Abstractive Summarization Guided by Latent Hierarchical Document Structure

Yifu QIU **Shay B. Cohen**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh, EH8 9AB
Y.QIU-20@sms.ed.ac.uk, scohen@inf.ed.ac.uk

Abstract

Sequential abstractive neural summarizers often do not use the underlying structure in the input article or dependencies between the input sentences. This structure is essential to integrate and consolidate information from different parts of the text. To address this shortcoming, we propose a hierarchy-aware graph neural network (HierGNN) that captures such dependencies through three main steps: 1) learning a hierarchical document structure through a latent structure tree learned by a *sparse* matrix-tree computation; 2) propagating sentence information over this structure using a novel message-passing node propagation mechanism to identify salient information; 3) using graph-level attention to concentrate the decoder on salient information. Experiments confirm HierGNN improves strong sequence models such as BART, with a 0.55 and 0.75 margin in average ROUGE-1/2/L for CNN/DM and XSum. Further human evaluation demonstrates that our model summaries are more relevant and less redundant than the baseline model, into which HierGNN is incorporated. We also find HierGNN synthesizes summaries by fusing multiple source sentences more, rather than compressing a single source sentence, and that it processes long inputs more effectively.[1]

1 Introduction

Sequential neural network architectures in various forms have become the mainstay in abstractive summarization (See et al., 2017; Lewis et al., 2020). However, the quality of machine-produced summaries still lags far behind the quality of human summaries (Huang et al., 2020a; Xie et al., 2021; Cao et al., 2022; Lebanoff et al., 2019). Due to their sequential nature, a challenge with neural summarizers is to capture hierarchical and inter-sentential dependencies inside the input article.

[1]Code for HierGNN is available in https://github.com/yfqiu-nlp/hiergnn

Article Sentences: 1. The town is home to the prestigious leander club, which has trained more than 100 Olympic medal-winning rowers. - *2 sentences are abbreviated here.* 4. The royal mail has painted more than 50 postboxes gold following Team GB's gold medal haul at London 2012. 5. Originally it said it was only painting them in winners home towns, or towns with which they are closely associated. 6. Town mayor Elizabeth Hodgkin said: " We are the home of rowing ... I feel very excited about it." - *5 sentences are abbreviated here.* 12. The Henley-on-Thames postbox was painted on Friday. - *one sentence is abbreviated here.*
Reference Summary: The royal mail has painted a postbox gold in the oxford-shire town of Henley-on-Thames - in recognition of its medal winning rowing club.
BART's Summary: A postbox in Henley-on-Thames has been painted gold as part of the royal mail's " Olympic gold " campaign.
Our HierGNN's Summary: A royal mail postbox in Henley-on-Thames has been painted gold in honour of the town 's Olympic rowing success.

Table 1: Example of an article from XSum with summaries given by human-written reference, BART (Lewis et al., 2020) and our HierGNN equipped with BART. BART's summary fails to capture all information pieces as the reference (as highlighted in various colors), while HierGNN has advantages in combining the information from multiple locations in the source side.

Progress in cognitive science suggests that humans construct and reason over a latent hierarchical structure of document when reading the text (Graesser et al., 1994; Goldman et al., 1999). Such *reasoning behavior* includes uncovering the salient contents and effectively aggregating all related clues spreading across the documents to understand the document. Meanwhile, in the progress of automatic summarization, Lebanoff et al. (2019) found that human editors usually prefer writing a summary by fusing information from multiple article sentences and reorganizing the information in summaries (i.e., sentence fusion), instead of dropping non-essential elements in an original sentence such as prepositional phrases and adjectives (i.e., sentence compression). Different summarization

benchmarks show there are between 60-85% summary sentences that are generated by sentence fusing. All these findings support our interests in mimicing such reasoning behavior with the hierarchical document structure in addition to the sequential neural summarizers.

To fill these gaps, we present a document hierarchy-aware graph neural network (HierGNN), a neural encoder with reasoning functionality that can be effectively incorporated into any sequence-to-sequence (seq2seq) neural summarizer. Our HierGNN first learns a latent hierarchical graph via a sparse variant of the matrix-tree computation (Koo et al., 2007; Liu et al., 2019a). It then formulates the sentence-level reasoning as a graph propagation via a novel message passing mechanism. During decoding, a graph-selection attention mechanism serves as a source sentence selector, hierarchically indicating the attention module which tokens to focus on in the input sentences.

Our empirical experiments with HierGNN incorporated into both pointer-generator networks (See et al., 2017) and BART (Lewis et al., 2020) confirm that HierGNN substantially improves both the non-pretrained and pretrained seq2seq baselines in producing high-quality summaries. Specifically, our best HierGNN-BART achieves an average improvement of 0.55 and 0.75 points in ROUGE-1/2/L on CNN/DM and XSum. Compared with the plain seq2seq model, HierGNN encourages the summarizers to favor sentence fusion more than sentence compression when generating summaries. Modeling the hierarchical document structure via our sparse matrix-tree computation also enables HierGNN to treat long sequences more effectively. Our sparse adaptive variant of the matrix-tree computation also shows a more powerful expressive ability over the original one (Koo et al., 2007; Liu et al., 2019a). We summarize our contributions as follows,

- We present a novel encoder architecture for improving seq2seq summarizers. This architecture captures the hierarchical document structure via an adaptive sparse matrix-tree computation, with a new propagation rule for achieving inter-sentence reasoning.
- We design a graph-selection attention mechanism to fully leverage the learned structural information during decoding in advantages over only using it in encoding.
- Results on CNN/DM and XSum demonstrates the effectiveness of HierGNN in improving the quality of summaries for both non-pretrained and pretrained baselines. In-depth analysis confirms our module prefers more in integrating information from multiple locations in the input article and it is more effective in processing long sequence inputs.

2 Related Work

Neural Abstractive Summarization Rush et al. (2015) first proposed to use a sequence-to-sequence model with an attention mechanism to perform sentence compression. The pointer-generator networks (PGN; See et al. 2017) enhances the attention model with a copying functionality. PGN has also been further extended to create summarization systems by incorporating the topic information (Liu et al., 2019b), document structural information (Song et al., 2018), semantic information (Hardy and Vlachos, 2018), and was improved by replacing the plain LSTM module with the more advanced Transformer model to overcome the difficulty in modeling long sequence input (Pilault et al., 2020; Wang et al., 2021). For the pretrained models, BERTSum (Liu and Lapata, 2019) adopted the BERT encoder for the summarizer, with a randomly initialized decoder. Lewis et al. (2020) presented BART which pre-trains both the underlying encoder and decoder. Dou et al. (2021) investigated "guidance signals" (e.g., keywords, salient sentences) for further boosting the performances.

