Improving Transformer with Sequential Context Representations for Abstractive Text Summarization *

Tian Cai[1,2], Mengjun Shen[1,2], Huailiang Peng[1,2], Lei Jiang[1], and Qiong Dai[1]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{caitian, shenmengjun, penghuailiang, jianglei, daiqiong}@iie.ac.cn

Abstract. Recent dominant approaches for abstractive text summarization are mainly RNN-based encoder-decoder framework, these methods usually suffer from the poor semantic representations for long sequences. In this paper, we propose a new abstractive summarization model, called RC-Transformer (RCT). The model is not only capable of learning long-term dependencies, but also addresses the inherent shortcoming of Transformer on insensitivity to word order information. We extend the Transformer with an additional RNN-based encoder to capture the sequential context representations. In order to extract salient information effectively, we further construct a convolution module to filter the sequential context with local importance. The experimental results on Gigaword and DUC-2004 datasets show that our proposed model achieves the state-of-the-art performance, even without introducing external information. In addition, our model also owns an advantage in speed over the RNN-based models.

Keywords: Transformer · Abstractive summarization.

1 Introduction

Automatic text summarization is the process of generating brief summaries from input documents. Having the short summaries, the text content can be retrieved effectively and easy to understand. There are two main text summarization techniques: extractive and abstractive. Extractive models [6] extract salient parts of the source document. Abstractive models [10] restructure sentences and may rewrite the original text segments using new words. As the abstractive summarization is more flexible and the generated summaries have a good matching with human-written summaries, we focus on abstractive text summarization.

Recently, most prevalent approaches for abstractive text summarization adopt the recurrent neural network (RNN)-based encoder-decoder framework with attention mechanism [7, 8]. The encoder aims to map the source article to a vector representation and the decoder generates a summary sequentially on the basis of the representation. The

* Corresponding author: Qiong Dai. This paper is Supported by National Key Research and Development Program of China under Grant No.2017YFB0803003 and National Science Foundation for Young Scientists of China (Grant No. 61702507)

encoder and the decoder are both based on the RNN structure, such as long-short-term memory (LSTM) and gated recurrent unit (GRU).

However, the training of RNN-based sequence-to-sequence(seq2seq) models is slow due to their inherent sequential dependence nature. Another critical problem of RNN-based models is that they can not capture distant dependency relationships for long sequences. Vaswani *et al.* [16] construct a novel encoder-decoder architecture with strong attention, namely Transformer, which is capable of learning long-term dependencies and has advanced the state-of-the-art on machine translation.

The Transformer has demonstrated to be effective for capturing the global contextual semantic relationships and parallel computing. The self-attention mechanism is able to learn the "word-pair" relevance. The word order information is accessed by positional encoding. However, for the reason that position information is important in natural language understanding, the positional encoding is only approximate to sequence information. Therefore, there is a practical demand for modeling word-level sequential context for the source article.

Motivated by the above observations, we propose a novel abstractive summarization model, called RC-Transformer, which improves Transformer with sequential context representations. The proposed architecture consists of two encoders and a decoder. We decouple the responsibilities of the encoder of capturing contextual semantic representations and modeling sequential context by introducing an additional RNN-based encoder. Since the local correlations contribute to learning syntactic information, we further construct a convolution module to capture different n-gram features. The salient information can be focused by filtering the sequential context with the local importance. Furthermore, we introduce lexical shortcuts to improve the semantic representations both in Transformer encoder and decoder.

We experimentally validate the effectiveness of our method for abstractive sentence summarization. Our RC-Transformer achieves the state-of-the-art performance and is able to generate high quality summaries, even without the external knowledge guidance. Moreover, in spite of introducing a RNN-based encoder, our RC-Transformer is also superior to the RNN-based seq2seq model in speed.

