Parallel Hierarchical Transformer with Attention Alignment for Abstractive Multi-Document Summarization

Harishchandra Chaudhary
Harishchandrachaudhary2000@gmail.com

Abstract:

In comparison to single-document summarization, abstractive Multi-Document Summarization (MDS) brings challenges on the representation and coverage of its lengthy and linked sources. This study develops a Parallel Hierarchical Transformer (PHT) with attention alignment for MDS. By incorporating word- and paragraphlevel multi-head attentions, the hierarchical architecture of PHT allows better processing of dependencies at both token and document levels. To guide the decoding towards a better coverage of the source documents, the attention-alignment mechanism is then introduced to calibrate beam search with predicted optimal attention distributions. Based on the WikiSum data, a comprehensive evaluation is conducted to test improvements on MDS by the proposed architecture. By better handling the inner- and cross-document information, results in both ROUGE and human evaluation suggest that our hierarchical model generates summaries of higher quality relative to other Transformer-based baselines at relatively low computational cost.

Keywords: Transformer, hierarchical structure, multi-document summarization, attention, decoding, scoring function.

1. Introduction:

Since Sutskever et al. (2014) propose the sequence-to-sequence (seq2seq) model for machine translation, the development of NLP applications has been almost inseparable from this framework. In the field of abstractive summarization, the seq2seq model is first applied by Rush et al. (2015) to summarize sentences. With respect to the recent bloom of the attention mechanism and pre-trained models, great effort has been made to improve neural machine summarization upon extensions of seq2seq (Gehrmann et al., 2018; See et al., 2017; Zhang et al., 2019). With the promising results on single documents (See et al., 2017; Gehrmann et al., 2018; Lewis et al., 2020; Pradhan et al., 2021; Liao et al., 2021; Liang et al., 2021), there are increasing recent attempts to study abstractive multi-document summarization (MDS) in the seq2seq framework (Liu et al., 2018; Lebanoff et al., 2018; Fabbri et al., 2019; Liu and Lapata, 2019; Ma et al., 2020).

This study makes an exploratory attempt to improve the established abstractive summarization models for multi-document summarization (MDS) utilizing the Transformer (Vaswani et al., 2017) architecture. In comparison to single-document summarization, MDS brings challenges on the representation and coverage of its lengthy and linked sources. Liu et al. (2018) propose a two-stage model to first extractively select the important paragraphs, then train the concatenated flat sequences using the Transformer-decoder with memory compressed attention (T-DMCA). Although the two-stage approach effectively reduces redundant information of source documents and retains salient information as inputs, it fails to take into account the cross-document relationship in its summaries. On the other hand, Liu and Lapata (2019) propose a Hierarchical Transformer (HT) with local and global encoder layers to represent cross-token and cross-document information. Summaries are then generated based on a vanilla Transformer (Vaswani et al., 2017) by concatenating document-information-enriched token embeddings to a flat sequence. The essentially flat nature of the model leads to restrictions on learning dependencies of input sequences longer than 2000 tokens (Liu et al., 2018).

As a solution to better process the long-term dependency and cross-document relationship in MDS, this study develops a novel Parallel Hierarchical Transformer (PHT) with the paragraph-level attention alignment. Operationally, PHT first creates the word- and paragraph-level context vectors from a shared encoder, then generates summaries by the the word- and paragraph-level multi-head attentions parallel to each other in the decoder. In this way, PHT allows a pure hierarchical learning structure extended from the vanilla Transformer

(Vaswani et al., 2017) to learn both cross-token and cross-document relationships. The word- and paragraphlevel context vectors are then jointly used to generate target sequences in order to address the long-dependency problem of the flat structure, thus to permit extended length of input document. Our experiments show the sole PHT model has already the capacity to outperform other strong Transformer-based summarization models.

To address the coverage of the multi-document summaries, the decoding inference is further modified according to the proposed attention-alignment algorithm. As the original beam search prefers to generate typical and dull sentences to avoid making mistakes (Holtzman et al., 2019), the paragraph-level attention alignment mechanism is designed to regulate generated summaries to attend to source contents with the optimal coverage of salient information. Inspired by Google's Neural Machine Translation (NMT) (Wu et al., 2016), attention alignment taps into the determination of the optimal attention distribution of source paragraphs on summaries, by predicting the reference attention from the source. The score function of the beam search is then refined in order to select summaries closest to the predicted optimal attention distribution. With significantly elevated ROUGE scores, it is evident that incorporating the attention-alignment mechanism further enhances the quality of generated summaries with minor computational cost-added from a shallow attention-prediction model, of which inputs and labels are both extracted from the PHT model.

With regards to the core target of developing an enhanced paradigm for multi-document summarization based on Transformer, the contribution of this study is twofold. First, the hierarchical architecture with parallel multihead attentions is designed to represent and exchange token- and document-level information for the generation of summaries based on the lengthy inputs. The effectiveness of the PHT model is investigated relative to a variety of summarization models, in terms of the ability to capture cross-document relationship, computational efficiency and improvements on the summarization quality. Second, the paragraph-level attention-alignment mechanism is proposed to guide the generated summaries in the decoding stage to calibrate the original beam search according to the learned attention distribution. The merits of attention alignment are not only reflected by promoting the optimal coverage of the generated summary to the source, but also its practical value of low computational cost and potential to be adopted by other attention-based summarization models.

The remaining of the paper is organized as follows. Section 2 discusses the related work. Section 3 and 4 introduce the methodology associated with the Parallel Hierarchical Transformer and the attention-alignment mechanism. Section 5 and 6 describe the experimental setups and analyze the results. Section 8 concludes.

2. Related Work

In general, hierarchical models are designed with strengthened capacity to handle lengthy inputs, which have been widely used in document classification (Yang et al., 2016) or large-document summariztion (Li et al., 2018b) tasks. In the filed of MDS, hierarchical structures allow not only to represent massive source inputs, but also to capture cross-document relationships. Fabbri et al. (2019) use a hierarchical RNN structure with Maximal Marginal Relevance (Carbonell and Goldstein, 1998) to better select salient paragraphs and reduce repetitions in the summary. Zhang et al. (2019) pre-train a hierarchical BERT (Devlin et al., 2018) by masking a sentence and using other sentences to generate the masked one. Liu and Lapata (2019) propose a Hierarchical Transformer for multi-document summarization to enrich token embeddings with cross-document relationships. A vanilla Transformer (Vaswani et al., 2017) is then used to conduct summarization after combining the enriched token embeddings in a flat sequence (Liu et al., 2018).