Graph Neural Approach for Summarization Graph neural networks have demonstrated their ability to capture rich dependencies in documents to be summarized. Wang et al. (2020) use a "heterogeneous graph" with sentence nodes and co-occurring word nodes to capture the sentence dependencies. Jin et al. (2020) use two separate encoders to encode the input sequence with a parsed dependency graph. Cui et al. (2020) use a bipartite graph with a topic model to better capture the inter-sentence relationships. Kwon et al. (2021) capture both intra- and inter-sentence relationships via a nested tree structure. Zhu et al. (2021) use entity-relation information from the knowledge graph to increase the factual consistency in summaries.

Our approach is related to the structural attention model (Balachandran et al., 2021; Liu et al., 2019a), but differs in two major ways: (i) we introduce an adaptive sparse matrix-tree construction to learn a latent hierarchical graph and a novel propagation rule; (ii) we investigate to use the structure

information both with the encoder and the decoder, and not just the encoder. These shows to be more effective for unsupervised learning of the latent hierarchical structure while can defeat the approach that leverages external graph constructor.

3 Hierarchy-aware Graph Neural Encoder

HierGNN learns the document structure in an end-to-end fashion without any direct structure supervision, and does not need an external parser to construct the structure, unlike previous work (Balachandran et al., 2021; Huang et al., 2020b; Wang et al., 2020; Cardenas et al., 2022). In addition, it empirically improves over supervised graph construction, which has been a challenge (Balachandran et al., 2021).

Sequential summarizers encode an N-token article, $X = (x_1, \cdots, x_N)$ as d-dimensional latent vectors using an encoding function $\mathbf{h}_{enc}(x_t) \in \mathbb{R}^d$ and then decodes them into the target summary Y. (We denote by $\mathbf{h}_{enc}(X)$ the sequence of x_t encodings for $t \leq N$.) Our model includes four modules in addition to this architecture: 1) a sparse matrix-tree computation for inferring the document hierarchical structure, ii) a novel message-passing layer to identify inter-sentence dependencies, iii) a reasoning fusion layer aggregating the outputs of the message-passing module; and vi) a graph-selection attention module to leverage the encoded structural information.

3.1 Learning the Latent Hierarchical Structure

We first introduce our latent structure learning algorithm that makes use of the matrix-tree theorem (Tutte, 1986; Koo et al., 2007).

Latent Document Hierarchical Graph. We represent the document as a complete weighted graph, with each node being a sentence. The edge weights are defined as the marginal probability of a directional dependency between two sentences. In addition, each sentence node has an extra probability value, the "root probability" which indicates the *hierarchical role* of the sentence, such as "the lead," "most important facts," or "other information." The roles are defined based on the "inverted pyramid" model for news articles (Pottker, 2003; Ytreberg, 2001). Intuitively, a sentence with a high root probability (i.e., high hierarchical position) conveys more general information; namely, it is a *connector*, while a sentence with a lower root probability (*information node*) carries details supporting its higher connectors. The underlying graph structure is latent and not fixed, summed out in our overall probability model using the matrix-tree theorem.

Sparse Matrix-Tree Computation. For an article with M sentence, we start from the sentence embeddings as the node initialization $H^{(0)} = [\mathbf{s}_1, ..., \mathbf{s}_i, ..., \mathbf{s}_M]$. We then use two independent non-linear transformations to obtain a pair of *parent* and *child* representation for each sentence,

$$\mathbf{s}_i^{(p)} = \sigma(W_p \mathbf{s}_i + b_p); \ \mathbf{s}_i^{(c)} = \sigma(W_c \mathbf{s}_i + b_c),$$

where W_p, W_c, b_p, b_c are parameters, σ is the ReLU activation function (Dahl et al., 2013).

The standard use of the matrix-tree theorem (Tutte, 1986) computation (MTC; Smith and Smith 2007; Koo et al. 2007) includes the exponential function to calculate a matrix with positive values $F \in \mathbb{R}^{M \times M}$ with each element f_{ij} representing the weight of the directional edge from a node s_i to s_j; and a positive vector of root scores $\mathbf{f}^{(root)} \in \mathbb{R}^M$. However, having a dense matrix degrades our graph reasoning module by including irrelevant information from redundant M sentence nodes. Inspired by the work about sparse self-attention (Zhang et al., 2021; Correia et al., 2019), we introduce an adaptive solution to inject sparsity into MTC. We replace the exponential scoring function with the ReLU function ($\mathrm{ReLU}(x \in \mathbb{R}) = \max\{x, 0\}$ and similarly coordinate-wise when x is a vector) and calculate the root $f_i^{(root)}$ and edge scores f_{ij} by a fully-connected layer and a bi-linear attention layer, respectively,

$$f_i^{(root)} = \textsc{ReLU}(W_r \mathbf{s}_i^{(p)} + b_r) + \varepsilon,$$
$$f_{ij} = \textsc{ReLU}({\mathbf{s}_i^{(p)}}^\top W_{bi} \mathbf{s}_j^{(c)}) + \varepsilon,$$

where W_{bi}, W_r, b_r are learnable. We use $\varepsilon = 10^{-6}$ to avoid matrix non-invertibility. Compared to the exponential function, ReLU relaxes F and $\mathbf{f}^{(root)}$ to be non-negative, thus being capable of assigning zero probability and pruning the dependency edges and roots simultaneously. We finally plug in these quantities to the standard MTC (Koo et al., 2007; Tutte, 1986) and marginalize the edge and root probabilities as the adjacency matrix $A(i, j) = P(z_{ij} = 1)$ and root probability p_i^r representing the hierarchical role (i.e., the likelihood to be a connector) of each sentence.

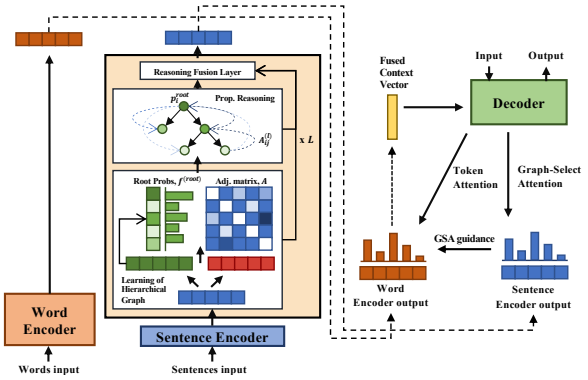

Figure 1: Architecture for the sequence-to-sequence model with HierGNN reasoning encoder.

