HiStruct+: Improving Extractive Text Summarization with Hierarchical Structure Information

Anonymous ACL submission

Abstract

Transformer-based language models usually treat texts as linear sequences. However, most texts also have an inherent hierarchical structure, i. e., parts of a text can be identified using their position in this hierarchy. In addition, section titles usually indicate the common topic of their respective sentences. We propose a novel approach to extract, encode and inject hierarchical structure (HiStruct) information into an extractive summarization model (HiStruct+ model) based on a pre-trained, encoder-only language model. Our HiStruct+ model achieves SOTA extractive ROUGE scores on three public summarization datasets (CNN/DailyMail, PubMed, arXiv), the improvement is especially substantial on PubMed and arXiv. Using various experimental settings, our HiStruct+ model outperforms a strong baseline, which differs from our model only in that the HiStruct information is not injected. The ablation study demonstrates that the hierarchical position information is the main contributor to our model's SOTA performance.

1 Introduction

Texts, especially long documents, contain internal hierarchical structure like sections, paragraphs, sentences, and tokens. When we manually summarize a text, the hierarchical text structure usually plays a key role. Taking a scientific paper as an example, we might focus more on the sections with the titles of "methodology", "discussion", and "conclusion" while paying less attention to the sections like "background". Furthermore, the sentences within one section could have closer relationship with each other, than the ones outside this section. Understanding not only the sequential relations between the sentences but also the internal hierarchical text structure helps us better determine the important sentences within a document. Similarly, a neural summarization model could benefit from these hierarchical structure information.

In this paper, we focus on extractive text summarization of single documents (ETS), which is the task of binary sentence classification with labels indicating whether a sentence should be included in a summary. Recently, pre-trained language models based on Transformers (Vaswani et al., 2017) (TLM), such as BERT (Devlin et al., 2019), have been widely used to extract contextual representations from texts. The pre-trained TLM can be easily reused for fine-tuning on the downstream tasks, so that the representations already learned from the large pre-training corpora are preserved. Liu and Lapata (2019) has achieved the state-of-the-art (SOTA) performance by fine-tuning BERT for extractive summarization on short document datasets including CNN/DailyMail. However, the TLMs consider merely the sequential-context-dependency by adding a linear positional encoding to each input token embeddings. The hierarchical text structure information is not taken into account explicitly.

We propose a novel approach to extract, encode and inject the hierarchical structure (HiStruct) information explicitly into an extractive summarization model (HiStruct+ model), which consists of a Transformer language model (TLM) for sentence encoding and two stacked inter-sentence Transformer layers for hierarchical learning and extractive summarization. We experiment with BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and Longformer (Beltagy et al., 2020) as underlying TLMs. The HiStruct information utilized in our work includes the section titles and the hierarchical positions of sentences, which are encoded using our proposed novel methods. The resulting embeddings can be injected into the TLM sentence representations to provide the HiStruct information for the summarization task.

We evaluate our HiStruct+ models on short documents (i.e., CNN/DailyMail (See et al., 2017)) and long documents (i.e., PubMed and arXiv (Cohan et al., 2018)). Our HiStruct+ models improve

the SOTA extractive ROUGEs on all three datasets. The improvements are especially substantial on PubMed and arXiv, which contain longer scientific papers with conspicuous hierarchical structures. We also compare the HiStruct+ models with the corresponding strong baselines, which differ from our models only in that the HiStruct information is not injected. Using various experimental settings, our models collectively out perform the baselines on the three datasets, indicating the effectiveness of the proposed HiStruct encoding methods. Ablation studies suggest that the performance gains are majorly contributed by the hierarchical position information of sentences.

Our contributions in this work are five-folds: (1) We highlight the importance of the hierarchical text structure which has been rarely considered in language modeling and text summarization. (2) We conceptualize novel measures to compare the hierarchical structure of the datasets. (3) We implement data preprocessing to extract the HiStruct information from the raw datasets. (4) We propose novel methods to encode and inject the HiStruct information into an extractive summarization model explicitly. The effects of different encoding settings and injection settings are systematically investigated. (5) The data containing the extracted HiStruct information, the best HiStruct+ models, as well as the implementation of preprocessing, training and evaluation is available on GitHub[1].

2 Related Work

2.1 Text Summarization

Extractive Text Summarization (ETS) is to identify the most informative sentences within a document. Liu and Lapata (2019) fine-tune BERT with two stacked inter-sentence Transformer layers with a sigmoid classifier for ETS (BERTSUMEXT). Zhang et al. (2019) pre-train a hierarchical Transformer encoder consisting of a sentence encoder and a document encoder (HIBERT). For long documents, Xiao and Carenini (2019) propose a RNN-based ETS model incorporating both the global and the local context (ExtSum-LG). To addressing the problem of redundancy in extractive summaries, the authors further improve their work by introducing redundancy reduction (Xiao and Carenini, 2020). They systematically explore and compare different methods including Trigram Blocking (Paulus et al., 2018), RdLoss, MMR-Select and MMR-Select+ (Xiao and Carenini, 2020). Trigram Blocking is a traditional redundancy reduction method that avoids adding a candidate sentence to the summary if it has trigram overlap with the previously selected sentences. Their previous model combined with the redundancy reduction methods produce SOTA performance in ETS on PubMed and arXiv.

Previous works on ETS take the HiStruct of documents into consideration by introducing a hierarchical attention, where they first learn contextual token representations based on the linear dependencies between tokens and then add additional CNN (Cheng and Lapata, 2016) or RNN (Nallapati et al., 2017) or Transformer (Zhang et al., 2019; Liu and Lapata, 2019) layer(s) to learn document-level representations for each sentence based on the linear dependencies between sentences. However, all these works learn hierarchical representations implicitly. Hierarchical text structure information is not encoded and injected into the model explicitly.

Abstractive text summarization (ATS) is to generate summaries with new sentences which are not present in the source text. BERTSUMABS (Liu and Lapata, 2019) uses the pre-trained BERT as the encoder in its encoder-decoder architecture. Instead of simply using the pre-trained BERT, recent works, including T5 (Raffel et al., 2020), BART (Lewis et al., 2020) and PEGAUSUS (Zhang et al., 2020) pre-train encoder-decoder Transformer models specifically for seq2seq tasks. The first attempt at addressing neural ATS of long documents is undertaken by Cohan et al. (2018). Gidiotis and Tsoumakas (2020) propose a divide-and-conquer approach to train a model to summarize each part of the document separately. To address the essential issue of the quadratic full attention operation of TLMs, Zaheer et al. (2020) propose BigBird with a sparse attention mechanism.

Hybrid text summarization combines ETS, ATS and other techniques as a hybrid system, such as Zhong et al., 2020 (MatchSum) and Pilault et al., 2020.

2.2 Injecting Additional Information

The central idea of our work is inspired by two former works, LAMBERT (Garncarek et al., 2021) and LayoutLM (Xu et al., 2020), where the visual layout information is injected into BERT by adjusting its input embeddings. They consider a document page as a coordinate system and define

[1] https://bit.ly/3CeCVj7

the layout position of each token/image as the coordinates of its bounding box. The layout position is then transformed into the layout embeddings which is added to the original BERT input embeddings. These models cannot be applied to plain texts since the layout positions have to be obtained from scanned document images. We adopt the idea of injecting extra features when using the pre-trained TLM. Unlike former works, our approach makes use of the internal HiStruct information, which can be found in most types of textual data.