With the aim to improve the quality of seq2seq summaries, existing studies tend to focus on the coverage of salient contents. Liu et al. (2018) use a text-ranking mechanism to extract important paragraphs, which are later input to the neural abstractive model. Gehrmann et al. (2018) train a selector to predict the phrases ought to appear in the final summaries and use them as summarization inputs. Chen and Bansal (2018) select and compress salient sentences that are later re-organized in the summaries. In addition to the two-stage extraction-abstraction approaches, attempts are made to build hybrid summarization models by incorporating the sentence-level attention (Mei et al., 2015; Cohan et al., 2018; You et al., 2019), graph neural networks (Tan et al., 2017; Liang et al., 2021; Li et al., 2020), Maximal Marginal Relevance (Lebanoff et al., 2018; Fabbri et al., 2019) or reinforcement learning (Yao et al., 2018). Additionally, Li et al. (2018a) add representations of key words to the summarizaiton model. Hua and Wang (2019) use an abstractive summarization model to concatenate extracted key phrases. Moreover, some studies suggest to improve the summarizaiton quality by modifying the objective function to encourage salient words (Pasunuru and Bansal, 2018) or to penalize repetitive generations (See et al., 2017; Welleck et al., 2019). More details on representative methods of this sort and their differences with attention alignment are discussed in Section 4.1.

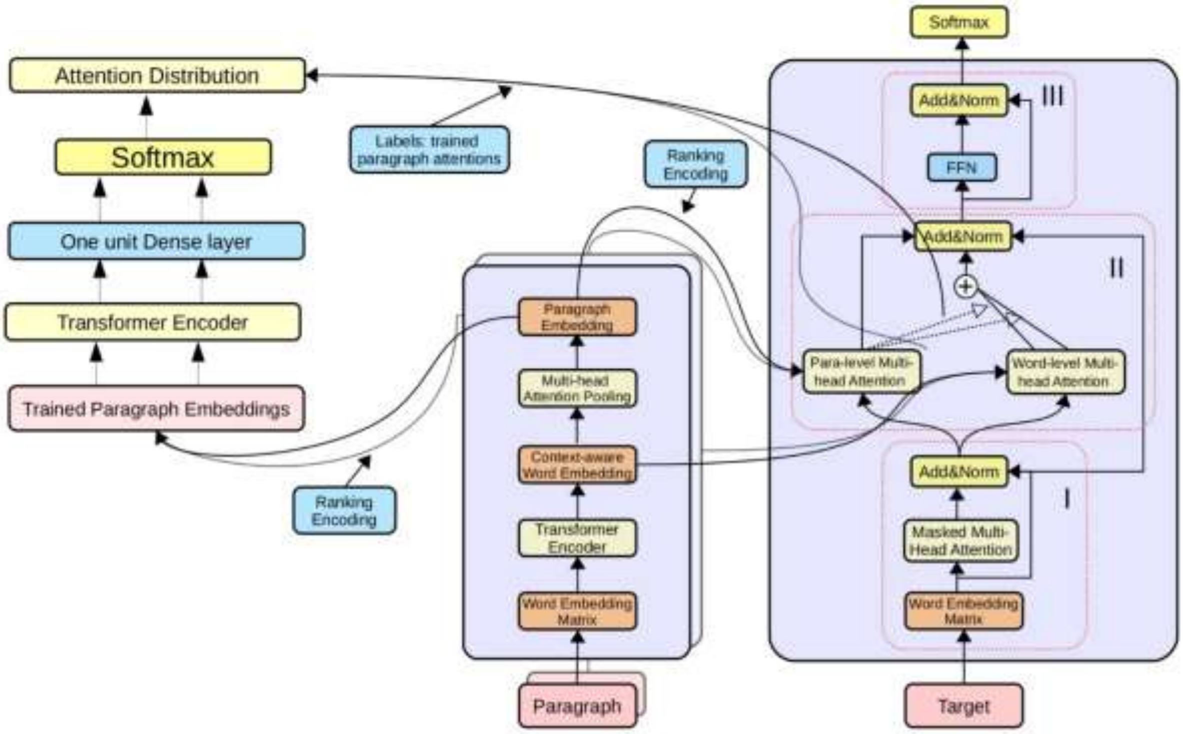

Figure 1: Model flowchart with two input paragraphs. Middle and right: PHT encoder and decoder (See Section 3). Left: prediction model of the optimal attention distribution (See Section 4).

3. Parallel Hierarchical Transformer

This section discusses the design of the Parallel Hierarchical Transformer for MDS. Figure 1 graphically presents the architecture of the proposed PHT. The encoder and decoder are displayed the second and third blocks on the left (highlighted in purple), respectively. The process of generating summaries is described as follows.

3.1. Encoder

As shown in Figure 1, the PHT encoder is shared by all paragraphs and consist of two major units, i.e. the Transformer encoder and the additional Multi-head Attention Pooling layer, to obtain the token- and paragraphembeddings, respectively. To be specific, context-aware word embeddings $C_p \in \mathbb{R}^{n \times d}$ in the paragraph p of length n are first produced as the output of the Transformer encoder TransE(·) based on the summation of word embeddings $W_p \in \mathbb{R}^{n \times d}$ in the paragraph and their fixed positional encodings $E \in \mathbb{R}^{n \times d}$ (Vaswani et al.,
2017).

$$C_p = \mathrm{TransE}(W_p + E) \tag{1}$$

The context-aware word embedding is then used to compute paragraph embeddings as well as being a part of inputs to the PHT decoder to calculate word-level cross attention.

At the second step, PHT achieves paragraph embeddings based on C_p using **Multi-head Attention Pooling**:

$$\text{HeadSplit}(C_p W_1)$$

$$\phi_p^h = \text{Softmax}(head_p^h \mathbf{W}_2))^\top head^h_p \tag{3}$$

$$\phi_p - \mathbf{W}_3 \left(\Big\Vert_{h-1}^{heads} \phi_p^h \right) \tag{4}$$

$$\varphi_p := \text{LayerNorm}(\varphi_p + \text{FFN}(\varphi_p) \tag{5}$$

where k represents concatenation, and $W_1 \in \mathrm{R}^{d \times d}$, $W_2 \in \mathrm{R}^{d_{head} \times 1}$, $W_3 \in \mathrm{R}^{d \times d}$ are linear transformation parameters. Besides, $head_p^h \in \mathbb{R}^{n \times d_{head}}$ denotes the h_{th} attention head, and $\phi_p^h \in \mathbb{R}^{d_{head}}$ is paragraph embedding of the head. These head embeddings are concatenated and fed to a two-layer feed forward network (FFN) with Relu activation function after linear transformation. The paragraph embedding is another input to the decoder for obtaining paragraph-level cross attention, together with the context-aware word embedding.