3.2 Reasoning by Hierarchy-aware Message Passing

We present a novel message passing mechanism over the learned hierarchical graph. This mechanism realizes the inter-sentence reasoning where connectors can aggregate information from their related information nodes while propagating the information to others. For the i-th sentence node, the edge marginal controls the aggregation from its K information nodes; and the root probability controls the neighbouring information is combined as i-th node's update $\mathbf{u}^{(l)}$ in l-th reasoning layer,

$$\mathbf{u}_i^{(l)} = (1-p_i^r)\mathcal{F}_r(\mathbf{s}_i^{(l)}) + (p_i^r)\sum_{k=1}^{K} A_{ik}\mathcal{F}_n(\mathbf{s}_k^{(l)}),$$

where $\mathcal{F}_r$ and $\mathcal{F}_n$ are parametric functions. Intuitively, if a sentence is a *connector*, it should have strong connectivity with the related *information nodes*, and aggregate more details. Each information node learns to either keep the uniqueness of its information or fuse the information from the connectors. To filter out the unnecessary information, we adopt a gated mechanism as the information gatekeeper in the node update,

$$\mathbf{g}_i^{(l)} = \sigma(\mathcal{F}_g([\mathbf{u}_i^{(l)}; \mathbf{h}_i^{(l)}])),$$
$$\mathbf{h}_i^{(l+1)} = \text{LN}(\mathbf{g}_i^{(l)} \odot \phi(\mathbf{u}_i^{(l)}) + (\mathbf{1} - \mathbf{g}_i^{(l)}) \odot \mathbf{h}_i^{(l)}),$$

where $\mathcal{F}_g$ is a parametric function and $\odot$ is the element-wise dot product. We use layer normalization (LN) to stabilize the output for the update function. The function σ is the sigmoid function, and ϕ can be any non-linear function.

3.3 Reasoning Fusion Layer

We construct *reasoning chains* that consist of L hops by stacking L HierGNN blocks together. To handle cases where fewer than L hops are needed, we add a fusion layer to aggregate the output from each reasoning hop to produce the final output of HierGNN. A residual connection is also introduced to pass the node initialization directly to the output,

$$\mathbf{h}_i^{(G)} = (W_g[\mathbf{h}_i^{(1)}, ..., \mathbf{h}_i^{(L)}] + b_g) + \mathbf{h}_i^{(0)},$$

where W_g, b_g are learnabale parameters. We use two approaches for layer use: (a) *Layer-Shared Reasoning (LSR)*: we construct a shared reasoning graph first, followed by L message passing layers for reasoning; (b) *Layer-Independent Reasoning (LIR)*: we learn the layer-wise latent hierarchical graphs independently, where each message passing layer uses its own graph.

3.4 Graph-selection Attention Mechanism

In addition to token-level decoding attention, we propose a *graph-selection attention mechanism* (GSA) to inform the decoder with learned hierarchical information, while realizing the sentence-level content selection. In each decoding step t, our decoder first obtains a graph context vector, $\mathbf{c}_G^t$, which entails the global information of the latent hierarchical graph. We first compute the graph-level attention distribution $\mathbf{a}_G^t$ by,

$$e_{v_i}^t = \text{ATTN}^{(G)}(\mathbf{h}^{(L)}, \mathbf{z}_t),$$
$$\mathbf{a}_G^t = \text{SOFTMAX}(\mathbf{e}^t),$$

where $\text{ATTN}^{(G)}$ is a graph attention function. The vectors $\mathbf{h}_i^{(L)} \in \mathbb{R}^d, \mathbf{z}_t \in \mathbb{R}^d$ are the L-th layer node embeddings for sentence i and decoding state at time t, respectively. The graph context vector $\mathbf{c}_G^t \in \mathbb{R}^d$ is finally obtained by summing all $\mathbf{h}_i^{(L)}$ weighted by $\mathbf{a}_G^t$. The value of $\mathbf{c}_G^t$ is used as an additional input for computing token-level attention,

$$e_i^t = \text{ATTN}^{(T)}(\mathbf{h}_{enc}(X), \mathbf{z}_t, \mathbf{c}_G^t),$$
$$\mathbf{a}_T^t = \text{SOFTMAX}(\mathbf{e}^t),$$

where $\text{ATTN}^{(T)}$ is a token-level attention function (Luong et al., 2015; Vaswani et al., 2017). Again, the token-attentional context vector $\mathbf{c}_f^t$ is computed by summing the encoder outputs weighted by $\mathbf{a}_T^t$. The final context vector $\mathbf{c}_f^t$ is fused from the graph $\mathbf{c}_G^t$ and token context vectors $\mathbf{c}_T^t$ with a parametric function g_f, $\mathbf{c}_f^t = g_f(\mathbf{c}_G^t, \mathbf{c}_T^t)$.

4 Experimental Setting

Benchmarks. We evaluate our model on two common document summarization benchmarks. The

Non-pretrained	R-1	R-2	R-L	BS
LEAD-3	40.34	17.70	36.57	-
PGN	39.53	17.28	36.38	-
StructSum ES	39.63	16.98	36.72	-
StructSum LS	39.52	16.94	36.71	-
StructSum (LS + ES)	39.62	17.00	**36.95**	21.70
PGN - Ours	39.07	16.97	35.87	23.74
HierGNN-PGN (LSR)	**39.87**	**17.77**	36.85	**25.64**
HierGNN-PGN (LIR)	39.34	17.39	36.44	25.26
Pretrained	**R-1**	**R-2**	**R-L**	**BS**
BERTSUMABS	41.72	19.39	38.76	29.05
BERTSUMEXTABS	42.13	19.60	39.18	28.72
T5-Large	42.50	20.68	39.75	-
BART	44.16	21.28	40.90	-
Hie-BART	44.35	21.37	41.05	-
HAT-BART	44.48	21.31	41.52	-
BART - Ours	44.62	21.49	41.34	33.98
BART + SentTrans.	44.44	21.44	41.27	33.90
HierGNN-BART (LSR)	44.93	21.7	41.71	34.43
HierGNN-BART (LIR)	**45.04**	**21.82**	**41.82**	**34.59**

Table 2: Automatic evaluation results in ROUGE scores, BertScore (BS) on CNN/DM. The top and bottom blocks show the comparison for non-pre-training and pre-training models separately. We use **bold** to mark the best abstractive model.

Non-pretrained	R-1	R-2	R-L	BS
LEAD-3	16.30	1.60	11.95	-
Seq2Seq (LSTM)	28.42	8.77	22.48	-
Pointer-Generator	29.70	9.21	23.24	23.16
PGN + Coverage	28.10	8.02	21.72	-
HierGNN-PGN (LSR)	30.14	10.21	**24.32**	27.24
HierGNN-PGN (LIR)	**30.24**	**10.43**	24.20	**27.36**
Pretrained	**R-1**	**R-2**	**R-L**	**BS**
BERTSUMABS	38.76	16.33	31.15	37.60
BERTSUMEXTABS	38.81	16.50	31.27	38.14
T5 (Large)	40.9	17.3	33.0	-
BART	45.14	22.27	37.25	-
HAT-BART	**45.92**	**22.79**	**37.84**	-
BART - Ours	44.97	21.68	36.47	52.89
BART + SentTrans.	45.12	21.62	36.46	52.95
HierGNN-BART (LSR)	45.19	21.71	36.59	52.94
HierGNN-BART (LIR)	45.39	21.89	36.81	**53.15**

Table 3: Automatic evaluation results in ROUGE scores, BertScore (BS) on XSum. All of our HierGNN-PGN models are trained without a coverage mechanism. We use **bold** for the best model.

first is the CNN/Daily Mail dataset in the news domain, with an average input of 45.7 sentences and 766.1 words, and a reference with an average length of 3.59 sentences and 58.2 words. We use the non-anonymized version of See et al. (2017), which has 287,084/13,367/11,490 instances for training, validation and testing. The second dataset we use is XSum, a more abstractive benchmark consisting of one-sentence human-written summaries for BBC news. The average lengths for input and reference are 23.26 sentences with 430.2 words and 1 sentence with 23.3 words, respectively. We follow the standard split of Narayan et al. (2018) for training, validation and testing (203,028/11,273/11,332).

