GraphLSS: Integrating Lexical, Structural, and Semantic Features for Long Document Extractive Summarization

Margarita Bugueño[1,2], Hazem Abou Hamdan[2], Gerard de Melo[1,2]
[1]Hasso Plattner Institute (HPI), [2]University of Potsdam
Potsdam, Germany
{margarita.bugueno, gerard.demelo}@hpi.de

Abstract

Heterogeneous graph neural networks have recently gained attention for long document summarization, modeling the extraction as a node classification task. Although effective, these models often require external tools or additional machine learning models to define graph components, producing highly complex and less intuitive structures. We present GraphLSS, a heterogeneous graph construction for long document extractive summarization, incorporating Lexical, Structural, and Semantic features. It defines two levels of information (words and sentences) and four types of edges (sentence semantic similarity, sentence occurrence order, word in sentence, and word semantic similarity) without any need for auxiliary learning models. Experiments on two benchmark datasets show that GraphLSS is competitive with top-performing graph-based methods, outperforming recent non-graph models. We release our code on GitHub[1].

1 Introduction

Extractive document summarization condenses documents into summaries by selecting only the most relevant sentences. One intuitive approach is to model cross-sentence relationships using graph structures, which offer unique advantages over traditional sequence-based models. Graph-based methods provide flexibility in handling varying document lengths and explicitly capture multi-granularity text relationships. This structured representation enhances document analysis, enabling improved contextual understanding and deeper insights into document structure (Cui et al., 2020; Phan et al., 2022; Bugueño and de Melo, 2023). While prior work considered homogeneous graphs (Tixier et al., 2017; Xu et al., 2020), recent heterogeneous graph proposals have shown high effectiveness (Wang et al., 2020; Jia et al., 2020), as they define complex relationships between multiple semantic units and capture long-distance dependencies. Despite their success in summarizing long documents such as scientific papers, many efforts have been made to devise more effective graph constructions. These vary in their definitions of nodes, often requiring external tools or additional machine learning models (Cui et al., 2020), and of edges, which despite being effective, may lead to complex structures that reduce the intuitiveness of the resulting graphs (Zhang et al., 2022).

This paper introduces GraphLSS, a graph construction that avoids the need for external learning models to define nodes or edges. GraphLSS utilizes **L**exical, **S**tructural, and **S**emantic features, incorporating two types of nodes (sentences and words) and four types of edges (sentence order, sentences semantic similarity, words semantic similarity, and word–sentence associations). We limit word nodes to nouns, verbs, and adjectives for their high semantic richness (Bugueño and Mendoza, 2020; Xiao and Carenini, 2019). Our document graphs are processed with GAT (Veličković et al., 2018) models on two summary benchmarks, PubMed and arXiv, which are preprocessed and labeled by us.

Our contributions are: **i.** A novel heterogeneous graph construction using lexical, structural, and semantic features, **ii.** State-of-the-art results on both benchmarks compared to previous graph strategies and recent non-graph methods, **iii.** We share our code, including calculated extractive labels and graph-data creation pipeline, on GitHub[1] for reproducibility and collaboration.

2 Previous Work

Graph Structure Developing an effective graph structure for summarization has been challenging, leading to a proliferation of diverse approaches. Wang et al. (2020) proposed connecting sentence nodes to word nodes by establishing undirected as-

[1] https://github.com/AbouClaude/GraphLSS

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers), pages 797–804
April 29 - May 4, 2025 ©2025 Association for Computational Linguistics

sociations with the contained words. Subsequently, Jia et al. (2020) extended this by introducing named entity nodes and three other edge types: directed edges for tracking subsequent named entities and words in a sentence, directed edges for entities and words within a sentence, and undirected edges for sentence pairs with trigram overlap.

Topic-GraphSum (Cui et al., 2020) was one of the first attempts to apply graph strategies to long document extractive summarization. It integrated a joint neural topic model to discover latent topics in a document, defining these as intermediate nodes to capture inter-sentence relationships across various genres and lengths. SSN (Cui and Hu, 2021) defined a sliding selector network with dynamic memory. SSN splits a given document into multiple segments, encodes them with BERT (Devlin et al., 2019), and selects salient sentences. Instead of representing the document as a graph, it uses a graph-based memory module, updated iteratively with a GAT (Veličković et al., 2018), to allow information to flow across different windows. HeterGraphLongSum (Phan et al., 2022) utilized words, sentences, and passages as nodes, while considering undirected edges for words in sentences, and directed edges for words in passages and passage to sentences. Instead of pre-trained embeddings, it used CNNs and bidirectional LSTMs for node encoding, yielding outstanding results. MTGNN-SUM (Doan et al., 2022) achieved similar results by capturing both inter and intra-sentence information when combining a homogeneous graph of sentence nodes with a heterogeneous graph of words and sentences, as in Wang et al. (2020).