3.2. Decoder

The PHT decoder accepts three classes of inputs, namely the target summary, context-aware word embeddings in the p_{th} paragraph $C_p \in \mathrm{R}^{n \times d}$ where n is the length of the paragraph, and paragraph embeddings $\mathbf{\Phi} \in \mathrm{R}^{m \times d}$ where m is the number of paragraphs. Let $X^{(\mathrm{I})} \in \mathrm{R}^{k \times d}$ denote the output of decoder part I, where k is the length of target sequence or the number of time steps. Note that both the word embedding and vocabulary in the decoder part are shared with the encoder.

Different from the token-level ranking encoding (Liu and Lapata, 2019), we intend to incorporate the information of paragraph importance to their embeddings. Specifically, ranking encoding[1] $R \in \mathrm{R}^{m \times d}$ created by the positional encoding function (Vaswani et al., 2017) are added to the original paragraph embeddings:

$$\mathbf{\Phi} := \mathbf{\Phi} + R \tag{6}$$

PHT decoder consists of three parts. Similar to a vanilla Transformer (Vaswani et al., 2017), the first and last parts of the PHT decoder are the masked multi-head attention and the feed forward network, whereas the second part includes two parallel-computing cross attention models to respectively capture the mutual information the target summary shares with source paragraphs and source words.

Paragraph-level Cross Attention. This cross attention model is to calculate the attention distribution that the decoder assigns to the paragraphs at each step, and at the same time represents the cross-paragraph relationships as paragraph-level context vectors. The query is the output of part I: $X^{(\mathrm{I})} \in \mathrm{R}^{k \times d}$ where k is the length of the target sequence. The key and value are context-aware paragraph embeddings $\mathbf{\Phi}$.

$$X_{\text{hparai}}, A_{\text{hparai}} = \text{MultiHead}\left(\boldsymbol{X}^{(\mathrm{I})}, \mathbf{\Phi}, \mathbf{\Phi}\right) \tag{7}$$

where $X^{\text{hparai}} \in \mathrm{R}^{k \times d}$ is the weighted summation of paragraph embeddings, and $A^{\text{hparai}} \in \mathrm{R}^{k \times m}$ is the paragraphs attention weights [2].

Word-level Cross Attention. This cross attention mechanism aims at modeling how the decoder attends to source tokens of the paragraph. It could be considered as the local cross attention of each paragraph since their calculations of different paragraphs are independent. By comparison, paragraph-level cross attention refers to the global cross attention which captures the dependencies among paragraphs. Since the calculations of the two cross attention are based on different encoder outputs, they are non-interfering and parallel. Finally, the mechanism produces the word-level context vectors for each paragraphs. The query of the self attention is $X^{(\mathrm{I})}$, whilst the key and value are context-aware word embeddings C_p.

$$X_p^{hwordi} = \text{MultiHead}\left(X^{(I)}, C_p, C_p\right) \tag{8}$$

where $X_p^{hwordi} \in R^{k\times d}$ denotes the word-level context vectors of all time steps in the p_{th} paragraph.

Multi-level Attention Fusion. Since the word-level cross attention model need to be implemented for each paragraph independently, there are totally m word-level context vectors X_p^{hwordi}, equivalently denoted as $X^{hwordi} \in R^{k\times d\times m}$. To fuse it with the paragraph-level context vectors $X^{hparai} \in R^{k\times d}$ and part I hidden states $X^{(I)} \in R^{k\times d}$, we need to integrate m groups of context vectors X^{hwordi} to one group. The straightforward way is to use mean pooling or max pooling, but both may cause loss of context information. An alternative approach is the adaptive attention pooling but not conductive to the computational efficiency. To handle the two problems, we directly integrate X^{hwordi} with knowledge learned by the paragraph-level cross attention model, i.e., using paragraph attention A^{hparai} to weight the context vectors X_p^{hwordi} of the corresponding paragraph. The related matrix calculation process is as follows:

$$X^{hinti} = X^{hwordi} A^{hparai}, \tag{9}$$

where $X^{hwordi} \in R^{k\times d\times m}$, $A^{hparai} \in R^{k\times m\times 1}$, and matrices are multiplied in the last two dimensions. The output of part II $X^{(II)}$ is expressed as:

$$X^{(II)} = \text{LayerNorm}\left(X^{(I)} + X^{\langle\text{para}\rangle} + X^{\langle\text{int}\rangle}\right). \tag{10}$$

With the outputs of part II, we are able to proceed to part III and compute the final probability distributions.

4. Attention-Alignment Mechanism

To further enhance the coverage of multi-document summarization, this section introduces the attentionalignment mechanism to guide the text decoding. The algorithm first predicts the optimal attention distribution of source paragraphs, then regulates the beam search according to the scoring function derived from the predicted attention distribution. Note that the attention-alignment mechanism is implemented after the training of PHT, in order to allow the extraction of the attention distribution from the trained parameters.

4.1. Learn from Neural Machine Translation (NMT)

The idea of the Attention-Alignment is inspired by Google's NMT (Wu et al., 2016), where candidates in the beam search are re-ranked according to a refined score function with the length normalization and coverage penalty. The penalty function is based on the assumption of one-to-one alignment in the translation so that $\sum_{t=1}^{T} \alpha_{t,i} = 1$, where $\alpha_{t,i}$ indicates the attention weight of the t_{th} translated word on the i_{th} source word. To penalize the situation that source words are not fully covered, i.e. the sum of attention weights is less than one, the coverage penalty is defined as:

$$cp = \sum_{i=1}^{n} \log\left(\min\left(\sum_{t=1}^{T} \alpha_{t,i}, 1\right)\right) \tag{11}$$

This assumption is not tenable for summarization as uniform coverage is no longer required. Pointergenerator (See et al., 2017) re-defines the coverage loss for summarization as:

$$cp_t = \sum_i \min\left(\alpha_{t,i}, \sum_{t'<t} \alpha_{t',i}\right. \quad ! \tag{12}$$

where $\alpha_{t,i}$ is the word-level attention distribution and $^{\text{P}}{}_{t0<t}\,\alpha_{t0,i}$ is the coverage vector. In this way, repeated attention is penalized according to the overlap between the attention distribution and the coverage til time step t.