Implementations. We experiment with the non-pretrained Pointer-Generator Network (See et al., 2017) and the pretrained BART (Lewis et al., 2020). The implementation details are in the Appendix A.

Baselines. We compare HierGNN with three types of baselines: 1) the base models for developing HierGNN; and 2) several strong non-pretrained and pretrained baselines; 3) previous summarizers boosted with the hierarchical information.

We compare HierGNN-PGN with the non-pretrained baselines. We first include the **LEAD-3** (Nallapati et al., 2017) that simply selects the top three sentences in the article as the summary. **StructSum** (Balachandran et al., 2021) is a PGN-based model, which incorporates structure information by an explicit attention mechanism (ES Attn) on a coreference graph and implicit attention mechanism (IS Attn) on an end-to-end learned document structure. StructSum ES+IS Attn uses both implicit and explicit structures.

We compare HierGNN-PGN with the pretrained baselines. **BERTSumAbs** and **BERTSumExtAbs** are two abstractive models by Liu and Lapata (2019) based on the BERT encoder. We also incorporate a strong multitask sequence generation model, **T5-Large**. **Hie-BART** (Akiyama et al., 2021) enhances BART by jointly modeling the sentence and token-level information in the self-attention layer. **HAT-BART** (Rohde et al., 2021) appends a sentential Transformer block on top of the BART's encoder to model the sentence-level dependencies. We also use a baseline, **BART+SentTrans.**, replacing our MTC block with a Transformer block. This baseline uses a comparable number of parameters to our HierGNN. We aim to verify the advantage of modeling the document's hierarchical information by MTC over just increasing the model size.

5 Results

Automatic Evaluation. We evaluate the quality of summaries through ROUGE F-1 scores (Lin and Och, 2004) by counting the unigram (R-1), bigram (R-2) and longest common subsequence

Model	Rel.	Inf.	Red.	Overall
BERTSUMABS	*-0.43	*-0.33	-0.11	*-0.29
T5	0.08	-0.09	0.05	0.01
BART	0.15	**0.24**	-0.04	0.12
HierGNN-BART	**0.20**	0.19	**0.09**	**0.16**

Table 4: Results for the human evaluation based on i) Relevance (Rel.), ii) Informativeness (Inf.), and iii) Redundancy (Red.). * indicates statistically significant improvements over the baselines with our model (*: by pair-wise t-test with $p < 0.05$, corrected using Benjamini–Hochberg method to control the False Discovery Rate (Benjamini and Hochberg, 1995) for multiple comparison). We **bold** the best results in each criteria and the overall evaluation. Detailed results are given in Appendix C.

	R-1	R-2	R-L	BS
Full Model	30.24	10.43	24.20	27.36
w/o HierGNN Module	-0.54	-1.22	-0.96	-4.20
w/o Graph-select (GSA)	-0.41	-0.41	-0.17	-0.27
w/o Sparse MTC	-0.14	-0.25	+0.05	-0.41
w/o Graph Fusion	-0.94	-0.81	-0.77	-1.39

Table 5: Ablation study of each modules in our HierGNN-PGN (LIR) model on XSum.

(R-L) overlaps. To avoid the use of pure lexical overlap evaluation (Huang et al., 2020a), we also use BERTScore (Zhang et al., 2019).

We summarize the results for non-pretrained and pretrained models on CNN/DM and XSum in the upper and bottom block of Table 2 and Table 3, respectively. Our HierGNN module improves the performance over both the PGN and BART for both CNN/DM and XSum, demonstrating the effectiveness of our reasoning encoder for both non-pretrained and pretrained summarizers. Secondly, the best model of HierGNN-PGN achieves higher scores than StructSum ES and ES+IS that explicitly construct the document-level graph representation using an external parser in pre-processing. This indicates our learned hierarchical structure is effective and beneficial for downstream summarization without any supervision. HierGNN-BART also outperforms Hie-BART, HAT-BART and BART+SentTrans., which indicates that the MTC encoder's inductive bias is effective in modeling useful structure.

Human Evaluations. We also invited human referees from Amazon Mechanical Turk to assess our model and additional three pure abstractive baselines including BERTSUMABS, T5-Large, BART on CNN/DM testing set. Our assessment focuses on three criteria: i) Relevance (*Whether the conveyed information in the candidate summary is relevant to the article*?), ii) Informativeness (*How accurate and faithful information does the candidate summary convey*?), and iii) Redundancy (*Whether the sentences in each candidate summary are non-redundant with each other*?). The detailed settings for human evaluation are presented in Appendix B. We ask the referees to choose the best and worst summaries from the four candidates for each criterion. The overall scores in Table 4 are computed as the percentage of times a summary was chosen as the best minus the times it was selected as the worst, which ranges from -1 (worst) to 1 (best). The results show that our HierGNN-BART achieves the overall best performance. Moreover, though BART has a slightly better informativeness score, HierGNN-BART produces better summaries in terms of Relevance and Redundancy.

Model	Coverage (↗)	Copy Length (↘)
Reference	**20.27** %	**5.10**
Pointer-Generator	11.78 %	18.82
Ours w/o Graph Select Attn.	13.74 %	18.88
Ours $w/$ Graph Select Attn.	**15.22** %	**16.80**

Table 6: Results of average copying length of sequences and coverage of the source sentences for the CNN/DM datasets. Arrows (↗ or ↘) indicate that larger or lower scores are better, respectively.

Ablations. We conduct an ablation study of the HierGNN encoder, graph-selection attention, sparse MTC and graph fusion layer. The ablation is done on our HierGNN-PGN LIR model trained on XSum. The ablation in HierGNN reasoning module significantly degrades the model, which suggests the positive contribution of the functionality in across-sentence reasoning. The scores without GSA also confirm the guidance of graph-level information is beneficial. By removing the graph fusion layer, we again observe the performance decreases, which proves the benefits of fusing the neighbor feature from multiple hopping distances. Finally, the results also confirm the superiority of the sparse MTC over the dense MTC for learning effective hierarchical structure for summarization.