3 Methodology

3.1 Hierarchical Structure Information

Hierarchical position (HP) of a sentence is represented as a vector of its positions at each hierarchy-level, adopting the ideas from LAMBERT and LayoutLM.

$$SSV_s = (a_s, b_s) \tag{1}$$

Given the s-th sentence within a document, its HP is represented as a 2-dimensional vector (a_s, b_s), denoted as the sentence structure vector SSV_s, where a_s represents the linear position of the section containing the sentence and b_s is the linear position of the sentence within the section. All sentences within the same section have the same value in the first dimension of the SSV, indicating the close relationships between them. The second dimension indicates more precisely their linear relations within the section. By this very simple numerical formulation, hierarchical relations between sentences are clearly identified.

Section titles (STs) exist in particular in long documents like scientific papers. They usually imply the section content and describe the common topic for its sub-sentences. In our work, we propose to utilize the corresponding ST as an additional HiStruct information when encoding its sub-sentences. There exist typical STs in scientific papers. Similar STs like "Conclusion", "Conclusions" and "Concluding remarks" have the same semantic meaning and can be grouped into one typical ST class of "Conclusions". This is also taken into consideration when encoding the STs.

3.2 Hierarchical Structure Encoding

Hierarchical position embedding is based on the existing linear position encoding methods (PE), including the sinusoidal method (sin) used by Transformer and the learnable method (la) used by BERT. We use one of the PEs to generate position embeddings for each dimension of the SSVs, the results are two DPEs. Using the sin PE, the DPEs are calculated simply by Equations 2 and 3. Using the la PE, the DPEs are initialized randomly and trained with the entire summarization model.

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}}) \tag{2}$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}}) \tag{3}$$

where pos is the absolute linear position of the encoded token within the input sequence and i is the i-th dimension of the position embedding.

Given the s-th sentence with the SSV of (a_s, b_s), and the desired size of the output embeddings d, its hierarchical position embedding sHE can be generated by Equations 4, 5, 6, using different combination modes.

$$sHE_{\text{sum}}(s,d) = PE(a_s,d) + PE(b_s,d) \tag{4}$$

$$sHE_{\text{mean}}(s,d) = \frac{PE(a_s,d) + PE(b_s,d)}{2} \tag{5}$$

$$sHE_{\text{concat}}(s,d) = PE(a_s,\frac{d}{2}) | PE(b_s,\frac{d}{2}) \tag{6}$$

where the symbol | denotes vector concatenation.

Using one of the PEs (i.e., sin or la) associated with one of the combination modes (i.e., sum, mean or concat), it totals 6 different settings of the hierarchical position encoding method: sin-sum, sin-mean, sin-concat, la-sum, la-mean and la-concat.

(Classified) section title embedding is generated by the pre-trained TLM, which is involved in the summarization model. A STE is generated by feeding the tokenized ST to the TLM and summing up the last hidden states at each token position. Similar STs (i.e., STs that have similar keywords and tokens) lead to embeddings that are already similar to each other in some way. Using the classified STE, all intra-class STEs are replaced with the embedding of its corresponding ST class. In the case that a ST does not belong to any class or it falls into more than one class, the original STE is used. Typical ST classes and the corresponding intra-class STs are manually pre-defined depending on the datasets and the domains.

3.3 Model Architecture

Figure 1 illustrates the overview architecture of the proposed HiStruct+ model. It consists of a base TLM for sentence encoding and two stacked inter-sentence Transformer layers for hierarchical learning and extractive summarization. The sequence on top is the input document, tokenized by

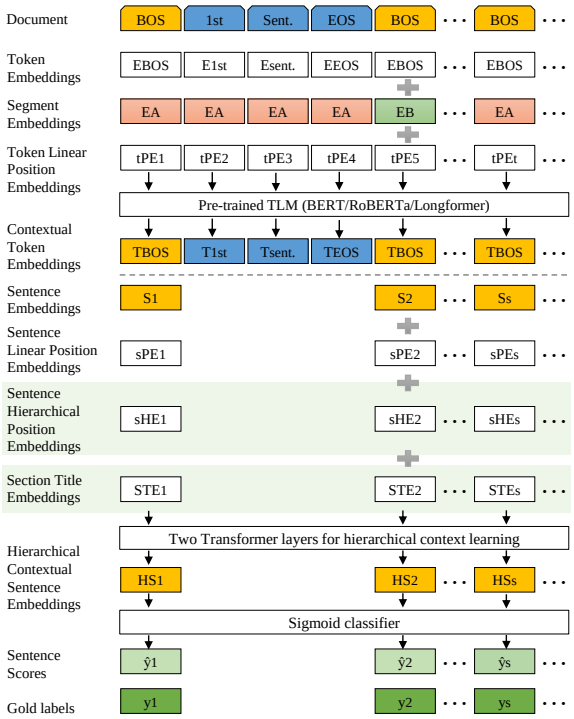


Figure 1: Architecture of the HiStruct+ model. The two blocks shaded in light-green are the HiStruct injection components.

the corresponding tokenizer of the involved TLM. In order to represent individual sentences, we insert a BOS token at the start of every sentence. Only the BOS token embeddings are preserved as the sentence representation (S_s). The S_s is first enriched with a Sentence Linear Position Embedding (sPE_s), which encodes its linear position within the whole document. An additional Sentence Hierarchical Position Embedding can be added (sHE_s). It is generated by encoding the HP of the sentence using the proposed hierarchical position encoding method. If STs are available, we can further enrich the sentence representation by adding a STE or classified STE (STE_s). The sentence representations with the injected HiStruct information are fed to the two stacked Transformer encoder layers to learn inter-sentence document-level hierarchical contextual features. The Self-Attention mechanism in the Transformer layers takes context sentences into consideration when encoding each sentence. The result is a set of Hierarchical Contextual Sentence Embeddings (HS_s). The final output layer is a sigmoid classifier, which calculates the confidence score $\hat{y}_s$ of including the s-th sentence in the extractive summary based on the HS_s. The loss of the model is the binary classification entropy of the prediction $\hat{y}_s$ against the gold label y_s.

The two HiStruct injection components shaded in light-green are optional. Removing these from the HiStruct+ model based on BERT, the architecture is identical to BERTSUMEXT (Liu and Lapata, 2019), which is a strong baseline against our models. When using RoBERTa and Longformer as the base TLM, we also construct a baseline model without the two components. The effectiveness of injecting HiStruct information using the proposed methods can be systematically investigated by comparing our models to the corresponding baselines.

4 Experimental Setup

4.1 Datasets

Our models are evaluated on three benchmark datasets for single document summarization, including CNN/DailyMail (See et al., 2017), PubMed and arXiv (Cohan et al., 2018). Table 4 presents detailed statistics of the datasets.