Li et al. (2018b) further corporate this concept to their structural-coverage regularization, forcing the generation to focus on different source sentences to raise the diversity of the summary. In detail, the structural-coverage is defined as:

$$strCov(\alpha_t) = 1 - \sum_i \min \left(\alpha_{t,i}, \sum_{t'<t} \alpha_{t',i} \right. \quad ! \tag{13}$$

which is rather similar to the coverage function of Pointer-generator (See et al., 2017) except that Li et al. (2018b) consider the sentence-level attention $\alpha_{t,i}$.

In summary, both the Pointer-generator (See et al., 2017) and structural-coverage regularization (Li et al., 2018b) build their models based on the principle of searching for words/sentences that have previously attracted less attention to avoid repetition, thus to increase coverage. Comparing NMT's coverage penalty with the coverage functions in the aforementioned models (See et al., 2017; Li et al., 2018b), the restriction of summarization is rooted in the absence of the optimal attention distribution of contents, that maps a holistic layout of the summary with comprehensive coverage. This motivates us to develop the attention-alignment inference to address this matter.

4.2. Paragraph-level Attention Alignment

4.2.1. Optimal attention distribution

To explicitly express the coverage of source content, the first step of attention alignment is to use the encoded paragraph embeddings to predict the optimal attention distribution of input paragraphs. Specifically, the attention prediction model is trained from the label of paragraph-level attention distribution $\eta \in \mathrm{R}^m$ (m is the number of paragraphs) calculated from paragraph attention weights A^{hparai} in Eq. 7 and

$$A^{(para)} = \begin{bmatrix} \alpha_{1,1} & \cdots & \alpha_{1,m} \\ & \ddots & \vdots \\ \alpha_{k,1} & & \alpha_{k,m} \end{bmatrix} \tag{14}$$

where $\alpha_{t,p} \in A^{\text{hparai}}$ denotes the attention weight of the t_{th} summary word on the p_{th} source paragraph[3].

$$\eta_p = \frac{\sum_{t=1}^{k} \alpha_{t,p}}{\sum_{p=1}^{m} \sum_{t=1}^{k} \alpha_{t,p}} \tag{15}$$

$$\eta = [\eta_1, \cdots, \eta_p, \cdots, \eta_m] \tag{16}$$

Since the reference summary is known for training data, η is regarded as the optimal attention distribution and serves as the training label of the attention-prediction model. In other words, the labelling process only utilizes paragraph attention weights from the already-trained PHT parameters. Besides, the inputs of the attentionprediction model are extracted from the PHT, which are paragraph embeddings $\mathbf{\Phi}$ in Eq. 6. The training process is displayed in Figure 1.

As for the construction of the attention prediction model, paragraph embeddings are first input to a Transformer-encoder to obtain the context-aware paragraph embeddings in order to make full usage of the context information between paragraphs. The context-aware paragraph embeddings are then linearly transformed and converted to m (i.e. the number of paragraphs) units before normalized by softmax. Given the nature of the prediction is regression, mean square error (MSE) is used as the loss function.

During inference, the source paragraphs are first fed to the PHT encoder to obtain the paragraph embeddings $\mathbf{\Phi}$, based on which the trained attention-prediction model predicts the optimal attention distribution η. b

[3] In the case of multiple decoder layers, the final paragraph attention are the summation of paragraph attentions in each layer.

4.2.2. Attention alignment score

In line with NMT, the score function of the pure beam-search is modified taking into account the predicted optimal attention distribution. With length normalization and the attention-alignment score, the score of each candidate hypothesis is given by:

$$score(\boldsymbol{y}) = \frac{\log\left(P(\boldsymbol{y}|\boldsymbol{x})\right)}{|\boldsymbol{y}|} + \beta * attAlign(\boldsymbol{y}) \tag{17}$$

$$attAlign(\boldsymbol{y}) = \sum_{p=1}^{m} \log\left(\min\left(\eta_p^{(\boldsymbol{y})}, \widehat{\eta}_p\right)\right) \tag{18}$$

where *x* denotes the source, *y* refers to a candidate hypothesis, and $\widehat{\eta}_p \in \widehat{\boldsymbol{\eta}}$. Notably, different from the η_p of reference, $\eta_p^{(y)}$ is obtained from the generated candidate summary.

This predicted optimal paragraph $\widehat{\eta}_p$ attentionis compared with the paragraph attention $\eta_p^{(y)}$ in the realtime generation. Given any deviation between the real and optimal attention distributions, paragraphs that are assigned with underestimated attention place negative impacts on the overall scoring, whereas those with overestimated attention receive a constant score of $\widehat{\eta}_p$. Regarding the length normalization, we direct use the length $|y|$, rather than $length^a$ (Wu et al., 2016), as it is empirically proven to be more suitable for longer summaries.

4.2.3. Why using trained paragraph attention to form the optimal attention distribution?

An alternative way to obtain the optimal attention distribution is to use the paragraph ranking generated by an extractive model that predicts the probability each source sentence/paragraph appears in the final summary. However, the prediction of extractive probabilities is a separate unit from the summarization model which results in problematic inconsistency with the paragraph attention during the decoding process.

To support this argument with empirical evidence, Figure 2 randomly selects 10,000 training samples to compare the paragraph rankings (Liu and Lapata, 2019) with the corresponding normalized paragraph attention by the trained HT decoder. In general, the decoder assigns higher attention to paragraphs with higher rankings. However, the outliers suggest that there are several cases of different judgements by the two approaches, which lead to potential conflicts during the inference given the inconsistent measures between the optimal and real attention. Therefore, we make our prediction model to learn paragraph attention from the trained decoder directly. These attentions are considered optimal as the training targets are gold summaries. The prediction model maps the connection between source documents and optimal attention distributions, to allow the predicted attention distribution to maximally approach the optimal attention distribution if the target is unknown.