2 Related Work

2.1 Abstractive Text summarization

Abstractive text summarization has received much attention in recent years since the seq2seq model was developed. Many neural network based models have achieved great performance over conventional methods. Rush *et al.* [10] introduce a RNN-based seq2seq model with attention to generate summaries. In addition, intra-temporal and intra-decoder attention mechanisms are proposed to overcome repetitions and reinforce algorithm has also been used to avoid the exposure bias [8]. For sentence summarization, Zhou *et al.*[20] introduce a selective gate network to filter secondary information and Shen *et al.* [13] optimize model at sentence-level to improve the ROUGE score.

All the abstractive summarization models mentioned above are based on RNNs. There are two notable problems with these models: (1) the sequential nature of RNN

prevents the computation in parallel. (2) Suffering from the difficulty of learning long-term dependencies, RNNs are limited to model relatively short sequences. However, the input articles are always long text in text summarization, there is a bottleneck to improve the performance of the RNN-based models.

Recently several encoder-decoder architectures, such as Convs2s [3] and Transformer [16] are exploited. For abstractive text summarization, Wang *et al.* [18] propose a convolutional seq2seq model which incorporates the topic information and achieves good performance. Liu *et al.* [5] alter the Transformer decoder to a language model to create Wikipedia articles from several reference articles.

2.2 Transformers

Although Transformers are effective in machine translation, for abstractive text summarization, this architecture does not behave well for its poor ability of modeling the word-level sequential context. Recently there are some related work about modifying positional encoding. Shaw *et al.* [12] extend the self-attention mechanism to efficiently consider representations of the relative positions. Takase *et al.* [15] propose an extension of sinusoidal positional encoding to control output sequence length. But neither of them is a complete strategy to tackle the insensitivity to sequential information for Transformer. In this paper, we introduce an additional encoder based on RNN to alleviate the problem in Transformer.

3 The Proposed Model

In this section, we describe (1) the problem formulation and our base model Transformer, (2) our proposed model, called RC-Transformer, which introduces an additional encoder with a bidirectional RNN to model sequential context and a convolution module to capture local importance.

3.1 Background

Based on the strong ability of learning the global contextual representation, we use the Transformer[16] model as our baseline. Formally, let $X = \{x_1, \cdots, x_m\}$ denote the source article with m words and $Y = \{y_1, \cdots, y_n\}$ denote the output sequence of n summary words.

The Transformer follows an encoder-decoder architecture. The encoder consists of a stack of N layers, each of them composes of two sub-layers: a multi-head self-attention mechanism and a fully-connected feed forward network. The self-attention is defined as:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

The encoder maps the input article into a sequence of continuous representation $Z = \{z_1, \cdots, z_m\}$. The decoder performs encoder-decoder attention to learn the correlation between the source text and the generated text.

