73	27.37	51.9	49.76	49.89
Random-10	28.58	48.30	34.39	7.39	12.76	8.95	25.7	43.29	30.90
Lead-10	27.53	53.63	34.88	8.15	16.54	10.45	24.90	48.41	31.52
SecLead-3-10	26.22	59.51	34.99	8.44	19.80	11.37	23.81	53.95	31.76
TextRank (Mihalcea and Tarau, 2004)	**34.13**	47.10	38.17	10.54	14.60	11.80	29.31	40.34	32.73
PacSum (Zheng and Lapata, 2019)	33.33	49.28	38.42	9.62	14.58	11.17	30.12	44.45	34.70
HipoRank (Dong et al., 2021)	33.76	49.30	38.72	10.64	15.85	12.29	**30.50**	44.40	34.94
BertSumExt (Liu and Lapata, 2019)	25.82	59.39	34.66	8.35	20.06	11.36	23.44	53.86	31.45
LongformerSumExt	26.65	**61.34**	35.93	9.08	21.64	12.37	24.24	**55.69**	32.66
HSG (Wang et al., 2020a)	30.90	60.97	39.68	11.31	**22.90**	**14.64**	27.98	55.08	35.90
SAPGraph-Longformer	26.81	56.88	35.24	7.76	16.76	10.25	24.13	51.05	31.69
SAPGraph	33.31	59.06	**41.22**	**11.59**	20.98	14.43	30.17	53.36	**37.30**

Table 8: Full results of limited-length ROUGE scores on ArXiv.

Models	R-1			R-2			R-L		
	P	R	F	P	R	F	P	R	F
SAPGraph	46.30	52.16	47.10	18.45	20.39	18.53	22.20	24.67	22.30
no sec pooling	46.17	51.34	46.64	18.03	19.75	18.04	21.96	24.16	21.96
no $FacetWeight$	45.30	51.35	46.02	17.66	19.66	17.72	21.82	24.35	21.85
no sec node	45.04	51.60	46.20	17.33	19.59	17.62	21.31	24.21	21.67
no ent node	44.46	51.21	45.58	17.04	19.33	17.29	21.53	24.50	21.82
only sentence	44.15	50.31	45.23	16.54	18.66	16.83	21.00	23.78	21.34

Table 9: Full results of ablation study on section embedding and node types.

	Introduction	R-1	R-2	R-L
Origin set	[0.35,0.1,0.15,0.35,0.05]	47.1	18.53	22.30
Intro/Conclu-0.5	[0.3,0.15,0.2,0.3,0.05]	46.43	17.80	21.90
Intro/Conclu+0.5	[0.4,0.05,0.1,0.4,0.05]	46.04	17.29	21.49
Intro/Conclu+1	[0.45,0,0.05,0.45,0.05]	44.49	15.78	20.51

Table 10: Results of different settings of $FacetWeight$ on graph edges.