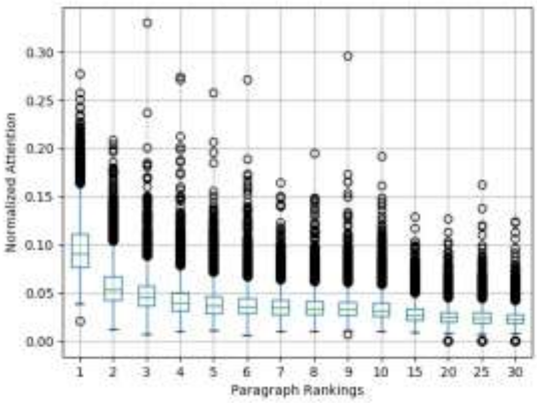

Figure 2: Box plot of paragraph attention with different initial rankings.

In addition to the inconsistency problem, it is easier to acquire the attention weight since attention exists in almost all neural abstractive summarization models, among which many (especially single-document summarization) do not require extractive probabilities. Besides, the attention-prediction model directly extracts input vectors and labels from the summarization model, whereas the extractive method requires extra effort on representing inputs and making labels.

5. Experiment Setup

5.1. WikiSum Dataset

Data sparsity has been the bottleneck of neural MDS models til WikiSum (Liu et al., 2018) came along. In this study, we use the ranked version of WikiSum provided by Liu and Lapata (2019). Each sample contains a short title, 40 ranked paragraphs with a maximum length of 100 tokens as source inputs, and a target summary with an average length of 140 tokens. Consistent with Liu and Lapata (2019), the dataset is split with 1,579,360 samples for training, 38,144 for validation and 38,205 for test. Subword tokenization (Bojanowski et al., 2017) is adopted to tokenize our vocabulary to 32,000 subword units to better represent unseen words.

5.2. Configuration

The proposed PHT is trained on a single *2080ti* with 0.3 dropout rate and an Adam optimizer of 16,000 warm-up steps. We stack 3-layer encoder-decoder of the PHT with 256 hidden units, 1024 FFN units and 4 headers, top 3000 tokens (30 paragraphs, 100 tokens) are used to train the PHT for approximately 600,000 steps. Checkpoints are saved every 20,000 steps and the best result on the validation set is used to generate the final summary. All parameters are randomly initialized including token embeddings. In the decoding process, we take 3000 tokens as inputs of PHT to generate summaries. Since the fixed positional encodings are used, so the attention-prediction model can accept inputs of dynamic length. We set the beam size to 5 and terminate the inference til the length exceeds 200. In addition, we disallow the repetition of trigrams, at the same time block two tokens (except commas) before the current step to prevent degeneration situations. For the attention prediction model, we construct a two-layer Transformer encoder with dropout rate 0.5. The complete set of the training data is used to train the attention prediction for approximately 100,000 steps.

5.3. Baselines

- **Lead** (Nenkova and McKeown, 2011) is an extractive model that extracts the top *K* tokens from the concatenated sequence. In MDS, we combine paragraphs in order and place the title at the beginning of the concatenated sequence.
- **LexRank** (Erkan and Radev, 2004) is a widely-used graph-based extractive summarizer.
- **Flat Transformer (FT)** is the vanilla Transformer encoder-decoder model (Vaswani et al., 2017). We adopt a 3-layer Transformer in this study.
- **T-DMCA** (Liu et al., 2018) is a Transformer-decoder model that splits a concatenated sequence into segments, and uses a Memory Compressed Attention to exchange information among them.
- **Transformer-XL** (Dai et al., 2019) is a language model that excels in handling excessively long sequences. This model improves the vanilla Transformer-decoder with the recurrent mechanism and relative positional encoding.
- **Liu's Hierarchical Transformer (Liu's HT)** (Liu and Lapata, 2019) uses a hierarchical structure to enrich tokens with information from other paragraphs before inputting to the Flat Transformer.
- **GraphSum** (Li et al., 2020) is an graph-based hierarchical transformer, where graph neural network is used to capture cross-document relationships.
- **Parallel Hierarchical Transformer (PHT)** is the model proposed in this paper. Different from Liu's HT, our hierarchical structure could compute token-level and paragraph-level dependencies, thus not requiring

Decoding strategy

- *PHT with vanilla beam search.* Beam search is a well-established baseline that is popularly-used in text generation tasks of all sorts.

PHT with the structural-coverage (strCov). As the original study (Li et al., 2018b) summarizes single document using a hierarchical decoding algorithm to first decode sentence by sentence then realize the sentence word by word, we need to modify the regularization to adjust to the word-by-word inference. Therefore, we re-define $\alpha_{t,i}$ (in Eq. 13) from the attention of the t_{th} generated sentence on the i_{th} source sentence to the t_{th} generated word on the i_{th} source paragraph. To obtain an independent observation on the effect of the structural coverage, we skip the structural-compression regularization and other modifications on the loss function as discussed in Li et al. (2018b).

PHT with extractive probability (extProb) also adopts attention alignment mechanism but replaces the learned optimal attention distribution η by the normalized extractive probabilities of paragraphs. We use the b

extractive method in Liu's HT to calculate these probabilities.

PHT with the attention alignment mechanism (attAlign) is PHT combined with the proposed attentionalignment mechanism. We probe the optimal value of the attention-alignment coefficient β in Eq. 17 by a numerical comparison for $\beta \in [0.2 : 0.2 : 1]$ on ROUGE. The result of development set suggests the optimal value of β is approximately 0.8 for the Wikisum dataset.

6. Results

6.1. Automatic Evaluation

In this section, we adopt a group of widely-used evaluation metrics ROUGE (Lin, 2004) to evaluate the MDS models. ROUGE-1 & -2 and ROUGE-L F_1 scores are reported in Table 1 assessing the informativeness and fluency of the summaries, respectively.

Model	ROUGE-1	ROUGE-2	ROUGE-L
Lead	36.40	16.66	32.95
FT	40.30	18.67	32.84
T-DMCA	41.09	19.78	33.31
Transformer-XL	41.11	19.81	33.72
Liu's HT	40.83	19.41	33.26
1-layer PHT	41.02	19.82	33.28
PHT	**41.99**	**20.44**	**34.50**

Table 1: Average ROUGE F_1 scores of different summarization models.