The three datasets represent different document types ranging from short news articles to long scientific papers. To emphasize the difference in the hierarchical structure among different datasets, we define the concepts of hierarchical depth (hi-depth) and hierarchical width (hi-width). The hi-depth refers to the number of the hierarchy-levels within the document. Scientific papers have a deeper hierarchy consisting of sections, paragraphs, sentences and tokens (i.e., hi-depth = 4). In news articles, paragraphs are not further grouped into sections (i.e., hi-depth = 3). In this case, we use paragraphs instead of sections as the highest hierarchy level when representing the HP of sentences (i.e., the first dimension of the SSVs). The hierarchical width $\text{hi-width} = \frac{N_s}{N_{hsh}}$ is the ratio of total number of sentences N_s and the number of the text-units regarding the highest hierarchy N_{hsh}.

The concept hi-width indicates how many sentences are there on average in every paragraph/section. The more sentences are there, the second dimension of the SSVs has a more wide range of values, and the first dimension of the SSVs differ a lot from the linear sentence positions. Larger hi-depth and larger hi-width indicate that the hierarchical text structure is more conspicuous.

CNN/DailyMail contains more than 310k short news articles. We use the standard splits given by See et al. (2017) for training, validation, and testing. The average hi-width over all documents is 1.33.

The hierarchical structure of the CNN/DailyMail documents is not as conspicuous as the documents in PubMed and arXiv. Compared to PubMed and arXiv, the gold summaries have higher proportions of novel 1-grams and 2-grams in CNN/DailyMail, which is one of the key difficulties in ETS.

During data preprocessing, we first split documents into sentences and paragraphs respectively with the Stanford CoreNLP toolkit (Manning et al., 2014) . The sentences and paragraphs are tokenized, resulting in the lists of sentence tokens and the lists of paragraph tokens. SSVs corresponding to each sentence can be obtained by comparing those lists side by side. For all three datasets, we use a greedy selection algorithm similar to (Nallapati et al., 2017) and (Liu and Lapata, 2019) to select sentences from documents as the gold extractive summaries (ORACLE). Sentences in the ORACLE summaries are assigned with the gold label 1.

PubMed and arXiv contain longer scientific papers. PubMed contains papers in the bio-medical domain, while arXiv contains papers in various domains. The average hi-width over all PubMed documents is 15.79, in arXiv it is 37.33. We use the original splits given by Cohan et al. (2018) for training, validation, and testing. SSVs are obtained by tokenizing the sentences and sections of every document respectively. The details on the generation of STEs and classified STEs can be found in Appendix A.2.

4.2 Implementation Details

We implement our extractive model based on BERTSUMEXT (Liu and Lapata, 2019) using HuggingFace Transformers (Wolf et al., 2020) to make use of the pre-trained instances of BERT, RoBERTa and Longformer. More implementation details are summarized in Appendix A.3 and A.4.

5 Results and Discussion

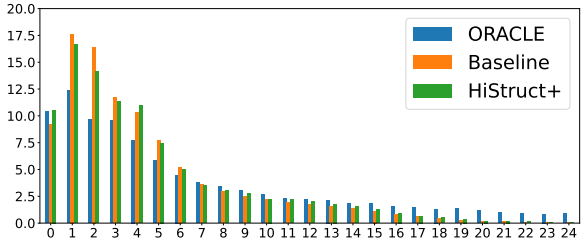
We evaluate the performance of our summarization models automatically using ROUGE metrics (Lin, 2004) including F1 ROUGE-1 (R1), ROUGE-2 (R2) and ROUGE-L (RL). Tables 1, 2 and 3 summarize the performance of our models in comparison to the baselines and the SOTA results on CNN/DailyMail, PubMed and arXiv respectively. The first three blocks in the tables highlight the results reported by the corresponding papers of abstractive, extractive, and hybrid summarization systems. Underlined are the best results regarding the respective type of the summarization system. Bold are the scores of the HiStruct+ models that are better than their corresponding comparison baselines. The symbol * indicates that the corresponding extractive SOTA ROUGE is improved by our model. The symbol ' indicates that the SOTA ROUGEs (incl. all types of summarization approaches) are outperformed.

5.1 Results on Short Documents

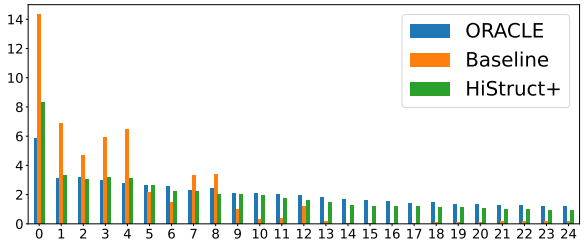
Model ↓ / Metric →	R1	R2	RL
Abstractive			
BERTSUMABS (2019)	41.72	19.39	38.76
BART (2020)	44.16	21.28	40.90
PEGASUS (2020)	44.17	21.47	41.11
BigBird PEGASUS (2020)	43.84	21.11	40.74
Extractive			
HIBERT (2019)			
(BERT-base)	42.31	19.87	38.78
(BERT-large)	42.37	19.95	38.83
BERTSUMEXT (2019)			
(BERT-base)	43.25	20.24	39.63
(BERT-large)	43.85	20.34	39.90
Hybrid			
MatchSum (2020)			
(BERT-base)	44.22	20.62	40.38
(RoBERTa-base)	44.41	20.86	40.55
Reproduced baselines			
ORACLE (512 tok.)	52.46	30.76	48.66
ORACLE (1,024 tok.)	55.45	32.78	51.59
LEAD-3	40.33	17.39	36.56
TransformerETS			
BERT-base (1,024 tok.)	43.32	20.27	39.69
BERT-large (512 tok.)	43.45	20.36	39.83
RoBERTa-base (1,024 tok.)	43.62	20.53	39.99
Our models (Extractive)			
HiStruct+			
BERT-base (1,024 tok.)	**43.38**	**20.33**	**39.78**
BERT-large (512 tok.)	**43.49**	**20.40***	**39.90***
RoBERTa-base (1,024 tok.)	**43.65**	**20.54***	**40.03***
Our models (Hybrid)			
HiStruct+ *RoBERTa-base (1,024 tok.)* & MatchSum (RoBERTa-base)	44.31	20.73	40.47

Table 1: Results on CNN/DailyMail

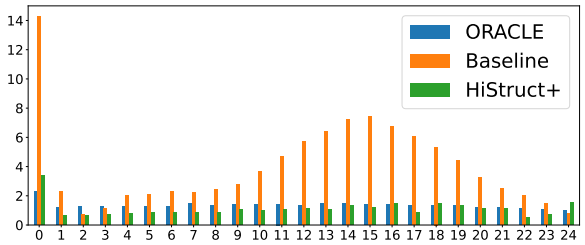
ROUGE results on CNN/DailyMail are summarized in Table 1. In the baselines block, the first two lines highlight the ORACLE results that build the upper bounds for ETS systems taking the same number of input tokens. The LEAD-n baselines simply select the first n sentences in a document as its extractive summary. Despite its simplicity, the LEAD-3 baseline already achieves relatively competitive performance and even outperforms several neural models as listed in the table.