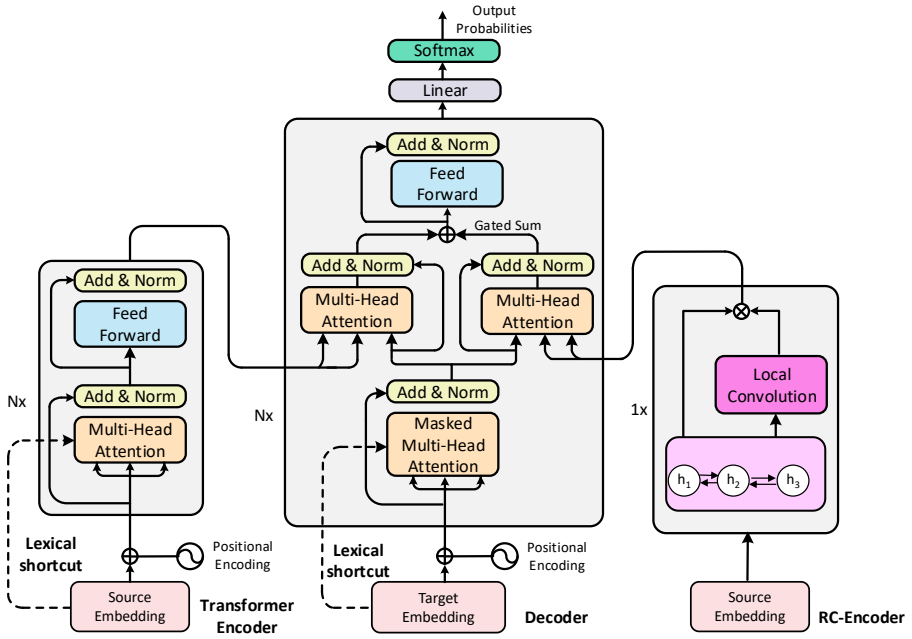


Fig. 1: An overview of RC-Transformer which has two encoders(left and right) and a decoder(middle). The model has a similar structure with Transformer [16]. We introduce an additional encoder called RC-Encoder with a Bi-RNN to model sequential context and a convolution module upon RNN to filter the sequential context with local importance via a gated unit. A lexical shortcut is employed between each layer and the embedding layer in both Transformer encoder and decoder.

3.2 RC-Transformer

Although the positional encoding retains the order information, the model is still not quite sensitive to the word order which is crucial in abstractive text summarization. The lack of the word-level sequential information limits the model's ability of natural language understanding in depth. The generated summaries often incorporate much non-salient information. To alleviate the problem, we propose a RC-Transformer model which decouples the encoder's responsibilities of learning contextual semantic representation and capturing the word order information by factoring it into two encoders. A RC-Encoder based on RNN is introduced to assist in learning the word-level sequential context. Upon the RNN structure, a convolutional module is applied to filter the sequential context with local importance. Our RC-Transformer contains three major components: a RC-Encoder, a Transformer encoder and a decoder. The graphical illustration of the RC-Transformer is shown in Fig. 1. We introduce the RC-Encoder and decoder in detail in this section.

RC-Encoder The encoder first maps the source text into a sequence of hidden states $H = \{h_1, \cdots, h_m\}$ via a bidirectional LSTM.

$$H = BiLSTM(E) \in \mathbb{R}^{m \times d_{hid}} \tag{2}$$

where E is the embedding representation of the source article X and d_{hid} is the output dimension. At each time step, the output is the concatenation of two directional hidden states ($h_i = \left[\overrightarrow{h_i}, \overleftarrow{h_i}\right]$). Since RNNs possess good capacity in modeling the word sequence, hence H represents the sequential context of the source article. Furthermore, the syntactic information can be captured by n-gram features, we also extract different n-gram features of the source article on the basis of the Bi-RNN.

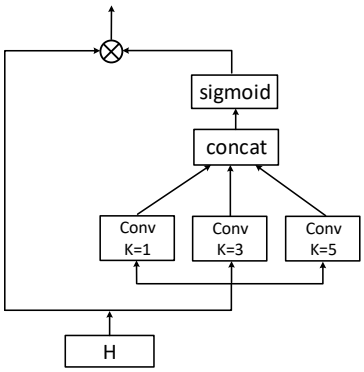


Fig. 2: An illustration of local convolution module with gated linear unit.

Local Convolution We further enhance the sequential context representation with a convolutional module. We implement a convolution module of different receptive fields to learn n-gram features with different sizes. Given the input hidden states sequence H, three convolution operations are applied to obtain three output vectors $D_{k=1}, D_{k=3}, D_{k=5}$, where k is the kernel size. We concatenate the three outputs to take different n-gram features into account.

$$D = \left[D_{k=1}, D_{k=3}, D_{k=5}\right]. \tag{3}$$

Instead of taking D as the output of the RC-Encoder, we set a learnable threshold mechanism to filter the sequential context according to the local importance. The gated linear unit (GLU) controls information flow by selecting features through a sigmoid function, which is demonstrated to be useful for language modeling [2]. We introduce a similar architecture(see Fig. 2) to select how much sequential context information should be retained as:

$$R = \sigma\left(W_d D + b_d\right) \odot \left(W_h H + b_h\right). \tag{4}$$

The RC-Encoder assists the original encoder in modeling the word order information and learning local interactions. We leave the Transformer encoder as it is to capture the global semantic representation.