Recent studies underscore the importance of structural information in long document summarization. HEGEL (Zhang et al., 2022) modeled documents as hypergraphs, with edges capturing keyword coreference, section structure, and latent topics. CHANGES (Zhang et al., 2023) introduced a sentence–section hierarchical graph, creating fully connected subgraphs for sentences and sections, and linking sentences to their sections.

Sentence Labeling There is no consensus on generating extractive ground truth labels. Most previous work (Jia et al., 2020; Zhang et al., 2022; Wang et al., 2024) used the Nallapati et al. (2017) greedy approach without specifying the ROUGE n-gram level, which significantly impacts sentence classifier performance. Some methods (Wang et al., 2020; Doan et al., 2022; Zhang et al., 2023) selected sentences by maximizing the ROUGE-2 score against the gold summary Liu and Lapata (2019), while others (Cui et al., 2020; Cui and Hu, 2021; Phan et al., 2022) used pre-labeled benchmarks (Xiao and Carenini, 2019) which maximized ROUGE-1. Cho et al. (2022) maximized the average of ROUGE-1 and ROUGE-2.

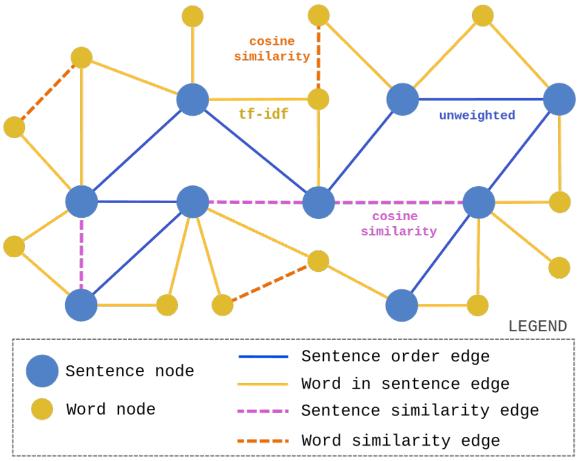

Figure 1: GraphLSS construction. Sentence order edges are unweighted to preserve document structure, word in sentence edges are weighted using tf-idf to reflect word importance, and similarity edges between words and between sentences are determined using cosine similarity.

3 GraphLSS

Graph Construction We propose a heterogeneous model that represents documents as undirected graphs, $G = (V, E)$. We use sentences and words as nodes, $V = V_\text{s} \cup V_\text{w}$, and four edge types to capture Lexical, Structural, and Semantic features, as $E = \{E_\text{ns}, E_\text{ss}, E_\text{ws}, E_\text{ww}\}$. Here, V_s corresponds to the n sentences in the document, and V_w denotes the set of m unique words of the document, limited to the most semantically rich ones, i.e., nouns, verbs, and adjectives[2] as in Bugueño and Mendoza (2020). For connections between nodes, boolean unweighted edges E_ns indicate the sequential order of sentences within a document, while E_ss includes sentence pair edges weighted by cosine similarity within a predefined window size. This constraint preserves local similarity and prevents dense graphs. To ensure that only strongly correlated sentences are connected, edges are established only when the cosine similarity surpasses a predefined threshold. Additionally, E_ws denotes

[2] Adverbs are excluded since they primarily serve as complements for adjectives and verbs rather than standalone semantic entities.

words in sentence edges weighted by tf-idf scores, and E_{ww} represents word pair edges using cosine similarity. The construction of GraphLSS is illustrated in Figure 1.