As shown in Table 1, the extractive model Lead exhibits overall inferior performance in comparison to the abstractive models, except that it produces a 0.11-higher ROUGE-L than the Flat Transformer. Although Liu's HT improves FT with a hierarchical structure, it fails to outperform the two extended flat models, i.e. T-DMCA and Transformer-XL, that are developed to learn lengthier inputs. Moreover, T-DMCA and Transformer-XL report comparable results in terms of the informativeness (ROUGE-1 & -2), whilst the latter outperforms the former by 0.41 in terms of the fluency (ROUGE-L).

Further, the proposed PHT model shows promising ROUGE results. Benefited from the pure hierarchical structure that allows prolonged token inputs, PHT outperforms Liu's HT in all domains of the ROUGE test. Moreover, the models' potential to be deepened is suggested by enhanced results of the 3-layer architecture over the 1-layer architecture. The ultimate 3-layer PHT stably surpasses T-DMCA and Transformer-XL, that are also tailored to handle long input sequences of 3,000 tokens, due to its hierarchical processing of token and document-level information.

6.1.1. Comparing the decoding strategies

Parallel HT	ROUGE-1	ROUGE-2	ROUGE-L
+ vanilla beam search	41.99	20.44	34.50
+ strCov (Li et al., 2018b)	41.74	20.25	33.88
+ extProb	42.17	20.46	34.79
+ attAlign	**42.58**	**20.84**	**35.66**

Table 2: Average ROUGE F_1 scores of different decoding strategies.

Table 2 shows the average ROUGE F_1 scores of all model combinations investigated. The attention-alignment mechanism promotes the quality of summaries by raising ROUGE-1 by 0.59, ROUGE-2 by 0.4, ROUGE-L by 1.16 for PHT with beam search. Technically, the attention alignment mechanism could be applied to all hierarchical models with an attention mechanism. Further, Table 2 provides empirical evidence to Section 4.2.3, suggesting that extractive probabilities (extProb) are not as good protocols as the paragraph attention for the optimal attention distribution, given the marked decline in the ROUGE scores in comparison to the proposed attention-alignment mechanism.

Besides, the structural-coverage mechanism hinders the performance of MDS with reduced ROUGE scores. We print its beam-search scores and find consecutive zeros as the generated sequences get longer, resulted from the t_{th} word attention on the i_{th} paragraph ($\alpha_{t,i}$) remains lower than the its cumulative attention (${}^{\mathrm{P}}{}_{t0<t}$ $\alpha_{t0,i}$). Therefore, it is concluded that the structural coverage regularization is not particularly suitable for word-byword summarization with lengthy inputs, that come along with multiple documents.

6.2. Human Evaluation

To provide a better comparison between the MDS models, we select 4 representative summarization models with the best ROUGE performances in the human evaluation, namely T-DMCA & Transformer-XL (flat structure), and Liu's HT & PHT (hierarchical structure), and two decoding strategies including the original Beam search and attention alignment.

In the survey, multi-document summaries are scored from four perspectives, including (A) **Informativeness** (Does the summary include important information in the gold summary), (B) **Fluency** (Is the summary fluent and grammatically-correct), (C) **Conciseness** (Does the summary avoid repetition and redundancy), (D) **Factual consistency** (Does the summary avoid common sense mistakes such as wrong date, wrong location, or anything else against facts). We specify five ratings from *Very poor* (1) to *Very good* (5) to assess criteria (A)-(C), and three ratings of *Much better* (2), *Better* (1), and *Hard to score* (0) to assess criterion (D). Twenty examples are randomly selected from generated summaries. Fifteen human evaluators participated in the experiment.

The results are displayed in Table 6.2. The general observation is twofold. A) Discrepancy exists in the ROUGE and human evaluations. For instance, T-DMCA tends to yield higher human scores relative to Transformer-XL although ROUGE suggests the opposite. Given the merits and weaknesses of the different metrics, we focus on discussing results that exhibit consistency in different parts of evaluation. B) Given the lowest average mark, factual consistency appears to be the bottleneck of abstractive summarization models that hinders human experience on the machine generated summaries.

As far as the summarization models are concerned, PHT achieves the highest human evaluation scores in all four areas investigated. On the other hand, the other hierarchical baseline, Liu's HT, turns to be less competitive than the flat structures in terms of informativeness, conciseness and factual consistency, possibly due to its length limit of input. With regards to the optimal attention distribution of summaries, attention-alignment is proven effective in improving the hierarchical model.

7. Analysis

This section discusses the experimental results obtained from Wikisum using different baseline models. Through the preliminary analysis on PHT, we intend to obtain an initial understanding of the hierarchical model on its capacity to better express the cross-document relationship, as well as the associated computational cost. The improvements on the summary quality by PHT with attention alignment is then investigated through the ROUGE analysis and human evaluation.

Model	Informativeness	Fluency	Conciseness	Factual consistency
T-DMCA	3.69	3.66	3.82	3.04
Transformer-XL	3.57	3.71	3.77	2.88
Liu's HT	3.34	3.76	3.75	2.82
PHT	4.11	3.97	3.81	3.11
PHT with attAlign	**4.39**	**4.22**	**3.95**	**3.35**
Average	3.77	3.89	3.84	3.05

Table 3: Human evaluation results.

7.1. Preliminary Analysis on PHT

7.1.1. The cross-document relationship

Cross-document relationships could be reflected by the distribution of paragraph attentions. If a model assigns higher attention weights to more important paragraphs and vice versa, the model is believed to have greater capacity of capturing cross-document relationships. To analytically assess the models' performance in this aspect, we use paragraph attentions of reference summaries as the gold attention distribution, and its cosine similarity to the attention distribution of generated summaries as the evaluation metric. To model the paragraph attention of the reference, we compute the normalized tf-idf similarities between the gold summary and each input paragraph as the gold attention distribution. For the baseline models, the summation of token weights in each paragraph are computed to indicate each paragraph's attention, whilst PHT returns the paragraph attention distribution directly from its paragraph-level multi-head attention.

Model	Cosine similarity
Lead	0.8098
Flat Transformer	0.8143
T-DMCA	0.8654
Transformer-XL	0.8447
Liu's HT	0.8769
PHT	**0.8936**

Table 4: Average cosine similarities between attention distributions of generated summaries and the reference.