(a) CNN/DailyMail



(b) PubMed



(c) arXiv

Figure 2: Proportions of the extracted sentences at each linear position. The x-axis values are linear sentence indices, the y-axis values are percentages of the extracted sentences.

The three TransformerETS models are the corresponding comparison baselines that use the same model architecture and experimental settings as our models but without injected HiStruct information.

The following block presents the results of our HiStruct+ models based on different TLMs with various input lengths. To make the evaluation results comparable to the SOTA extractive model BERTSUMEXT, we follow their approach and report the averaged results of three best checkpoints.

Regardless the base TLM and input length, our HiStruct+ models collectively outperform the corresponding baselines by merely injecting the HP information of sentences. Comparing to BERTSUMEXT, our best HiStruct+ model, which is based on RoBERTa-base with 1,024 input tokens, increases the SOTA extractive ROUGE-2 and ROUGE-L by 0.2 and 0.13. The performance improvements gained by our models on CNN/DailyMail are small. One of the reasons might be that we merely inject the sHE, STs are not available. Furthermore, as discussed in Section 4, the hierarchical structure of the CNN/DailyMail documents is not so obvious as in PubMed and arXiv. Another possible reason may lie in the abstractiveness of the gold summaries in CNN/DailyMail, which makes it difficult to achieve high ROUGE-1 and ROUGE-2 by simply copying sentences from the source document.

Our HiStruct+ models are the competitive extractive models which can be reused in many hybrid approaches. When we apply MatchSum based on our best model, the ROUGE results are further increased to 44.31/20.73/40.47.

Ablation studies on CNN/DailyMail (see the evaluation results and detailed discussions in Appendix A.5) suggest that the setting la-sum works best for HP encoding . Two stacked Transformer layers in the summarization model perform better than one or three Transformer layers. When taking longer inputs than the length limit of the TLM, significant improvements are achieved by using the copied token position embeddings for initialization instead of random initialization.

The extracted summaries are analyzed in more detail by plotting the proportions of the extracted sentences at each linear position within the whole document as shown in Figure 2a. The model in green is our best HiStruct+ model. The model in orange is the corresponding comparison baseline without injected HiStruct information. The model in blue is the ORACLE system, which produces the gold extractive summaries. We can observe that the ORACLE summary sentences are distributed across documents more smoothly, while our HiStruct+ model and the baseline model tend to select the first sentences and fail to select sentences that appear at later positions within the documents. Compared to the baseline, the HiStruct+ model leads to more similar proportions as the ORACLE summaries at the most sentence indices.

5.2 Results on Long Documents

5.2.1 Results on PubMed

ROUGE results on PubMed are summarized in Table 2. As shown in the baselines block, the ORACLE extractive upper bounds are increased significantly by increasing the input length, which makes it possible to exploit potential gains from modeling longer input. The LEAD-n baselines do not produce competitive results on PubMed. It indicates that the first sentences in PubMed are not so informative as in CNN/DailyMail. The last two

Model ↓ / Metric →	R1	R2	RL
Abstractive			
PEGASUS (2020)	45.49	19.90	42.42
BigBird PEGASUS (2020)	46.32	20.65	42.33
DANCER PEGASUS (2020)	46.34	19.97	42.42
Extractive			
Sent-CLF (2020)	45.01	19.91	41.16
Sent-PTR (2020)	43.30	17.92	39.47
ExtSum-LG+ (2020)			
RLoss	45.30	20.42	40.95
MMR-Select+	45.39	20.37	40.99
Hybrid			
TLM-I+E(G,M) (2020)	42.13	16.27	39.21
Reproduced baselines			
ORACLE (4,096 tok.)	49.73	27.29	45.26
ORACLE (9,600 tok.)	52.80	28.95	48.08
ORACLE (15k tok.)	53.04	29.08	48.31
LEAD-7	38.30	12.54	34.31
LEAD-10	38.59	13.05	34.81
TransformerETS			
Longformer-base (15k tok.)	41.69	15.76	37.48
Longformer-large (15k tok.)	41.69	15.79	37.49
Our models (Extractive)			
HiStruct+			
Longformer-base (15k tok.)			
sHE+STE(classified)	**46.59*'**	**20.39**	**42.11***
sHE+STE	**46.49*'**	**20.29**	**42.02***
sHE	**45.76***	**19.64**	**41.34***
Longformer-large (15k tok.)			
sHE+STE(classified)	**46.38*'**	**20.17**	**41.92***
sHE	**45.67***	**19.60**	**41.26***

Table 2: Results on PubMed

TransformerETS models in the block are the comparison baselines that are not aware of HiStruct.

The last block presents the results of two groups of HiStruct+ models grouped by the base TLM used in the summarization model. In PubMed, we can choose to inject the sHE with or without the STE. STE can be replaced by classified STE. This can result in three different injection settings for a model group, namely sHE, sHE+STE, and sHE+STE(classified). For each model setting, we report the results of the best-performed checkpoint.

Our best HiStruct+ model on PubMed is a model based on Longformer-base taking 15,000 input tokens, which injects the sHE and the classified STE into the extractive model. It achieves ROUGE results of 46.59/20.39/42.11. Compared to the baseline, our model increases ROUGEs by 4.9/4.63/4.63, which indicates the effectiveness of the proposed hierarchical structure encoding and injection methods. Our results also beat the SOTA extractive model ExtSum-LG+MMR-Select+ collectively on all three ROUGE metrics with improvements of 1.2/0.02/1.12. Taking the SOTA abstractive and hybrid approaches into account, our results are still very competitive.

All HiStruct+ models produce the competitive results that are better than or very close to the former extractive SOTA results. They also collectively outperform the baselines by a large margin on all evaluation metrics. This overperformance is much more substantial than that on CNN/DailyMail. One of the reasons might be that we include the STE in addition to the sHE while training on PubMed. Furthermore, the HiStruct of the documents is more obvious than in CNN/DailyMail. The gold summaries in PubMed contain less novel 1-grams and 2-grams, which also makes it easier to achieve higher ROUGEs by performing extractive summarization.

Ablation studies on PubMed suggest that the largest improvement of our models against the comparison baseline is contributed by the sHE. This is observed when we compare the three models in the first group of HiStruct+ models with the baseline. Injecting merely sHE, the results are already increased by 4.07/3.88/3.86. When the STE are included additionally, the results are further increased by 0.73/0.65/0.68. When using classified STE instead, the ROUGEs are increased by a small margin of 0.1/0.1/0.09. In the second group of HiStruct+ models, it is also observed that injecting the sHE leads to the largest performance gain.

The extracted summaries analysis on PubMed test set is demonstrated in Figure 2b. The model in green is our best HiStruct+ model, the model in orange is the corresponding baseline, the model in blue is the ORACLE system. It is observed that the ORACLE summaries are distributed across documents evenly. The comparison baseline favors the first 5 sentences and ignores the sentences appearing at later positions. In contrast, our HiStruct+ model overcomes the problem of focusing merely on the first sentences. The outputs of the HiStruct+ model are close to the ORACLE summaries. It indicates that by injecting HiStruct information explicitly using our proposed method, the model successfully learns the deeper internal hierarchical structure of the PubMed documents and relies less on the linear sentence positions.