6 Discussion

Informativeness. We use two metrics in (See et al., 2017) to evaluate the informativeness in Table 6. The coverage rate measures how much informa-

	R-1	R-2	BS
BART	49.41	21.70	19.12
HierGNN-BART	**49.62**	**21.74**	**20.32**

Table 7: Summarization performance on PubMed. We test BART and HierGNN-BART with the same hyper-parameters settings.

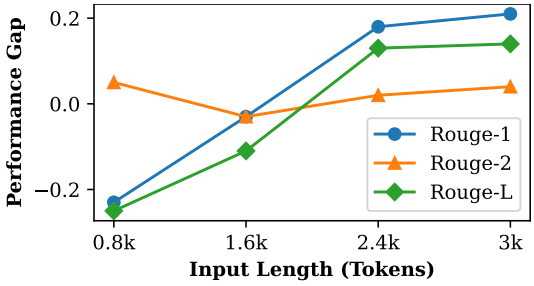

Figure 2: Performance gap on PubMed between HierGNN-BART with BART when summarizing articles truncated at different lengths. The gap between HierGNN and BART consistently increase with the longer input sequences.

tion in the source article is covered by the summary, while the average copy length indicates to what extent that summarizer directly copies the source sequence as its output. The higher coverage rate achieved by our HierGNN indicates that it can produce summaries with much richer information in the source article. Secondly, previous works (Balachandran et al., 2021) find that PGN tends to skew towards copying content from the source sequence thus degenerating as an extractive model, particularly in a more extractive dataset like CNN/DM. As a result, the model might fails to stop copying source at an ideal length. We find that graph-selection attention reduce the average copy length considerably, meaning that it can inform the decoder to stop copying by leveraging the learned structural information in encoder and reduce the reliance on PGN's copying functionality (See et al., 2017). We show an example for the graph-selection attention outcome in Appendix D.

Compression or Fusion? To assess whether sentence fusion happens often, we quantify the ratio of sentence compression and sentence fusion that the model uses to generate summaries (Table 8) (Lebanoff et al., 2019). In comparison to BART, HierGNN reduces the proportion of leveraging sentence compression to synthesize summary sentences in both CNN/DM and XSum. Furthermore, we note that summarization models tend to

CNN/DM	Comp.	2-hop	3-hop	4-hop
Reference	63.03	32.08	4.59	0.31
BART	79.52	17.81	2.43	0.24
HierGNN-BART	78.13(↓)	19.29(↑)	2.36(↓)	0.21(↓)
XSum	**Comp.**	**2-hop**	**3-hop**	**4-hop**
Reference	34.87	42.50	18.79	3.83
BART	28.47	42.51	23.05	5.98
HierGNN-BART	27.27(↓)	42.53(↑)	24.31(↑)	5.89(↓)

Table 8: Percentages of summary sentences are synthesized by compression (information is extracted from a single source sentence) and fusion (information is combined from two or more source sentences). We use ↓ and ↑ to mark the changes between BART and HierGNN.

adopt sentence compression more than in human-written references for CNN/DM, while more sentence fusion is used for XSum. This observation reveals that mechanism for producing summaries is different than that humans use. Human editors can flexibly switch between compression and fusion; the summarization model may simply adopt one of them to produce output.

Effectiveness for Longer Sequence. The performance of sequence-to-sequence models decays as the length of the input sequence increases (Liu et al., 2018), due to the lack of capturing the long-range dependencies. We hypothesize that HierGNN has better capability in capturing the dependencies via its learned document's hierarchical structure, thus enhancing the performance for long-sequence inputs. To verify this, we further conduct experiments on PubMed (Cohan et al., 2018), a long-document summarization dataset with scientific articles in the medical domain.

We summarize the performance in Table 7. We notice that HierGNN improves BART with a large margin. We further evaluate HierGNN's advantages over BART with respect to inputs with various lengths. As shown in Figure 2, when the input length is higher than 1.6K tokens, HierGNN has a positive advantage over BART. As the input length increases, the advantage of HierGNN becomes larger consistently.

Sparse MTC or Dense MTC? We also study the expressive ability of our adaptive sparse variant of the matrix tree computation. We design two quantitative metrics: 1) *Intra-layer diversity* measures the diversity for the marginal distributions of roots and edges in each MTC layer, which is calculated by the range of the probability distribu-

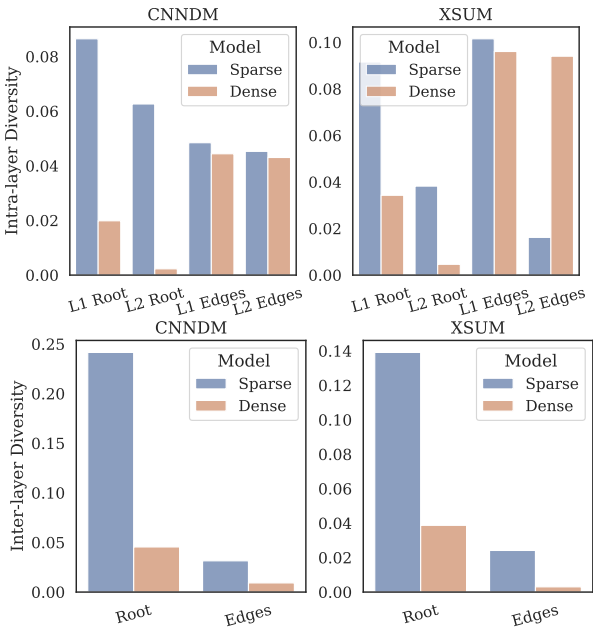

Figure 3: Layer-wise intra-layer diversity (top) and inter-layer diversity (bottom) for BART with 2-layer HierGNN with Sparse and Dense MTC.

tion; 2) *Inter-layer diversity* measures the diversity for the marginal distributions of roots and edges between MTC layers, which is calculated by the average Jensen-Shannon (JS) Divergence between the marginal distributions of roots and edges in different layers (Zhang et al., 2021; Correia et al., 2019). We compare both intra-layer and inter-layer diversity for our adaptively sparse MTC and the original dense MTC (Koo et al., 2007; Liu et al., 2019a; Balachandran et al., 2021).

Figure 3 shows that our sparse variant of MTC has a higher diversity in both intra- (Top) and inter-layer (Bottom) metrics for CNN/DM and XSum, indicating that our sparse MTC has a more powerful expressive ability than dense MTC. We find that the sparsity of HierGNN is very different across layers and datasets: 1) 99.66% of HierGNN's predictions for XSum instances have at least one element that is sparsified to zero, while this proportion is 24.22% for CNN/DM. 2) Almost all the sparsified elements in HierGNN's predictions for XSum are edges, while roots for CNN/DM. 3) We find that 90.32% of elements of edge distribution in the second MTC layer are sparsified in XSum, but no any sparsified element in the first layer. In CNN/DM, the proportion of sparsified elements in the first and second layer are almost identical. These observations reveal that sparse MTC can adaptively choose whether sparse out elements in root or edge distributions, thus boosting the richness of the structural information represented by MTC.