Adaptive Class Weights Our graphs are processed by a heterogeneous GAT (Veličković et al., 2018) followed by a sentence node classifier to conduct the extractive summarization. Since the extractive ground truth labels for long documents are highly imbalanced, we optimize the model using weighted cross-entropy loss. We assign initial class weights to relevant and irrelevant sentences, employing adaptive class weights for the relevant class and static weights for non-summary sentences:

$$\lambda^{i+1} = \lambda^{i} - \left(\tau - \frac{\tau}{\log(\tau)}\right), \qquad (1)$$

with τ the portion of sentences predicted as relevant for the summary over all the existing sentences.

4 Experiments

Datasets We use two publicly available benchmarks for long document summarization, PubMed and arXiv (Cohan et al., 2018). Both comprise scientific English articles and are widely used by previous work. Statistics are given in Appendix A.

Extractive Labels Extractive labels are obtained by greedily optimizing the ROUGE-1 score, an intuitive and widely used method that allows us to label more sentences as relevant than alternative strategies. Although we adopted the same labeling approach, we identified substantial sentence tokenization errors in the dataset from Xiao and Carenini (2019). Hence, we independently preprocessed and labeled the data, removing duplicates, empty samples, and instances where abstracts exceeded source document lengths. We also replaced special characters (e.g., \, . . . , », “”, \n) with blanks. We applied sentence tokenization using NLTK and merged particularly short sentences with their preceding ones (cf. Appendix A). For word node definitions, we converted sentence text to lowercase, removing non-ASCII characters, punctuation, and stopwords. The resulting graph datasets are described in Table 1.

Comparison Methods For a more detailed comparative analysis with the models that achieved the best benchmark results (Topic-GraphSum, SSN, and HeterGraphLongSum), we also executed our model using the preprocessed data and sentence-level relevance labels provided by Xiao and Carenini (2019). We also include results from recent non-graph extractive summarizers in Table 2 for reference: Lodoss (Cho et al., 2022) learns sentence representations through simultaneous summarization and section segmentation, Topic-Hierarchical-Sum (Wang et al., 2024) uses local topic information and hierarchical extraction modules, and LOCOST (Le Bronnec et al., 2024) is an abstractive summarization model based on state-space models for conditional text generation.

	Nodes		Edges				Disk
Dataset	V_{s}	V_{w}	E_{ns}	E_{ss}	E_{ws}	E_{ww}	[KB]
PubMed	80 34%	156 66%	80 9%	60 6%	738 82%	27 3%	365
arXiv	123 44%	154 56%	122 11%	50 5%	879 83%	10 1%	421

Table 1: GraphLSS statistics for PubMed and arXiv, with average disk usage presented in kilobytes (KB).

Experimental Setup We trained a GAT model (Veličković et al., 2018) with 4 attention heads and 1–2 hidden layers, minimizing binary cross-entropy loss with adaptive class weights (Equation 1). We initialized word nodes using GloVe Wiki-Gigaword 300-dim. embeddings (Pennington et al., 2014) and pre-trained SBERT (`All-MiniLM-L6-v2`) embeddings for sentence nodes (Reimers and Gurevych, 2019). Notably, our word nodes are restricted to the top 50,000 most frequent words in the respective dataset's vocabulary. For establishing E_{ss}, the window size was empirically set at 40% of the total sentence count of the document, a value determined through preliminary experiments to balance local connectivity while preventing overly dense graphs. Within this window, sentence pair edges were created only if their cosine similarity exceeded 0.7, ensuring that only strongly correlated sentences were linked. Further details are given in Appendix B.

5 Results & Analysis

Table 2 presents the results of different approaches, with graph-based models listed first, followed by non-graph baselines as reference, and our results. ROUGE-1/-2/-L F1-score is measured to assess the informativeness and fluency of the summaries.

Summarization Results GraphLSS significantly outperforms all compared approaches in ROUGE-1/-2/-L scores on PubMed and arXiv, effectively