5.2.2 Results on arXiv

ROUGE results on arXiv are summarized in Table 3. Similar as on PubMed, the LEAD-n baselines perform badly on arXiv. The results of the HiStruct+ models are presented in two groups. The first group takes 15k input tokens, while the sec-

Model ↓ / Metric →	R1	R2	RL
Abstractive			
PEGASUS (2020)	44.70	17.27	25.80
BigBird PEGASUS (2020)	46.63	19.02	41.77
DANCER PEGASUS (2020)	45.01	17.60	40.56
LED-large (2020)	46.63	19.62	41.48
Extractive			
Sent-CLF (2020)	34.01	8.71	30.41
Sent-PTR (2020)	42.32	15.63	38.06
ExtSum-LG + (2020)			
RLoss	44.01	17.79	39.09
MMR-Select+	43.87	17.50	38.97
Hybrid			
TLM-I+E(G,M) (2020)	41.62	14.69	38.03
Reproduced baselines			
ORACLE (15k tok.)	53.58	26.19	47.76
ORACLE (28k tok.)	53.97	26.42	48.12
LEAD-10	37.37	10.85	33.17
TransformerETS			
Longformer-base (15k tok.)	38.49	11.59	33.85
Longformer-base (28k tok.)	38.47	11.56	33.82
Our models (Extractive)			
HiStruct+			
Longformer-base (15k tok.)			
sHE+STE(classified)	**44.94***	**17.42**	**39.90***
sHE+STE	**45.02***	**17.48**	**39.94***
sHE	**43.04**	**15.87**	**38.13**
Longformer-base (28k tok.)			
sHE+STE(classified)	**45.17***	**17.61**	**40.10***
sHE+STE	**45.22***	**17.67**	**40.16***

Table 3: Results on arXiv

ond group increases the input length to 28k. In the groups, different injection settings are compared.

Our best HiStruct+ model trained on arXiv is based on Longformer-base with 28k input tokens, injecting the sHE with the original STE. This model beats the results achieved by ExtSum-LG+RLoss and sets the new SOTA extractive summarization ROUGEs on arXiv to 45.22/17.67/40.16.

Our HiStruct+ models collectively outperform the corresponding baselines (the last two models in the baselines block) by a large margin on all ROUGEs. This overperformance is much more significant than that on both CNN/DailyMail and PubMed. The arXiv dataset has the largest hi-width among the three datasets and the hierarchical structure is most conspicuous, which might be the reason for the largest performance improvements by injecting HiStruct information on arXiv.

Ablation studies in the first HiStruct+ group also suggest that the largest improvement of our HiStruct+ model against the comparison baseline is contributed by the sHE. The effect of using the classified STE on arXiv is opposite to that on PubMed. The results are decreased slightly when we replace the STE with the classified STE. This phenomenon occurs in the second group of HiStruct+ models as well. We notice the fact that there are 500k unique STs in arXiv, while PubMed contains 164k unique STs. It is no wonder that it becomes much more difficult to group a large number of STs correctly into several section classes. Furthermore, the PubMed dataset contains papers mostly in the bio-medical domain. The structure of those papers tends to follow specific writing conventions in the bio-medical sciences. The arXiv dataset, in contrast, contains scientific papers that are not limited to a specific domain. The document structure and the writing styles are more diverse.

The extracted summaries analysis on arXiv is demonstrated in Figure 2c. The baseline (in orange) tends to select the first sentence and the sentences indexed between 10 and 20, while it excludes sentences at later positions. It is clearly observed that the summary sentences extracted by our model are evenly distributed, the informative sentences appearing at later positions are not ignored.

6 Conclusions

In this paper, we propose a novel approach to extract, encode and inject the **hi**erarchical **struct**ure (HiStruct) information into an extractive summarization model based on pre-trained TLM. We evaluate our models systematically on CNN/DailyMail, PubMed and arXiv. Our models increase the SOTA extractive ROUGEs on all three datasets. The improvement is especially substantial on PubMed and arXiv, which contain longer scientific papers with conspicuous hierarchical structures. On PubMed, our model increases the former extractive SOTA ROUGE-1 by 1.2 and ROUGE-L by 1.12. On arXiv, our model increases the former extractive SOTA ROUGE-1 by 1.21 and ROUGE-L by 1.07. Using various experimental settings, our HiStruct+ models collectively outperform the corresponding strong baselines, which differ from our models only in that the HiStruct information is not taken into account. Ablation studies on PubMed and arXiv indicate that the improvements are mostly gained by providing the hierarchical position information of sentences to the summarization model. The idea of extracting, encoding and injecting the HiStruct information can be easily adopted in abstractive summarization. We see great potential in an encoder-decoder architecture with the proposed HiStruct injection components.

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv:2004.05150*.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494, Berlin, Germany. 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 *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*.

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.

Łukasz Garncarek, Rafał Powalski, Tomasz Stanisławek, Bartosz Topolski, Piotr Halama, Michał Turski, and Filip Graliński. 2021. Lambert: Layout-aware language modeling for information extraction. In *Document Analysis and Recognition – ICDAR 2021*, pages 532–547, Cham. Springer International Publishing.

Alexios Gidiotis and Grigorios Tsoumakas. 2020. A divide-and-conquer approach to the summarization of academic articles. *ArXiv*, abs/2004.06190.

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, Online. Association for Computational Linguistics.

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

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.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. 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 *31st AAAI Conference on Artificial Intelligence, AAAI 2017*.

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

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.

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

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

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.

Wen Xiao and Giuseppe Carenini. 2020. Systematically exploring redundancy reduction in summarizing long documents. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 516–528, Suzhou, China. Association for Computational Linguistics.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020. LayoutLM: Pre-training of Text and Layout for Document Image Understanding. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297. Curran Associates, Inc.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

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

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208, Online. Association for Computational Linguistics.

A Appendix

A.1 Statistics of the datasets

Dataset	CNN/DailyMail	PubMed	arXiv
Raw documents			
avg. #words	792.24	2,967.22	5,825.68
avg. #sentences	40.31	86.37	206.3
avg. #sections*	31.2	5.91	5.55
avg. hi-width	1.33	15.79	37.33
Raw gold summaries			
avg. #words	53.25	202.42	272
avg. #sentences	3.75	6.85	9.61
Novel n-grams in gold summaries			
avg. % novel			
1grams	13.97	0.2	0.15
2grams	51.79	2.69	2.73
Nr. of documents			
#train	287,227	119,924	203,037
#val	13,368	6,633	6,436
#test	11,490	6,658	6,440
Documents tokenized by the RoBERTa tokenizer			
avg. doc length	964	4,252	8,991
75% doc length	1,219	5,382	11,289
85% doc length	1,448	6,709	14,294
99% doc length	2,345	15,277	35,559

Table 4: Statistics of the datasets. * avg. #paragraphs in CNN/DailyMail.