Top-3 Sentences with Highest Root Probabilities (Our Sparse MTC):
8th Sent. (9.77%): A lunar eclipse happens when the sun, Earth and moon form a straight line in space, with the Earth smack in the middle.
6th Sent. (9.40%): The sun shines on the Earth and creates a shadow.
10th Sent. (7.79%): Parts of South America, India, China and Russia also will be able to see the eclipse, but it won't be visible in Greenland, Iceland, Europe, Africa or the Middle East.

Top-3 Sentences with Lowest Root Probabilities (Our Sparse MTC):
20th Sent. (Sparsified): Share your photos with CNN iReport.
18th Sent. (Sparsified): If you want to learn more about the eclipse, NASA astronomer Mitzi Adams will take questions on Twitter NASA_Marshall.
19th Sent. (0.02%): Did you see the total lunar eclipse?

Reference:
The total eclipse will only last 4 minutes and 43 seconds. People west of the Mississippi River will have the best view. Parts of South America, India, China and Russia also will see the eclipse.

Ours:
A total lunar eclipse started at 3:16 a.m. Pacific Daylight Time. People west of the Mississippi River will have the best view. Parts of South America, India, China and Russia also will be able to see the eclipse. The total eclipse will only last four minutes and 43 seconds.

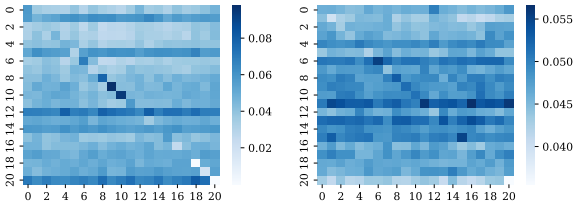

Figure 4: Top: the top-3 sentences with highest/lowest root probabilities, reference and summaries in article 23 in CNN/DM testing split. We underline the relevant contents; Bottom: visualizations for our sparse (Left) and the dense (Right) MTC layer for HierGNN-BART.

We finally show a qualitative case with three sentences per article, having the highest or lowest root probabilities (See the Figure 4), and the heatmap visualization of the learned hierarchical structures from sparse and dense MTC. We observe that the highest-probability root sentences tend to be summary-worthy while also scattering in different positions of the article, and the lowest probability is irrelevant. The structure learned by Sparse MTC tends to be more diverse and can successfully sparsify out the sentence nodes with irrelevant contents, e.g., 18th and 20th sentence.

7 Conclusion

We propose HierGNN that can be used in tandem with existing generation models. The module learns the document hierarchical structure while being able to integrate information from different parts of the text as a form of reasoning. Our experiments verify that HierGNN is effective in improving the plain sequential summarization models.

Limitations

The inductive bias of our HierGNN model has an assumption that the source article follows the type of hierarchical structure (e.g., the inverted pyramid scheme). This may pose limitations in the generalization of our model to other categories of input documents with no or a weak hierarchical structure. Future work includes understanding the limitations of HierGNN in different input domains (e.g., conversation summarization). Additionally, as other large-scale pretrained neural summarizers, our approach with an additional HierGNN encoder increases the model complexity. To train our BART-based system, GPUs with at least 32GB of memory are required. Future work may focus on distilling the large HierGNN model into a much smaller size while retaining the original performance.

Ethical and Other Considerations

Human evaluations. Human workers were informed of the intended use of the provided assessments of summary quality and complied with the terms and conditions of the experiment, as specified by Amazon Mechanical Turk[2]. In regards to payment, workers were compensated fairly with the wage of £9 hourly (higher than the maximum minimum wage in the United Kingdom[3]), i.e. £4.50 per HIT at 2 HITs per hour.

Computing time. We first report the computing time for our most computationally intense HierGNN-BART (471 million parameters) using NVIDIA Tesla A100 with 40G RAM: with CNN/DM, the training takes around 81 GPU hours, and the inference takes 9.39 GPU hours. With XSum, the training takes around 32 GPU hours, and the inference takes 4.41 GPU hours.

Additionally, training of HierGNN-PGN (32 million parameters) on CNN/DM takes 0.79 seconds per iteration using 1 NVIDIA V100 GPU card with 16GB. We estimate the inference time is 4.02 documents per second.

Acknowledgements

We thank the anonymous reviewers for their helpful comments. We also appreciate the feedback received from other members of the Cohort, specifically, Zheng Zhao and Marcio Fonseca for their valuable discussions and comments. The human evaluation was funded by a grant from the Scottish Informatics and Computer Science Alliance. We thank the EPCC Cirrus (Edinburgh) and the Baskerville services (Birmingham) for providing valuable computational resources for this work.

[2] https://www.mturk.com

[3] https://www.gov.uk/national-minimum-wage-rates

References

Kazuki Akiyama, Akihiro Tamura, and Takashi Ninomiya. 2021. Hie-BART: Document summarization with hierarchical BART. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 159–165, Online. Association for Computational Linguistics.

Vidhisha Balachandran, Artidoro Pagnoni, Jay Yoon Lee, Dheeraj Rajagopal, Jaime Carbonell, and Yulia Tsvetkov. 2021. StructSum: Summarization via structured representations. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2575–2585, Online. Association for Computational Linguistics.

Yoav Benjamini and Yosef Hochberg. 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300.

Meng Cao, Yue Dong, and Jackie Cheung. 2022. Hallucinated but factual! inspecting the factuality of hallucinations in abstractive summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3340–3354, Dublin, Ireland. Association for Computational Linguistics.

Ronald Cardenas, Matthias Galle, and Shay B Cohen. 2022. On the trade-off between redundancy and local coherence in summarization. *arXiv preprint arXiv:2205.10192*.

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.

Gonçalo M Correia, Vlad Niculae, and André FT Martins. 2019. Adaptively sparse transformers. 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 2174–2184.

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.

George E Dahl, Tara N Sainath, and Geoffrey E Hinton. 2013. Improving deep neural networks for lvcsr using rectified linear units and dropout. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 8609–8613. IEEE.

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. 2021. GSum: A general framework for guided neural abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4830–4842, Online. Association for Computational Linguistics.

Susan R Goldman, Arthur C Graesser, and Paul van den Broek. 1999. *Narrative comprehension, causality, and coherence: Essays in honor of Tom Trabasso*. Routledge.

Arthur C Graesser, Murray Singer, and Tom Trabasso. 1994. Constructing inferences during narrative text comprehension. *Psychological review*, 101(3):371.