As suggested by Table 4, hierarchical structures place significant improvements on the flat models in learning cross-document dependencies by assigning paragraph attentions in a way that is closer to the gold summaries. Moreover, PHT generates summaries of the greatest similarity 89.36% with the gold summaries, most likely due to its paragraph-level multi-head attention in addition to the token-level one, allowing the exchanging of cross-document information.

7.1.2. Computational efficiency

This section assesses the computational efficiency of PHT comparing to other neural abstractive models in three aspects, i.e. the memory usage, parameter size and validation speed. We uniformly hire the 3-layer architecture and 1600 input tokens in this part to ensure fairness. During the experiment, we increase the batch size until out of memory in a 2080ti GPU, and the model with the maximum batch size occupies the lowest memory space. To measure the parameter size, we count the number of parameters in the neural network. Finally, we run each trained model in the validation set (38,144 samples), and the average time consumed in each checkpoint is used to evaluate the speed of forward-propagating in the model.

As indicated by higher batch sizes in Table 5, models in the hierarchical structure (second panel) appears to be overall more memory-saving than those in the flat structure (first panel), with higher requirements on the parameters. In particular, models based on the Transformer-decoder, i.e. T-DMCA and Transformer-XL, demonstrate absolute superiority in reducing the parameter size. As for the speed of forward-propagating, Transformer-XL dominates due to its recurrent mechanism, whereas others share close performance in the inference speed. Between the two hierarchical models, PHT is proven to outperform Liu's HT in all three aspects, due to its parallel, rather than sequential, computation of the word & paragraph-level attention mechanisms.

Model	Max Batch Size	Parameters (MB)	Validation Speed (s)
Flat Transformer	11	165.0	634
T-DMCA	10	131.1	656
Transformer-XL	8	**130.4**	**489**
Liu's HT	11	190.8	639
PHT	**17**	182.4	601

Table 5: Computational efficiency.

As an extension on attention alignment to explore its potential application on summary compression, we further argue that the algorithm provides an easy way to compress source paragraphs before inference – by ranking paragraph attention weights according to their predicted values and choosing the top *s* paragraphs with highest attentions. Table 6 presents the results given different values of *s*. According to the ROUGERecall and ROUGE-F_1 scores, the tested compression mechanism improves original summaries by limiting the number of paragraphs to [20 : 25].

s	ROUGE-1			ROUGE-2			ROUGE-L		
	F_1	P	R	F_1	P	R	F_1	P	R
5	-2.1	-3.3	-1.2	-2.0	-2.5	-1.8	-2.1	-3.8	-1.5
10	-1.2	-3.0	-0.2	-1.2	-2.2	-0.9	-1.3	-3.4	-0.5
15	-0.4	-2.6	**+0.7**	-0.5	-1.9	0	-0.1	-2.7	**+0.5**
20	0	-1.5	**+0.7**	-0.1	-1.2	**+0.2**	**+0.2**	-1.9	**+0.5**
25	**+0.1**	-0.9	**+0.6**	-0.1	-0.8	**+0.3**	**+0.1**	-1.0	**+0.5**
30	42.6	58.6	37.8	20.8	30.6	18.5	35.7	55.6	35.9

Table 6: Average ROUGE scores with different *s*, No compression when *s* = 30. F_1: F_1 score, *R*: Recall, *P*: Precision.

8. Conclusion

This study develops a Parallel Hierarchical Transformer with attention alignment inference for multidocument summarization. Using the Wikisum dataset, we empirically show that the proposed hierarchical architecture with token- and paragraph-level multi-head attentions excels in capturing the cross-document relationship of lengthy sources, and generates summaries of greater quality than other existing Transformer-based models. Further, the paragraph-level attention-alignment algorithm is designed to address the coverage issue by predicting the optimal attention distribution according to the multi-document sources. In theory, the decoding strategy has the potential to accommodate all seq2seq summarization models in the presence of the attention mechanism. Our experiment shows that attention alignment places significant improvements on the summaries generated by the original beam search.

Given the fact that the attention mechanism is nowadays almost a necessity in the seq2seq architecture, the authors target at investigating the capacity of word-level attention-alignment in the future study. The application of word-level attention alignment is no longer confined to the hierarchical architecture and can be adopted to all attention-based seq2seq models including the pre-trained model BART (Lewis et al., 2020). Different from paragraph-level attention alignment, word-level attention alignment though requires the processing of numerous attention units, bringing challenges in obtaining the optimal attention distribution where a dynamic scoring function might be developed for text decoding.

9. Acknowledgements

This work was supported by Xi'an Jiaotong-Liverpool University Key Programme Special Fund KSF-A-14.

References

Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.. Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics 2017;5:135–146. URL: https://www.aclweb.org/anthology/Q17-1010. doi:10.1162/ tacl_a_00051.

Carbonell, J., Goldstein, J.. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New York, NY, USA: ACM; SIGIR '98; 1998. p. 335–336. URL: http://doi.acm.org/10.1145/290941.291025. doi:10.1145/ 290941.291025.

Chen, Y.C., Bansal, M.. Fast abstractive summarization with reinforce-selected sentence rewriting. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) 2018;URL: http://dx.doi.org/10.18653/v1/P18-1063. doi:10.18653/v1/p18-1063.

Cohan, A., Dernoncourt, F., Kim, D.S., Bui, T., Kim, S., Chang, W., Goharian, N.. A discourse-aware attention model for abstractive summarization of long documents. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers) 2018;URL: http://dx.doi.org/10.18653/v1/N18-2097. doi:10.18653/v1/n18-2097.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q., Salakhutdinov, R.. Transformer-xl: Attentive language models beyond a fixed-length context. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics 2019;URL: http://dx.doi.org/10.18653/v1/p19-1285. doi:10.18653/v1/p19-1285.

Devlin, J., Chang, M.W., Lee, K., Toutanova, K.. Bert: Pre-training of deep bidirectional transformers for language understanding. 2018. arXiv:1810.04805.

Erkan, G., Radev, D.R.. Lexrank: Graph-based lexical centrality as salience in text summarization. J Artif Int Res 2004;22(1):457–479.

Fabbri, A., Li, I., She, T., Li, S., Radev, D.. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: Association for Computational Linguistics; 2019. p. 1074–1084. URL: https://www.aclweb.org/anthology/P19-1102. doi:10.18653/v1/P19-1102.

Gehrmann, S., Deng, Y., Rush, A.. Bottom-up abstractive summarization. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium: Association for Computational Linguistics; 2018. p. 4098–4109. URL: https://www.aclweb.org/anthology/D18-1443. doi:10.18653/v1/D18-1443.