We used the CNN/DailyMail[2] and the PubMed and arXiv datasets[3]. We use the original splits used by See et al. (2017) and Cohan et al. (2018) for training, validation and testing.

A.2 Pre-defined ST classes

The pre-defined dictionaries of the typical ST classes and the corresponding in-class STs will be released in our GitHub project [4].

There are 164,195 unique STs in PubMed, and 500,015 in arXiv, which are encoded as STE respectively using the base TLM.

For PubMed, we define 8 ST classes: introduction, background (i.e., background, review and related work), case (i.e., case reports), method, result, discussion, conclusion and additional information (i.e., additional information such as conflicts of interest, financial support and acknowledgements). For arXiv, we define 10 classes: introduction, background, case, theory (i.e., problem formulation and proof of theorem), method, result, discussion, conclusion, reference and additional information. Classified STEs are prepared accordingly.

[2] https://cs.nyu.edu/~kcho/DMQA/

[3] https://github.com/armancohan/long-summarization

[4] https://bit.ly/3CeCVj7

A.3 Implementation Details

The learning rate schedule follows (Liu and Lapata, 2019) with warming-up. On CNN/DailyMail, we train our models 50,000 steps with 10,000 warming-up steps. On PubMed and arXiv, we train our models 70,000 steps with 10,000 warming-up steps when taking 15,000 tokens as input. When we train models on arXiv with 28,000 input tokens, we train the models 100,000 steps with 10,000 warming-up steps.

The number of the extracted sentences is various depending on the dataset. On CNN/DailyMail, we follow (Liu and Lapata, 2019) to select 3 sentences for each document as its extractive summary and apply Trigram Blocking (Paulus et al., 2018) to reduce the redundancy of the selected sentences. On PubMed and arXiv, 7 sentences are extracted. Trigram Blocking is not applied on PubMed and arXiv.

The length limit of the original TLM is overcome by adding extra token linear position embeddings (tPE) to cover the desired length. The additional tPE are trained with the whole summarization model. Instead of initializing them randomly, we copy the original tPE of the base TLM multiple times until the desired length is covered.

The HiStruct+ models are trained on 3 GPUs (NVIDIA® Quadro RTX™ 6000 GPUs with 24GB memory) with gradient accumulation every two steps. Checkpoints are saved and evaluated on the validation set every 1,000 steps. The top-3 checkpoints based on the validation loss are kept. The batch size varies with the base TLM and the input length. The base TLM is not fine-tuned when training the summarization model on PubMed and arXiv due to resource limitation.

A.4 Model Architectures and Experimental Settings

The detailed model architectures and experimental settings for models trained on CNN/DailyMail, PubMed and arXiv are summarized in Table 5, Table 6 and Table 7. The detailed model architectures and experimental settings include:

Base TLM: the base Transformer language model used for sentence encoding in the summarization system

Input length: How many tokens are taken as input

Extra tPE: How to initialize the extra input token position embeddings when taking longer input. We can choose to randomly initialize them or copy the original ones.

FT: Whether the base TLM is fine-tuned with the entire summarization model

TL: The number of the Transformer layers stacked upon the base TLM for extractive summarization

WS: Warmup steps, how many steps are used for warming-up of the learning rate

TS: Training steps, the total training steps

BS: Batch size, how many documents are used as one batch during training

AC: Accumulation count, gradient accumulation every k steps

GPU: The number of GPUs used for training, we use NVIDIA® Quadro RTX™ 6000 GPUs with 24GB memory

HiStruct: The injection setting. Hierarchical structure information that can be injected into the summarization model are: sHE (i.e., sentence hierarchical position embeddings), STE (i.e., section title embeddings), or STE(classified) (i.e., classified section title embeddings)

HPE: The hierarchical position encoding method used in the model. The method is based on the sinusoidal (sin) or the learnable (la) linear position embedding method associated with a combination mode (sum/mean/concat)

#PE: The numbers of the learned position embeddings for each hierarchy-level and the linear sentence positions, when using the learnable position embedding method. We set them to a same value during training.

SS: Saving steps, save checkpoints every k steps

n: Select n sentences as the extractive summary for each document

TB: Trigram Blocking, whether to apply Trigram Blocking during sentence selection

A.5 Ablation studies on CNN/DailyMail

The effect of token HP embeddings is investigated in experiments. The HP embeddings of tokens are generated as followings:

Given the t-th token within the document,its HP can be represented by Equation 7:

$$TSV_t = (a_t, b_t, c_t) \tag{7}$$

where a_t represents the linear position of the section which contains the token, b_t is the sentence's position within the section and c_t is the linear position of the token within the sentence.

Given the t-th token whose TSV is a 3-dimensional vector (a_t, b_t, c_t), and the desired size of the output embeddings d, we can embed its token-level hierarchical position embeddings tHE by Equations 8, 9, 10, using different combination settings.

$$tHE_{\text{sum}}(t,d) = PE(a_t,d) + PE(b_t,d) + PE(c_t,d) \tag{8}$$

$$tHE_{\text{mean}}(t,d) = \frac{PE(a_t,d) + PE(b_t,d) + PE(c_t,d)}{3} \tag{9}$$

$$tHE_{\text{concat}}(t,d) = PE(a_t, \frac{d}{3}) | PE(b_t, \frac{d}{3}) | PE(c_t, \frac{d}{3}) \tag{10}$$

Initial experiments are conducted to assess the summarization performance of the HiStruct+ models with or without the tHE. For this purpose, we compare a HiStruct+ model merely injecting sentence HP embeddings (i.e., sHE) with a HiStruct+ model with both sentence and token HP embeddings (i.e., sHE& tHE). The former is denoted as HiStruct(sHE)+ in Table 8, while the latter is denoted as HiStruct(sHE&tHE)+. The HiStruct(sHE&tHE)+ models add the corresponding tHEs to the input embeddings at each input position, and sHEs to the TLM sentence representations. The HiStruct(sHE)+ models merely add sHEs. The averaged summarization ROUGEs of three best checkpoints are reported in the Table 8. The table summarize three groups of HiStruct+ models based on different TLM with various input lengths. The detailed model architectures and experimental settings of all models in 8 are summarized in Table 9.

The experimental results suggest that the HiStruct(sHE)+ models with merely sHE consistently outperform the HiStruct(sHE&tHE)+ models

Models/Settings	Base TLM	Input length	Extra tPE	BS	HiStruct	HPE	#PE
Reproduced baselines							
TransformerETS							
BERT-base (1,024 tok.)	BERT-base	1,024	copied	200	none	-	-
BERT-large (512 tok.)	BERT-large	512	-	100	none	-	-
RoBERTa-base (1,024 tok.)	RoBERTa-base	1,024	copied	250	none	-	-
Our models (Extractive)							
HiStruct+							
BERT-base (1,024 tok.)	BERT-base	1,024	copied	200	sHE only	la-sum	407
BERT-large (512 tok.)	BERT-large	512	-	100	sHE only	la-sum	407
RoBERTa-base (1,024 tok.)	RoBERTa-base	1,024	copied	250	sHE only	la-sum	407

Table 5: Detailed model architectures and experimental settings for models trained on CNN/DailyMail (also see Table 1). The settings not included in the table are the same for all models. FT: yes, TL:2, WS:10,000, TS:50,000, AC:2, GPU:3, SS:1,000, n: 3, TB:yes.