Decoder The model encodes the source text into a global semantic representation and a sequential context representation. Then the two representations are integrated in the decoder to generates summaries. As shown in Fig. 1, we follow [16] to use a stack of N decoder layers to compute the target-side representations. Each layer is composed of four sub-layers. Specifically, we employ two encoder-decoder attention sub-layers, each of which perform an attention between the encoder representation and the decoder representation. More precisely, let $C^{(n)}$ be the output of the masked multi-head self-attention at the n-th decoder layer, then the two encoder-decoder self-attention sub-layers calculate two representations:

$$T_R^{(n)} = MultiHead(C^{(n)}; R; R), \tag{5}$$

$$T_Z^{(n)} = MultiHead(C^{(n)}Z; Z). \tag{6}$$

The outputs of the two attention mechanisms are combined via a gated sum.

$$g = \sigma\left(W_g\left[T_R^{(n)}, T_Z^{(n)}\right] + b_g\right), \tag{7}$$

$$S^{(n)} = g \odot T_R^{(n)} + (1-g) \odot T_Z^{(n)}. \tag{8}$$

Subsequently, the output $S^{(n)}$ is fed to the feed forward layer. In the previous works, there are two other strategies for integrating two encoders and a decoder architecture, called "Gated Sum in Encoder" and "Stacked in Decoder". We elaborate these two strategies in Section 4.4 and conduct experiments to demonstrate that our method performs better than the two strategies in this case.

3.3 Lexical Shortcuts

Within the Transformer encoder and decoder, each sub-layer takes the output of the immediately preceding layer as input. The lexical features are learned and propagated upward from the bottom of the model. For the higher-level layer to learn the semantic representation, the lexical features must be retained in the intermediate representation. Therefore, the model is unable to fully leverage its capacity of capturing semantic representations. To alleviate the problem, we add a gated connection called lexical shortcut between the embedding layer and each subsequent self-attention sub-layer within the encoder and decoder (see Fig. 1).

In each self-attention sub-layer, the K,V vectors are recalculated to carry part of lexical features. A transform gate aims to select how much lexical features should be carried in each dimension. Take K for illustration:

$$T_l^K = \sigma\left(W_k\left[E, K_l\right]\right), \tag{9}$$

then the current features and the lexical features are combined by calculating their weighted sum.

$$K_l^{new} = E \odot T_l^K + K_l \odot \left(1 - T_l^K\right) \tag{10}$$

The equation 1 utilize the new K and V vectors to calculate the self-attention. This method enhances the semantic representations by exposing lexical content and position information to the following layers.

4 Experiments

4.1 Datasets and Evaluation Metrics

We evaluate our methodology on English Gigaword and DUC-2004 datasets which are the standard benchmark datasets for abstractive text summarization. The English Gigaword is a sentence summarization dataset. We follow the experimental settings in [10] to preprocess the corpus. The extracted corpus contains about 3.8M samples for training, 8K for validation and 0.7K for testing. Each sample in the dataset is a sentence pair, which consists of the first sentence of the source articles and the corresponding headline. The DUC-2004 dataset is a summarization evaluation set which consists of 500 news articles. Each article in the dataset is paired with four human-written reference summaries. Compared to [10] tuning on DUC-2003, we directly use the model trained on the Gigaword to test on the DUC-2004 corpus.

We employ ROUGE [4] as our evaluation metric. ROUGE measures the quality of summary by computing the overlapping lexical units between the generated summaries and the reference summaries. Following the previous work, we report full-length F-1 scores of ROUGE-1, ROUGE-2 and ROUGE-L metrics.