Holtzman, A., Buys, J., Forbes, M., Choi, Y.. The curious case of neural text degeneration. 2019. arXiv:1904.09751.

Hua, X., Wang, L.. Sentence-level content planning and style specification for neural text generation. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) 2019;URL: http://dx.doi.org/10.18653/v1/d19-1055. doi:10.18653/v1/d19-1055.

Lebanoff, L., Song, K., Liu, F.. Adapting the neural encoder-decoder framework from single to multi-document summarization. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium: Association for Computational Linguistics; 2018. p. 4131–4141. URL: https://www.aclweb.org/anthology/D18-1446. doi:10.18653/v1/D18-1446.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L.. 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. Online: Association for Computational Linguistics; 2020. p. 7871–7880. URL: https://www.aclweb.org/anthology/2020.acl-main.703. doi:10.18653/v1/2020.acl-main.703.

Li, C., Xu, W., Li, S., Gao, S.. Guiding generation for abstractive text summarization based on key information guide network. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers). New Orleans, Louisiana: Association for Computational Linguistics; 2018a. p. 55–60. URL: https://www.aclweb.org/anthology/N18-2009. doi:10.18653/v1/N18-2009.

Li, W., Xiao, X., Liu, J., Wu, H., Wang, H., Du, J.. Leveraging graph to improve abstractive multi-document summarization. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Online: Association for Computational Linguistics; 2020. p. 6232–6243. URL: https://www.aclweb.org/anthology/2020.acl-main.555. doi:10.18653/v1/2020.acl-main.555.

Li, W., Xiao, X., Lyu, Y., Wang, Y.. Improving neural abstractive document summarization with structural regularization. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium: Association for Computational Linguistics; 2018b. p. 4078–4087. URL: https://www.aclweb.org/anthology/D18-1441. doi:10.18653/v1/D18-1441.

Liang, Z., Du, J., Shao, Y., Ji, H.. Gated graph neural attention networks for abstractive summarization. Neurocomputing 2021;431:128–136. URL: https://www.sciencedirect.com/science/article/pii/S0925231220315940. doi:https://doi.org/10.1016/j.neucom.2020.09.066.

Liao, W., Ma, Y., Yin, Y., Ye, G., Zuo, D.. Improving abstractive summarization based on dynamic residual network with reinforce dependency. Neurocomputing 2021;448:228–237. URL: https://www.sciencedirect.com/science/article/pii/ S0925231221002745. doi:https://doi.org/10.1016/j.neucom.2021.02.028.

Lin, C.Y.. ROUGE: A package for automatic evaluation of summaries. In: Text Summarization Branches Out. Barcelona, Spain: Association for Computational Linguistics; 2004. p. 74–81. URL: https://www.aclweb.org/anthology/W04-1013.

Liu, P.J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L., Shazeer, N.. Generating wikipedia by summarizing long sequences. 2018. arXiv:1801.10198.

Liu, Y., Lapata, M.. Hierarchical transformers for multi-document summarization. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics 2019;URL: http://dx.doi.org/10.18653/v1/p19-1500. doi:10.18653/v1/p19-1500.

Ma, C., Zhang, W.E., Guo, M., Wang, H., Sheng, Q.Z.. Multi-document summarization via deep learning techniques: A survey. 2020. arXiv:2011.04843.

Mei, H., Bansal, M., Walter, M.R.. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. Computer Science 2015;.

Nenkova, A., McKeown, K.. Automatic summarization. Now Publishers Inc, 2011.

Pasunuru, R., Bansal, M.. Multi-reward reinforced summarization with saliency and entailment. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers) 2018;URL: http://dx.doi.org/10.18653/v1/N18-2102. doi:10.18653/v1/n18-2102.

Pradhan, T., Bhatia, C., Kumar, P., Pal, S.. A deep neural architecture based meta-review generation and final decision prediction of a scholarly article. Neurocomputing 2021;428:218–238. URL: https://www.sciencedirect.com/science/article/ pii/S0925231220317318. doi:https://doi.org/10.1016/j.neucom.2020.11.004.

Rush, A.M., Chopra, S., Weston, J.. A neural attention model for abstractive sentence summarization. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Lisbon, Portugal: Association for Computational Linguistics; 2015. p. 379–389. URL: https://www.aclweb.org/anthology/D15-1044. doi:10.18653/v1/D15-1044.

See, A., Liu, P.J., Manning, C.D.. Get to the point: Summarization with pointer-generator networks. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) 2017;URL: http://dx.doi.org/10. 18653/v1/p17-1099. doi:10.18653/v1/p17-1099.

Sutskever, I., Vinyals, O., Le, Q.V.. Sequence to sequence learning with neural networks. In: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2. Cambridge, MA, USA: MIT Press; NIPS'14; 2014. p. 3104–3112.

Tan, J., Wan, X., Xiao, J.. Abstractive document summarization with a graph-based attentional neural model. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vancouver, Canada: Association for Computational Linguistics; 2017. p. 1171–1181. URL: https://www.aclweb.org/anthology/P17-1108. doi:10. 18653/v1/P17-1108.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.. Attention is all you need. 2017. arXiv:1706.03762.

Welleck, S., Kulikov, I., Roller, S., Dinan, E., Cho, K., Weston, J.. Neural text generation with unlikelihood training. 2019. arXiv:1908.04319.

Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Dean, J.. Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv 2016;.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E.. Hierarchical attention networks for document classification. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego, California: Association for Computational Linguistics; 2016. p. 1480–1489. URL: https://www.aclweb.org/anthology/N16-1174. doi:10.18653/v1/N16-1174.

Yao, K., Zhang, L., Luo, T., Wu, Y.. Deep reinforcement learning for extractive document summarization. Neurocomputing 2018;284:52–62. URL: https://www.sciencedirect.com/science/article/pii/S0925231218300377. doi:https://doi.org/10. 1016/j.neucom.2018.01.020.

You, Y., Jia, W., Liu, T., Yang, W.. Improving abstractive document summarization with salient information modeling. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019. p. 2132–2141.

Zhang, X., Wei, F., Zhou, M.. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: Association for Computational Linguistics; 2019. p. 5059–5069. URL: https://www.aclweb.org/anthology/P19-1499. doi:10.18653/v1/P19-1499.