Models/Settings	Base TLM	BS	HiStruct	HPE	#PE
Reproduced baselines					
TransformerETS					
Longformer-base (15k tok.)	Longformer-base	500	none	-	-
Longformer-large (15k tok.)	Longformer-large	256	none	-	-
Our models (Extractive)					
HiStruct+					
Longformer-base (15k tok.)					
sHE+STE(classified)	Longformer-base	500	sHE+STE(classified)	la-sum	450
sHE+STE	Longformer-base	500	sHE+STE	la-sum	450
sHE	Longformer-base	500	sHE only	la-sum	450
Longformer-large (15k tok.)					
sHE+STE(classified)	Longformer-large	256	sHE+STE(classified)	la-sum	450
sHE	Longformer-large	256	sHE only	la-sum	450

Table 6: Detailed model architectures and experimental settings for models trained on PubMed (also see Table 2). The settings not included in the table are the same for all models. Input length: 15,000; Extra tPE: copied; FT: no; TL:2; WS:10,000; TS:70,000; AC:2; GPU:3; SS:1,000; n: 7; TB:no.

under various circumstances. The reason might be that we directly fine-tune the TLM on the extractive summarization task. When adding extra tHE to the input embeddings to the TLM, we do not pre-train the TLM with the adjusted inputs. It is reasonable that the TLM has difficulties in understanding of the new inputs based on the knowledge learned from the original format of encodings. Previous works, such as LayoutLM (Xu et al., 2020), LamBERT (Garncarek et al., 2021) and HIBERT (Zhang et al., 2019), which adjust the input embeddings or the encoder architecture of the pre-trained TLM, continue to pre-train the released instances of pre-trained TLM on their own data. Continuing pre-training of the language models is a core part of these works and leads to significant improvements on downstream tasks. Due to lack of computing resources, we are not able to pre-train the language models. Furthermore, the key goal of our work is to experiment with various methods to make use of the internal hierarchical text structure information for extractive summarization. In this work, we conduct further experiments without token HP information and leave for future work the pre-training of language models with the adjusted input embeddings.

The effect of different settings for HP encoding is also investigated. As explained previously, based on different PE methods (i.e., the sin. or la. PE) associated with various combination modes (i.e., sum, mean, concat), we have totally 6 different settings for hierarchical position encoding. We investigate the effect of those 6 settings systematically in experiments while keeping the rest settings and parameters the same. Therefore, their summarization results are comparable.

Table 10 summarizes the ROUGE results of 6 HiStruct+ models using the 6 hierarchical position encoding settings respectively, which are all trained on CNN/DailyMail based on BERT-base with 1,024 input tokens, injecting merely sHE. The detailed model architectures and experimental set-

Models/Settings	Input length	TS	BS	HiStruct	HPE	#PE
Reproduced baselines						
TransformerETS						
Longformer-base (15k tok.)	15,000	70,000	500	none	-	-
Longformer-base (28k tok.)	28,000	100,000	500	none	-	-
Our models (Extractive)						
HiStruct+						
Longformer-base (15k tok.)						
sHE+STE(classified)	15,000	70,000	500	sHE+STE(classified)	la-sum	720
sHE+STE	15,000	70,000	500	sHE+STE	la-sum	720
sHE	15,000	70,000	500	sHE only	la-sum	720
Longformer-base (28k tok.)						
sHE+STE(classified)	28,000	100,000	500	sHE+STE(classified)	la-sum	1300
sHE+STE	28,000	100,000	500	sHE+STE	la-sum	1300

Table 7: Detailed model architectures and experimental settings for models trained on arXiv (also see Table 3). The settings not included in the table are the same for all models. Base TLM: Longformer-base; Extra tPE: copied; FT: no; TL:2; WS:10,000; AC:2; GPU:3; SS:1,000; n: 7; TB:no.

Experimental Results	R1	R2	RL
BERT-base (512 tok.)			
HiStruct(sHE)+	43.23	20.15	39.65
HiStruct(sHE&tHE)+	40.76	18.03	37.08
BERT-base (1,024 tok.)			
HiStruct(sHE)+	43.38	20.33	39.78
HiStruct(sHE&tHE)+	41.04	18.25	37.41
BERT-large (512 tok.)			
HiStruct(sHE)+	43.46	20.4	39.85
HiStruct(sHE&tHE)+	40.58	17.71	36.83

Table 8: Ablation study on CNN/DailyMail (a)

tings are summarized in Table 11.

We observe that when using the la PE, the combination mode sum leads to better results compared to the mean and concat modes (see the first three columns in Table 10). When using the sin PE, the various combination modes do not make a conspicuous difference in summarization performance. The sum and concat modes perform slightly better . When using sum mode, the la and the sin PE produce similar results (see the first row of ROUGEs in Table 10).

The effect of using the sin vs the la PE method is further investigated in experiments. As discussed above, the HP encoding methods la-sum and sin-sum lead to similar results. We conduct experiments to further investigate the effect of using the la-sum vs sin-sum method. We also compare our HiStruct+ models with the corresponding strong baseline model which differs from our models only in that it does not take into account extra HiStruct information.

Table 12 includes the ROUGEs of three set of comparison models, which use an extended BERT-base model taking 1,024 input tokens, an original BERT-large instance and an extended RoBERTa-base model with 1,024 input tokens respectively as the base TLM in the extractive summarization system. In each block, the first row is the baseline. The second row is a HiStruct+ model which injects sHE encoded by the method la-sum, denoted as HiStruct(la-sum)+. The third row is a similar HiStruct+ model using the sin-sum method for HP encoding, denoted as HiStruct(sin-sum)+. The detailed model architectures and experimental settings of all models included in Table 12 are summarized in Table 13.

Regardless of the hierarchical position encoding method used, all HiStruct+ models produces better ROUGE-1, ROUGE-2 and ROUGE-L on CNN/DailyMail compared to the strong baseline. This indicates the potential benefits of the hierarchical structure information and the effectiveness of our proposed methods for hierarchical position encoding. However, the improvements compared to the baseline are not significant. It is also observed in Table 12 that the HiStruct(la-sum)+ models outperform slightly the HiStruct(sin-sum)+ models under all the three different settings. The differences of using the sin and the la PE method are not significant on CNN/DailyMail.