4.2 Implementation Details

We implement our experiments in PyTorch on 4 NVIDIA TITAN X GPUs. In preprocessing, we use the Byte pair encoding (BPE) algorithm [11] to segment words. We set the hyper-parameter to fit the vocabulary size to 15,000. The baseline Transformer model is trained with the same hyper-paremeters as in the base model in [16]. And our extended RC-Transformer model uses 8 attention heads and a dimension of 1024 for the feed forward network. We set the Transformer encoder and decoder layer number as 4. Moreover, the RC-Encoder is implemented with a two-layer bidirectional GRU. For convolution module, we employ three convolution layers with kernel size $1, 3, 5$ respectively and we keep the same output size of each convolution operation with padding size $1, 3, 5$. In training, cross entropy is used as the loss function and label smoothing is introduced to reduce overfitting. Each model variants are trained approximately 5 epochs. During test, we use beam search of size 5 to generates summaries and limit the maximum output length as 15 and 20 for Gigaword and DUC2004 dataset respectively.

4.3 Comparison with State-of-the-Art Methods

In addition to the base model Transformer, we also introduce the following state-of-the-art baselines to compare the effect of our approach. **ABS and ABS+** [10] are both the RNN-based seq2seq models with local attention. The difference is that ABS+ extracts additional hand-crafted features to revise the output of ABS model. **RAS-LSTM** [14] model introduces a convolutional attention-based encoder and a RNN decoder. **SEASS** [20] extends the seq2seq model with a selective gate mechanism. **DRGD** [9] is a seq2seq model equipped with a deep recurrent generative model. **RNN+MRT** [9] employs the minimum risk training strategy which directly optimizes model parameters in

Table 1: Comparisons with the state-of-the-art methods on abstractive text summarization benchmarks.

	Gigaword			DUC-2004		
Models	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**
ABS	29.55	11.32	26.42	26.55	7.06	22.05
ABS+	29.76	11.88	26.96	28.18	8.49	23.81
RAS-LSTM	32.6	14.7	30.0	28.97	8.26	24.06
SEASS	36.2	17.5	33.6	29.21	9.56	25.51
DRGD	36.3	17.6	33.6	31.79	10.75	27.48
RNN+MRT	36.54	16.59	33.44	30.41	10.87	26.79
ConvS2S	35.88	17.48	33.29	30.44	10.84	26.90
using external information guidance						
Feats	32.7	15.6	30.6	28.61	9.42	25.24
RL-Topic-ConvS2S	36.92	18.29	34.58	31.15	10.85	27.68
Re^{3}Sum	37.04	**19.03**	34.46	-	-	-
our methods						
Transformer	35.96	17.11	33.46	28.62	9.95	25.62
RCT	**37.27**	18.19	**34.62**	**33.16**	**14.7**	**30.52**

the sentence level with respect to the evaluation metrics. **ConvS2S** [3] is a convolutional seq2seq model.

We also compare our model with several state-of-the-art methods utilizing external information to guide the summaries generating. **FeatS2S** [7] uses a full RNN-based seq2seq model which enhances the encoder by adding some hand-crafted features such as POS tag and NER. **RL-Topic-ConvS2S** [18] is a convolutional seq2seq model training with reinforcement learning objective and jointly attends to topics and word-level alignment to improve performance. **Re^{3}Sum** model [1] proposes to use existing summaries as soft templates to guide the seq2seq model.

As shown in Table 1, our approach achieves significant improvements over the current baseline, bettering RNN+MRT model by an absolute 2% and 8% increase in ROUGE-1 F1 score on the Gigaword and DUC2004 dataset respectively. We also compare our model with Feats, RL-Topic-ConvS2S and Re^{3}Sum. We can see that even without introducing external information and the REINFORCE, our model still performs better. It shows that considering the sequential context representation and global semantic representation, our model is able to capture salient information and generate high quality summaries.

4.4 Comparison with Different Integration Strategies

In this section, we introduce different integration strategies for two encoders and one decoder architecture. Voita *et al.* [17] introduce a context-aware neural machine translation model where the decoder keeps intact while incorporating context information on the encoder side. Zhang *et al.* [19] employ a new context encoder which is then incorporated into both the original encoder and decoder with a context attention stacked on the self-attention sub-layer. We conclude the two strategies as below:

Table 2: Effect of different integration strategies of the two encoders and one decoder architecture.