Model	PubMed			arXiv		
	R-1	R-2	R-L	R-1	R-2	R-L
Oracle (Xiao and Carenini, 2019)	55.05	27.48	38.66	53.88	23.05	34.90
→ Topic-GraphSum (Cui et al., 2020) †	⋆48.85	21.76	35.19	46.05	⋆19.97	33.61
→ SSN (Cui and Hu, 2021) †	46.73	21.00	34.10	45.03	19.03	32.58
→ HeterGraphLongSum (Phan et al., 2022) †	⋆48.86	⋆22.63	⋆44.19	⋆47.36	19.11	⋆41.47
→ MTGNN-SUM (Doan et al., 2022)	48.42	22.26	43.66	46.39	18.58	40.50
→ HEGEL (Zhang et al., 2022)	47.13	21.00	42.18	46.41	18.17	39.89
→ CHANGES (Zhang et al., 2023)	46.43	21.17	41.58	45.61	18.02	40.06
→ Lodoss (Cho et al., 2022)	49.38	23.89	44.84	48.45	20.72	42.55
→ Topic-Hierarchical-Sum (Wang et al., 2024)	46.49	20.52	42.06	45.84	19.03	40.36
→ LOCOST (Le Bronnec et al., 2024)	45.70	20.10	42.00	43.80	17.00	39.70
Our Oracle	60.58	36.91	55.32	63.57	30.40	54.10
→ GraphLSS + Labels by Xiao and Carenini (2019) †	47.85	21.74	42.22	45.91	18.35	40.07
→ GraphLSS + Our labels	**51.42**	**24.32**	**49.48**	**55.14**	**23.00**	**50.83**

Table 2: ROUGE F1 results with scores from respective papers. Models using data from Xiao and Carenini (2019) are marked with † for direct comparison. Best results are marked with ⋆, and second-best are underlined. Bold highlights the GraphLSS improvement, whose results are averaged over 3 runs.

identifying relevant sentences in highly imbalanced settings (Equation 1). These results are based on our preprocessing and labeling. The Oracle results using our labels also greatly exceed those achieved with the data by Xiao and Carenini (2019). With the latter labels, GraphLSS remains competitive (especially regarding ROUGE-L), despite not relying on auxiliary tools and models. This demonstrates close alignment with reference summaries in terms of the longest common subsequence, while alternative approaches yield contaminated summaries. Only HeterGraphLongsum surpasses GraphLSS by using CNN and LSTM networks to learn text embeddings from scratch, whereas we leverage pre-trained embeddings to reduce memorization and bias. These results also suggest that GraphLSS, even with pre-labeled data, outperforms recent non-graph models. Other graph methods are included for reference only, as they are not directly comparable due to the use of different labeling strategies in part requiring extrinsic resources.

Labeling Impact Table 2 highlights the significant variability in summarization results, which depend not only on the graph construction and model choice but also on the strategy used for generating extractive labels. This crucial aspect has been overlooked in related work, which often focuses on ROUGE results without considering whether the corresponding methods are using the same labeling methodology. Moreover, preprocessing steps conducted prior to label calculation can also affect the results. Although Xiao and Carenini (2019) and our study aimed to maximize the ROUGE-1 score, the resulting labels differ significantly. Therefore, ensuring comparable experimental setups is essential for accurately evaluating model effectiveness.

Balance of Precision & Recall Table 3 shows that a two-layer heterogeneous GAT outperforms a single-layer GAT on both datasets, indicating the benefit of extended message passing across the multiple semantic units. Additionally, previous work has not adequately addressed the balance between precision and recall, focusing solely on reporting the F1 score without analyzing the individual values and their implications. Our results show that precision and recall are similar for the experiments on PubMed, reflecting a strong alignment between generated and gold summaries for both ROUGE-1 and ROUGE-2. In contrast, recall considerably exceeds precision on the arXiv dataset, suggesting our model retrieves relevant information but generated summaries still harbors additional text. This effect is more pronounced with a two-layer GAT. Interestingly, this discrepancy is not observed when using the pre-labeled data from Xiao and Carenini (2019), where precision and recall are balanced, albeit lower. This suggests that the observed differences are due to data labeling artifacts rather than the graph construction or the GAT model, emphasizing our earlier discussion.

Resources The complexity and richness of the information encoded in our graphs can lead to increased computational costs. While alternative methods consider constructing the graphs on the fly, creating the graphs in advance is often more efficient in a long document setting.

Dataset	L	ROUGE-1			ROUGE-2			Time [h]
		P	R	F1	P	R	F1	
PubMed	1	49.75	50.00	49.92	22.61	24.71	23.17	19.9
	2	52.59	50.11	51.42	23.91	23.82	24.32	26.1
	2 †	46.43	49.42	47.85	22.42	21.14	21.74	26.2
arXiv	1	45.66	66.68	54.23	17.14	30.20	22.31	22.8
	2	45.20	71.04	55.14	17.02	35.74	23.00	31.9
	2 †	44.88	47.04	45.91	19.96	16.99	18.35	32.2

Table 3: ROUGE scores as precision (P), recall (R), and F1-score (F1). L indicates the number of GAT layers employed, and † marks results using data from Xiao and Carenini (2019).