Hardy Hardy and Andreas Vlachos. 2018. Guided neural language generation for abstractive summarization using Abstract Meaning Representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 768–773, Brussels, Belgium. Association for Computational Linguistics.

Dandan Huang, Leyang Cui, Sen Yang, Guangsheng Bao, Kun Wang, Jun Xie, and Yue Zhang. 2020a. What have we achieved on text summarization? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 446–469.

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

Hanqi Jin, Tianming Wang, and Xiaojun Wan. 2020. Semsum: Semantic dependency guided neural abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8026–8033.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Terry Koo, Amir Globerson, Xavier Carreras Pérez, and Michael Collins. 2007. Structured prediction models via the matrix-tree theorem. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 141–150.

Jingun Kwon, Naoki Kobayashi, Hidetaka Kamigaito, and Manabu Okumura. 2021. Considering nested tree structure in sentence extractive summarization with pre-trained transformer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4039–4044, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Logan Lebanoff, Kaiqiang Song, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019. Scoring sentence singletons and pairs for abstractive summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2175–2189.

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 *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Chin-Yew Lin and Franz Josef Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 605–612, Barcelona, Spain.

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

Yang Liu and Mirella Lapata. 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.

Yang Liu, Ivan Titov, and Mirella Lapata. 2019a. 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, Minneapolis, Minnesota. Association for Computational Linguistics.

Zhengyuan Liu, Angela Ng, Sheldon Lee, Ai Ti Aw, and Nancy F. Chen. 2019b. Topic-aware pointer-generator networks for summarizing spoken conversations. In *2019 IEEE Automatic Speech Recogni-*

tion and Understanding Workshop (ASRU), pages 814–821.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. 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. In *Thirty-First AAAI Conference on Artificial Intelligence*.

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

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

Horst Pottker. 2003. News and its communicative quality: the inverted pyramid—when and why did it appear? *Journalism Studies*, 4(4):501–511.

Tobias Rohde, Xiaoxia Wu, and Yinhan Liu. 2021. Hierarchical learning for generation with long source sequences. *arXiv preprint arXiv:2104.07545*.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

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

David A Smith and Noah A Smith. 2007. Probabilistic models of nonprojective dependency trees. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 132–140.

Kaiqiang Song, Lin Zhao, and Fei Liu. 2018. Structure-infused copy mechanisms for abstractive summarization. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1717–1729.

Simeng Sun, Ori Shapira, Ido Dagan, and Ani Nenkova. 2019. How to compare summarizers without target length? pitfalls, solutions and re-examination of the neural summarization literature. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 21–29, Minneapolis, Minnesota. Association for Computational Linguistics.

W. T. Tutte. 1986. Graph theory, by w. t. tutte, encyclopedia of mathematics and its applications, volume 21, addison-wesley publishing company, menlo park, ca., 1984, 333 pp. price: 45.00. *Networks*, 16:107–108.

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 *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010.

Danqing Wang, Pengfei Liu, Yining Zheng, Xipeng Qiu, and Xuan-Jing 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.

Haonan Wang, Yang Gao, Yu Bai, Mirella Lapata, and Heyan Huang. 2021. Exploring explainable selection to control abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13933–13941.

Wenhao Wu, Wei Li, Xinyan Xiao, Jiachen Liu, Ziqiang Cao, Sujian Li, Hua Wu, and Haifeng Wang. 2021. BASS: Boosting abstractive summarization with unified semantic graph. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6052–6067, Online. Association for Computational Linguistics.

Yuexiang Xie, Fei Sun, Yang Deng, Yaliang Li, and Bolin Ding. 2021. Factual consistency evaluation for text summarization via counterfactual estimation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 100–110, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Espen Ytreberg. 2001. Moving out of the inverted pyramid: narratives and descriptions in television news. *Journalism Studies*, 2(3):357–371.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2021. Sparse attention with linear units. *arXiv preprint arXiv:2104.07012*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. 2015. A c-lstm neural network for text classification. *arXiv preprint arXiv:1511.08630*.

Chenguang Zhu, William Hinthorn, Ruochen Xu, Qingkai Zeng, Michael Zeng, Xuedong Huang, and Meng Jiang. 2021. Enhancing factual consistency of abstractive summarization. *North American Chapter of the Association for Computational Linguistics (NAACL) 2021*.

A Implementation Details

HierGNN-PGN is developed based on the Pointer-Generator Network (See et al., 2017)[4] . To obtain the sentence representations, we use a CNN-LSTM encoder to capture both the n-gram features and sequential features (Kim, 2014; Zhou et al., 2015). The CNN's filter windows sizes are set to be $\{1, 2, 3, 4, 5, 7, 9\}$ with 50 feature maps each. We set the dimension of the representations to be 512. The number of reasoning layers L is set to 3 after a development set search in $\{1, 2, 3, 5, 10\}$. Other settings follow the best hyperparameters for CNN/DM as in (See et al., 2017), and we use 60K iterations to train the coverage mechanism. For XSum, we discard the coverage training due to its redundancy for extreme summarization (Narayan et al., 2018), and we use a beam of size 6. We search the best model by the validation ROUGE scores on both datasets with one search trial per hyperparameter.

#Layer	Val. PPL (↘)	R-1 (↗)	R-2 (↗)	R-L (↗)
1	8.61	30.06	10.09	24.23
2	8.58	29.94	10.00	24.13
3	**8.51**	**30.24**	**10.43**	24.20
5	8.54	30.14	10.23	**24.32**
10	8.61	29.99	9.93	24.13

Table 9: Performance of HierGNN-PGN (LIR) on XSum with respect to the number of reasoning layers. (↗) and (↘) indicates the larger and lower is better, respectively.

HierGNN-BART uses the pretrained architecture BART (Lewis et al., 2020)[5]. We use the same approach to obtain the sentence representation as in (Akiyama et al., 2021). On top of the sentence encoder, we add a two-layer HierGNN to boost the sentence representations. The GSA for HierGNN-BART is implemented as the cross-attention in Transformer decoder, which first attends to the output of the reasoning encoder then the token encoder. For both CNN/DM and XSum, we follow the same fine-tuning settings as in (Lewis et al., 2020) except that we use 40K and 20K training steps for each dataset. We search the best model by the label smoothed cross entropy loss on validation set with one search trial per hyperparameter.

Evaluation Metrics. We use the implementation for ROUGE (Lin and Och, 2004) from

[4] https://github.com/atulkum/pointer_summarizer

[5] https://github.com/facebookresearch/fairseq/tree/main/examples/bart

Google Research[6]. We use the official implementation[7] for BERTScore (Zhang et al., 2019). BERTScore is used with model setting in `roberta-large_L17_noidf_version=0.3.9` and `rescale_with_baseline` as suggested.