The effect of the number of the stacked Transformer layers in the HiStruct+ models is investigated in our experiments. We fine-tune an extended BERT-base model with 1,024 input tokens for extractive summarization. The method sin-sum is used to generate sHE. We build the HiStruct+ mod-

Models/Settings	Base TLM	Input length	Extra tPE	BS	HiStruct	HPE	#PE
BERT-base (512 tok.)							
HiStruct (sHE) +	BERT-base	512	-	400	sHE only	sin-sum	-
HiStruct (sHE&tHE) +	BERT-base	512	-	400	sHE & tHE	sin-sum	-
BERT-base (1,024 tok.)							
HiStruct (sHE) +	BERT-base	1024	copied	200	sHE only	la-sum	407
HiStruct (sHE&tHE) +	BERT-base	1024	copied	200	sHE & tHE	la-sum	407
BERT-large (512 tok.)							
HiStruct (sHE) +	BERT-large	512	-	100	sHE only	sin-sum	-
HiStruct (sHE&tHE) +	BERT-large	512	-	100	sHE & tHE	sin-sum	-

Table 9: Detailed model architectures and experimental settings for ablation study (a) on CNN/DailyMail (also see Table 8). The settings not included in the table are the same for all models. FT: yes; TL:2; WS:10,000; TS:50,000; AC:2; GPU:3; SS:1,000; n: 3; TB:yes.

	la PE			sin PE		
	R1	R2	RL	R1	R2	RL
HiStruct+BERT-base (1,024 tok.)						
sum	43.38	20.33	39.78	43.37	20.27	39.75
mean	43.33	20.31	39.73	43.33	20.28	39.72
concat	43.22	20.18	39.61	43.37	20.29	39.74

Table 10: Ablation study on CNN/DailyMail (b)

Models/Settings	HiStruct	HPE	#PE
HiStruct+BERT-base (1,024 tok.)			
la-sum	sHE only	la-sum	407
la-mean	sHE only	la-mean	407
la-concat	sHE only	la-concat	407
sin-sum	sHE only	sin-sum	-
sin-mean	sHE only	sin-mean	-
sin-concat	sHE only	sin-concat	-

Table 11: Detailed model architectures and experimental settings for ablation study (b) on CNN/DailyMail (also see Table 10). The settings not included in this table are the same for all models. Base TLM:BERT-base; Input length:1,024; Extra tPE:copied; FT: yes; TL:2; WS:10,000; TS:50,000; BS:200, AC:2; GPU:3; SS:1,000; n: 3; TB:yes.

Experimental Results	R1	R2	RL
BERT-base (1,024 tok.)			
baseline	43.32	20.27	39.69
HiStruct(la-sum)+	43.38	20.33	39.78
HiStruct(sin-sum)+	43.37	20.27	39.75
BERT-large (512 tok.)			
baseline	43.45	20.36	39.83
HiStruct(la-sum)+	43.49	20.4	39.9
HiStruct(sin-sum)+	43.46	20.4	39.85
RoBERTa-base (1,024 tok.)			
baseline	43.62	20.53	39.99
HiStruct(la-sum)+	43.65	20.54	40.03
HiStruct(sin-sum)+	43.64	20.56	40.02

Table 12: Ablation study on CNN/DailyMail (c).

els with 1, 2, 3 stacked transformer layers respectively, while keeping all other settings the same. The ROUGEs of those three HiStruct+ models are reported in the first block in Table 14. The detailed model architectures and experimental settings of all models in the table can be found in Table 15. Our experimental results suggest that two stacked Transformer layers perform best in our HiStruct+ models for extractive summarization.

The effect of random initialization vs copied initialization for the additional input token position embeddings is also investigated in experiments. When taking input texts longer than the original input length of the base TLM, we need to add extra input token position embeddings (tPE) for each extended position. We can choose to randomly initialize the extra tPE or copy the original ones to cover the extended input length. To investigate the effect of different initialization strategies, we use the basic settings of the HiStruct+ model with two summarization layers, namely the second model in the first block in Table 14. To build the comparison model, only the initialization strategy is changed to random. As shown in the second block in Table 14, substantial improvements are achieved by using the copied tPEs for initialization instead of random initialization. ROUGE-1, ROUGE-2 and ROUGE-L are increased by 2.84, 2.51 and 2.95 respectively. We assume that the released token position embeddings of the pre-trained TLM already capture local structure within the 512 tokens window. The knowledge about the local structure is preserved when we copy the released tPEs to an additional text window containing 512 tokens for initialization. This might be the reason for the

Models/Settings	Base TLM	Input length	Extra tPE	BS	HiStruct	HPE	#PE
BERT-base (1,024 tok.)							
baseline	BERT-base	1024	copied	200	none	-	-
HiStruct (la-sum) +	BERT-base	1024	copied	200	sHE only	la-sum	407
HiStruct (sin-sum) +	BERT-base	1024	copied	200	sHE only	sin-sum	-
BERT-large (512 tok.)							
baseline	BERT-large	512	-	100	none	-	-
HiStruct (la-sum) +	BERT-large	512	-	100	sHE only	la-sum	407
HiStruct (sin-sum) +	BERT-large	512	-	100	sHE only	sin-sum	-
RoBERTa-base (1,024 tok.)							
baseline	RoBERTa-base	1024	copied	250	none	-	-
HiStruct (la-sum) +	RoBERTa-base	1024	copied	250	sHE only	la-sum	407
HiStruct (sin-sum) +	RoBERTa-base	1024	copied	250	sHE only	sin-sum	-

Table 13: Detailed model architectures and experimental settings for ablation study (c) on CNN/DailyMail (also see Table 12). The settings not included in this table are the same for all models. FT: yes; TL:2; WS:10,000; TS:50,000; AC:2; GPU:3; SS:1,000; n: 3; TB:yes.

Experimental Results	R1	R2	RL
HiStruct(sin-sum,sHE)+ BERT-base (1,024 tok.)			
-#Transformer layers for summarization			
1	43.29	20.25	39.69
2	43.37	20.27	39.75
3	43.16	20.15	39.56
-Extra input token position embeddings(tPE)			
Randomly initialized	40.53	17.76	36.8
Copied	43.37	20.27	39.75
-With/without sentence position embeddings(sPE)			
With sPE	43.37	20.27	39.75
Without sPE	43.31	20.25	39.69

Table 14: Ablation study on CNN/DailyMail (d).

Models/Settings	Extra tPE	TL
HiStruct(sin-sum,sHE)+BERT-base (1,024 tok.)		
# transformer layers for summarization		
1	copied	1
2	copied	2
3	copied	3
Extra input token position embeddings (tPE)		
Randomly initialized	randomly initialized	2
Copied	copied	2
With/without sentence position embeddings (sPE)		
With sPE	copied	2
Without sPE	copied	2

Table 15: Detailed model architectures and experimental settings for ablation study (d) on CNN/DailyMail (also see Table 14). The settings not included in this table are the same for all models. Base TLM: BERT-base; Input length:1,024; FT: yes; WS:10,000; TS:50,000; BS:200; AC:2; GPU:3; HiStruct: sHE only; HPE:sin-sum; #PE:-; SS:1,000; n: 3; TB:yes.

significant superiority over random initialization.

The effect of the linear sentence position embeddings is also investigated in experiments. As shown in Figure 1, besides the hierarchical positions of each sentence, we also take the linear position of each sentence within the whole document into account by adding a linear sentence position embedding (sPE) to each sentence representation. We assess the effect of the linear sentence position embeddings by comparing two HiStruct+BERT-base models with or without the sPE. The experimental results are summarized in the third block in Table 14. The HiStruct+ model with sPE outperforms the HiStruct+ model without sPE by a small margin regarding all ROUGE metrics.