This strategy incurs the graph creation cost only once, significantly reducing computational overhead by eliminating the need for reconstruction in each epoch and model variant. Our experiments show that storage demands primarily arise from high-dimensional node embeddings, while edges require significantly less space, as they are typically stored as single-value attributes. As a result, the disk usage of GraphLSS primarily depends on the number of nodes. Although arXiv articles are approximately 50% longer than those in PubMed, the resulting graph size increases by only 15% in nodes and 75% in edges, leading to a 15% increase in disk usage (56 KB per graph). Such an increase is also reflected in the GAT training time (Table 3). In contrast, increasing model complexity from one to two GAT layers extends training time by 32% on PubMed and 40% on arXiv. In order to reduce the disk usage of graph datasets, potential optimizations could involve reducing node counts or strategically limiting the embedding dimensionality (Jang et al., 2024).

Ablation Study We conducted an ablation study on PubMed to assess the contributions of each edge type (Table 4). The results indicate that word-in-sentence edges have the highest impact on GraphLSS performance, as their removal significantly reduces ROUGE scores. This highlights the importance of cross-granularity interactions for effective document representation. Notably, around 80% of node associations are discarded when removing such edges, isolating words and sentences into separate components. Sentence edges are also important, with a comparable effect on ROUGE. However, sentence similarity edges are relatively more influential than sentence order ones due to their lower edge count. In turn, word similarity edges have the least impact, reflecting their low representation in the graph (only 3%; Table 1).

	R-1	R-2	R-L
GraphLSS	51.42	24.32	49.48
(–) Word in Sentence E_{ws}	47.91	21.96	46.02
(–) Sentence Similarity E_{ss}	48.87	22.39	46.68
(–) Sentence Occurrence E_{ns}	48.99	22.41	46.65
(–) Word Similarity E_{ww}	50.84	23.78	48.80

Table 4: Ablation study on PubMed. Results were obtained by removing one specific edge type.

6 Conclusions

We introduced GraphLSS, a heterogeneous graph for long document extractive summarization incorporating lexical, structural, and semantic features. Experiments on PubMed and arXiv highlight the impact of extractive labels due to their inherent imbalance. GraphLSS proves competitive with top-performing graph-based methods and outperforms recent non-graph models by using a greedy labeling strategy and adaptive weights during training. Future work will focus on integrating an abstractive summarization model built upon our extractive results, while also investigating alternative methods to optimize storage and improve scalability.

Limitations

While we showed the impact and potential of GraphLSS for long document extractive summarization, there are some points to keep in mind.

Storing document graphs as a data structure obtained from the original documents (texts) involves significant additional disk usage. Previous strategies create such structures on the fly while training the underlying GNN models, and others opt for storing such graphs on disk to speed up model training. We follow the latter strategy. Therefore, the training time reported does not consider the creation of the underlying graphs.

Furthermore, our proposal was only validated on English datasets. Applying GraphLSS to other languages may yield significantly different results, since pre-trained word and sentence embeddings are required for node initialization and thus, training the heterogeneous GAT model. Analyzing this aspect would be particularly interesting for low-resource languages. Additionally, our experiments focus on scientific papers. Although they cover multiple scientific domains, exploring other kinds of long document, e.g., narrative and legal documents, is encouraged. Also, additional data collections should be analyzed in order to generalize our findings to broader domains.

Ethics Statement

While extractive summaries are less prone to hallucinated content, in some instances, they may be misleading due to missing context (Yang et al., 2017). Another concern is that of possible bias during the content selection. Depending on the graph construction applied, a GAT model may favor certain types of content over others, such as popular sentences and entities with high degrees, as they might receive more attention. Thus, special care must be taken when relying on summaries to make high-stakes decisions, for example in the legal or medical domains.

Summarizing articles often involves extracting information related to trending topics, institutions, people, and other entities. Balancing the delivery of valuable summaries while respecting the privacy of these entities is essential. One strategy to alleviate such concern is anonymization, which ensures that the summary content does not reveal sensitive features. In our study, we conduct all experiments on publicly available scientific articles, and hence have forgone such anonymization.