Datasets. We describe all our pre-processings for the used datasets as followed,

- **CNN/DM**: For HierGNN-PGN, we directly use the data processed by See et al.[8]. For HierGNN-BART, we remain all the pre-processing steps to be the same as Lewis et al.[9].
- **XSum**: Following Lewis et al., we do not pre-process the XSum dataset, and use the original version in (Narayan et al., 2018)[10].
- **PubMed**: We use the same pre-processing script in `https://github.com/HHousen/ArXiv-PubMed-Sum`. We remove the instances with article have less 3 sentences or abstract have less 2 sentences. We also remove three special tokens: newlines, <S> and </S>.

B Details for Human Evaluation

We adopt several settings to control the quality of human evaluation: 1) we only use summaries shorter than 35 tokens (Sun et al., 2019; Wu et al., 2021). 2) When publishing the tasks on MTurk, we require all referees to be professional English speakers located in one of the following countries: i) Australia, ii) Canada, iii) Ireland, iv) New Zealand, v) the United Kingdom and vi) the United States, with the HIT Approval Rate and number of HITs Approved to be greater than 98% and 1,000. 3) We evaluate 25 instances in CNN/DM testing set in total, while each task is evaluated by three workers on MTurk. These settings give us the results with an inter agreement in the average of 58.96%, 64.92% and 51.52% for Relevance, Informativeness and Redundancy, separately.

C Detailed Results for Human Evaluation

We show the detailed proportions for each choice in human evaluation in Table 10.

[6] `https://github.com/google-research/google-research/tree/master/rouge`

[7] `https://github.com/Tiiiger/bert_score`

[8] `https://github.com/abisee/pointer-generator`

[9] `https://github.com/artmatsak/cnn-dailymail`

[10] `https://github.com/EdinburghNLP/XSum`

Rel.	**Best(↗)**	**Worst(↘)**	**Score(↗)**
HierGNN-BART	**0.40**	0.20	**0.20**
BART	0.29	**0.15**	0.14
T5-Large	0.25	0.17	0.08
BERTSUMABS	0.04	0.48	*-0.44
Inf.	**Best(↗)**	**Worst(↘)**	**Score(↗)**
HierGNN-BART	0.35	**0.16**	0.19
BART	**0.43**	0.19	**0.24**
T5-Large	0.17	0.27	-0.09
BERTSUMABS	0.05	0.39	*-0.34
Red.	**Best(↗)**	**Worst(↘)**	**Score(↗)**
HierGNN-BART	**0.31**	**0.21**	**0.10**
BART	0.21	0.25	-0.04
T5-Large	**0.31**	0.25	0.06
BERTSUMABS	0.17	0.28	-0.11

Table 10: Detailed summary for the human evaluation in terms of Relevance (Rel.), Informativeness (Inf.) and Redundancy (Red.). We show the proportion of each option to be selected as the Best/Worst among the four candidates. (↗) and (↘) indicates the larger is better and lower is better, respectively. *: HierGNN-BART's scores are significantly (by pair-wise t-test with $p < 0.05$, corrected using Benjamini–Hochberg method to control the False Discovery Rate (Benjamini and Hochberg, 1995) for multiple comparison) better than the corresponding system.

D Qualitative Case for Graph-Selection Attention

To demonstrate the effectiveness of the graph-selection attention (GSA) on HierGNN, we visualize the graph-selection attention and compare the token attentions whether graph-selection attention is used (See Figure 5). It turns out graph-selection attention mostly focuses on the top sentences but still captures the critical information in the latter. In this case, graph-selection attention successfully captures *fifth title in Miami* and *Andy Murray* from the middle part of the article during decoding (marked in blue). In contrast, the model without graph-selection attention continuously produces content about the event *Novak Djokovic beat John Isner* (marked in red).

Article 4384:	**Summaries:**
Two hours before the Miami open semifinal, Novak Djokovic practiced his returns in an empty stadium, the ball coming at him quickly because his hitting partner stood three feet inside the baseline to emulate big-serving John Isner. The drill helped. Djokovic achieved a breakthrough service break against Isner and won Friday night, 7-6 (3), 6-2. 'He's probably the best server we have in the game,' Djokovic said. *(2 sentences are abbreviated here)* Novak Djokovic beat John Isner in straight sets to reach the final of the Miami Open on Friday night. *(4 sentences are abbreviated here)* The No. 1-seeded Djokovic closed to within one win of his fifth Key Biscayne title. His opponent Sunday will be two-time champion andy Murray, who defeated Tomas Berdych 6-4, 6-4. *(6 sentences are abbreviated here)* Djokovic is aiming to win his fifth title in Miami and will take on Scotsman Murray in Sunday's Final. *(3 sentences are abbreviated here)*	**Reference:** Novak Djokovic beat John Isner 7-6. The world No. 1 will take on Andy Murray in Sunday's Final. Djokovic is bidding to win his fifth title at Key Biscayne. **HierGNN-PGN LIR w/ GSA:** Novak Djokovic beat John Isner in straight sets to reach the Miami Open. The No.1-seeded Djokovic closed to within one win of his fifth Key Biscayne title. Djokovic will be two-time champion andy Murray, who defeated Tomas Berdych 6-4. **HierGNN-PGN LIR w/o GSA:** Novak Djokovic beat John Isner in straight sets to reach the final of the Miami Open on Friday night. Djokovic achieved a breakthrough service break against Isner and won Friday night, 7-6 (3), 6-2. His opponent Andy Murray defeated Tomas Berdych 6-4, 6-4.

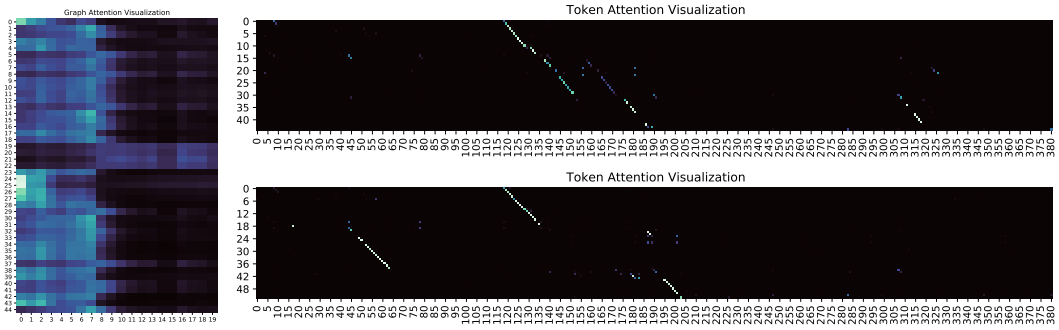

Figure 5: Top Table: CNN/DM testing article 4384 and produced summaries; Bottom Figure: visualization for GSA (left) and HierGNN LIR's token-level attention w/ GSA (right-bottom), and HierGNN-PGN LIR w/o GSA (right-top). X-axis, Y-axis are the encoding and decoding steps, respectively.