References

Margarita Bugueño and Marcelo Mendoza. 2020. Learning to combine classifiers outputs with the transformer for text classification. *Intelligent Data Analysis*, 24(S1):15–41.

Margarita Bugueño and Gerard de Melo. 2023. Connecting the dots: What graph-based text representations work best for text classification using graph neural networks? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8943–8960, Singapore. Association for Computational Linguistics.

Sangwoo Cho, Kaiqiang Song, Xiaoyang Wang, Fei Liu, and Dong Yu. 2022. Toward unifying text segmentation and long document summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 106–118, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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 *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Peng Cui and Le Hu. 2021. Sliding selector network with dynamic memory for extractive summarization of long documents. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5881–5891, Online. Association for Computational Linguistics.

Peng Cui, Le Hu, and Yuanchao Liu. 2020. Enhancing extractive text summarization with topic-aware graph neural networks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5360–5371, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Xuan-Dung Doan, Le-Minh Nguyen, and Khac-Hoai Nam Bui. 2022. Multi graph neural network for extractive long document summarization. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5870–5875, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Yunhui Jang, Dongwoo Kim, and Sungsoo Ahn. 2024. Graph generation with k^2-trees. *Preprint*, arXiv:2305.19125.

Ruipeng Jia, Yanan Cao, Hengzhu Tang, Fang Fang, Cong Cao, and Shi Wang. 2020. Neural extractive summarization with hierarchical attentive heterogeneous graph network. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3622–3631, Online. Association for Computational Linguistics.

Florian Le Bronnec, Song Duong, Mathieu Ravaut, Alexandre Allauzen, Nancy Chen, Vincent Guigue, Alberto Lumbreras, Laure Soulier, and Patrick Gallinari. 2024. LOCOST: State-space models for long document abstractive summarization. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1144–1159, St. Julian's, Malta. Association for Computational Linguistics.

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

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. SummaRuNNer: A recurrent neural network based sequence model for extractive summarization of documents. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1).

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, Doha, Qatar. Association for Computational Linguistics.

Tuan-Anh Phan, Ngoc-Dung Ngoc Nguyen, and Khac-Hoai Nam Bui. 2022. HeterGraphLongSum: Heterogeneous graph neural network with passage aggregation for extractive long document summarization. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 6248–6258, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Antoine Tixier, Polykarpos Meladianos, and Michalis Vazirgiannis. 2017. Combining graph degeneracy and submodularity for unsupervised extractive summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 48–58, Copenhagen, Denmark. Association for Computational Linguistics.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *Proceedings of the 2018 International Conference on Learning Representations (ICLR)*.

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

Ting Wang, Chuan Yang, Maoyang Zou, Jiaying Liang, Dong Xiang, Wenjie Yang, Hongyang Wang, and Jia Li. 2024. A study of extractive summarization of long documents incorporating local topic and hierarchical information. *Scientific Reports*, 14(1):10140.

Wen Xiao and Giuseppe Carenini. 2019. Extractive summarization of long documents by combining global and local context. 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 3011–3021, Hong Kong, China. Association for Computational Linguistics.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Discourse-aware neural extractive text summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5021–5031, Online. Association for Computational Linguistics.

Qian Yang, Yong Cheng, Sen Wang, and Gerard de Melo. 2017. HiText: Text reading with dynamic salience marking. In *Proceedings of WWW 2017*, pages 311–319. International World Wide Web Conferences Steering Committee.

Haopeng Zhang, Xiao Liu, and Jiawei Zhang. 2022. HEGEL: Hypergraph transformer for long document summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10167–10176, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Haopeng Zhang, Xiao Liu, and Jiawei Zhang. 2023. Contrastive hierarchical discourse graph for scientific document summarization. In *Proceedings of the 4th Workshop on Computational Approaches to Discourse (CODI 2023)*, pages 37–47, Toronto, Canada. Association for Computational Linguistics.

A Dataset Statistics

We use two publicly available benchmarks for long document summarization, PubMed and arXiv (Cohan et al., 2018). PubMed comprises biomedical scientific papers collected from pubmed.ncbi.nlm.nih.gov, while arXiv covers various scientific domain articles collected from arXiv.org. The statistics of both datasets are presented in Table 5.

	PubMed	arXiv
#Training	115,776	197,650
#Validation	6,584	6,435
#Testing	6,620	6,439
Avg. # Tokens in doc.	2,768	3,913
Avg. # Tokens in summary	205	203
Avg. # Sentences in doc.	89	133
Avg. # Sentences in summary	8	7

Table 5: Datasets statistics.

A.1 Preprocessing Details

As described in Section 4, we removed duplicate and empty documents and instances where the article is shorter than the corresponding summarization. Subsequently, we split the documents via NLTK's sentence tokenizer. However, since the sentence tokenizer splits text based on punctuation, this can often result in non-sensical sentences. For example, the sentence *“Neptune masses can be excluded by our limits determinations (fig.1)"* results in a head sentence S_h =*“Neptune masses can be excluded by our limits determinations (fig."* and a tail sentence S_t =*“1)."*. In such cases, we merged tail sentences with the preceding ones to maintain text coherence.

B Further Experimental Details

Experimental Setup We trained a GAT model (Veličković et al., 2018) with 4 attention heads, varying the number of hidden layers between 1 and 2. We applied Dropout after every GAT layer with a retention probability of 0.7. The final representation is fed into a sigmoid classifier. We initialized word nodes using GloVe Wiki-Gigaword 300-dim. embeddings (Pennington et al., 2014) and pre-trained SBERT (`All-MiniLM-L6-v2`) embeddings for sentence nodes (Reimers and Gurevych, 2019).

All experiments used a batch size of 64 samples and were trained for a maximum of 20 epochs using Adam optimization with an initial learning rate of 10^{-3}. The training was stopped if the validation loss did not improve for 7 consecutive iterations. The objective function of each model was to minimize the binary cross-entropy loss using adaptive class weights, as described in Equation 1. All experiments are based on PyTorch Geometric and conducted on an NVIDIA GeForce RTX 3050. We share our code and graph creation pipeline on `https://github.com/AbouClaude/GraphLSS`.

Baseline Comparison Topic-GraphSum, SSN, and HeterGraphLongSum were excluded from our experiments due to constraints related to code availability and compatibility with our experimental framework. For instance, HeterGraphLongSum is implemented using the DGL library, whereas our experiments are conducted in PyTorch Geometric, leading to technical incompatibilities. In addition to the lack of available code, detailed reproduction steps were missing for such baselines, posing significant challenges. Given these limitations and resource constraints, we report their results as published in the respective papers.

Adaptive Class Weights Figure 2 illustrates how the adaptive class weights evolve across epochs during training. Specifically, we update the weights solely for the relevant class (summary sentences), maintaining static weights for the irrelevant class.

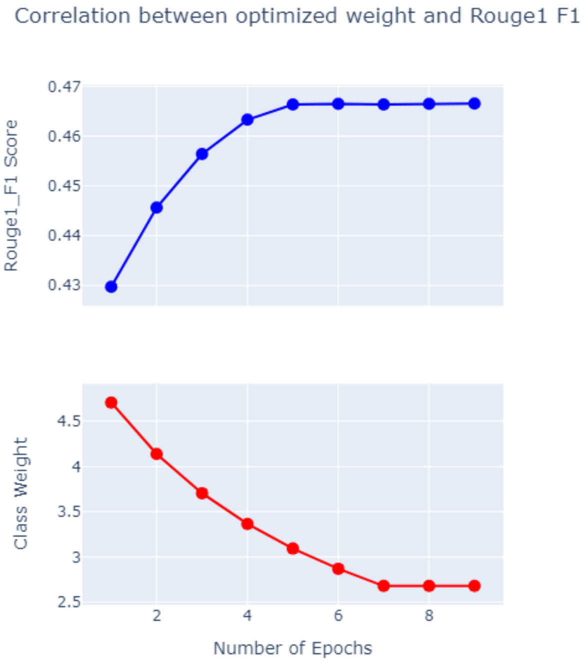

Figure 2: Effect of adaptive class weights on PubMed.

C Libraries Used

The experiments were conducted using the following libraries:

Library	Version
`nltk`	`3.8.1`
`pytorch`	`2.2.1`
`transformers`	`4.38.2`
`rouge`	`1.0.1`
`scikit-learn`	`1.3.0`
`torchmetrics`	`1.2.1`
`torch_geometric`	`2.5.0`

Table 6: Libraries and versions.