Models	ROUGE-1	ROUGE-2	ROUGE-L
Gated Sum in Encoder	36.13	17.09	33.51
Stacked in Decoder	36.33	17.28	33.63
Our Methods	**37.27**	**18.19**	**34.62**

Gated Sum in Encoder: Integrate the output representations of the two encoders on the encoder side by combining the two representations via a gated sum.

Stacked in Decoder: Integrate the output representations of the two encoders into the decoder by employing two encoder-decoder attention sub-layers stacked with the original layers.

We conduct experiments to verify the performance of the two strategies and our method. As shown in Table 2, it is clear that our method that combines the two encoder-decoder attention outputs with a gated sum is effective.

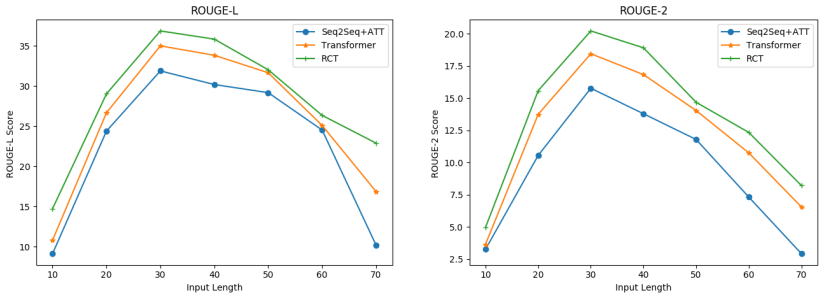
4.5 Ablation Study

Table 3: Ablation study on the English Gigaword dataset. "LS" is used for the abbreviation of lexical shortcut. RT denotes the model without convolution module.

Models	ROUGE-1	ROUGE-2	ROUGE-L
Transformer	35.86	17.11	33.26
Transformer + LS	36.21	17.41	33.65
RT	36.88	17.72	34.08
RCT o/GLU	36.44	17.5	33.72
RCT w/GLU	37.27	18.19	34.62

In this section, we conduct experiments to evaluate the contributions brought by different components. The experiments are conducted on the Gigaword test set. Experimental results are presented in Table 3. The baseline is the original Transformer(base). To validate the effectiveness of the lexical shortcut, we train a counterpart model that only lexical shortcut is included. As the result shown in the second row, lexical shortcut improves the performance by about 0.35 ROUGE-1 points. The third row in Table 3 corresponds the model that takes the RNN output as the encoder output without convolution operations. And the fourth row and the fifth row in Table 3 is the method using RNN and convolution module, the difference between them is whether filtering the sequential context with GLU. The results show that it is necessary to model sequential context for abstractive text summarization. The RNN makes up the shortcoming of the Transformer on insensitivity to word order. And the convolution module captures n-gram features which also helps boost performance. The RCT without GLU reduces performance because the sequential information is weakened.

4.6 Effect of different lengths of input



(a) (b)

Fig. 3: F1 scores of ROUGE-2 and ROUGE-L on different groups of source articles according to their length on English Gigaword test sets.

In this section we investigate how the different input lengths affect the performance of our model. We group the input article with an interval of 10 and get 7 groups whose length ranges from 10 to 70. We plot the performance curve of ROUGE-2 F1 and ROUGE-L F1 on our RCT model, the base Transformer and the seq2seq+attention baseline in Fig. 3. As we can see, our model consistently improves over the other two models for all lengths and our model is more robust to inputs of different lengths.

4.7 Speedup over RNN-based Seq2seq Model

Table 4: Speed and memory usage comparison between the proposed model and RNN-based models, all with batch size 64.

	Training	Inference	Memory Usage
RNN-based	15.2 hours	4.8 samples/s	10GB
RCT	10.8 hours	4 samples/s	5GB
speedup	1.4x	1.2x	0.5x

In addition to ROUGE scores, we also benchmark the speed of our model against the RNN-based encoder-decoder model. We use the same hardware and compare the time cost of training the same samples for one epoch between our model and the RNN-based model with batch size 64 for a fair comparison. We mostly adopt the default settings in the original code [10]. As Table 4 shows, our model is 1.4x and 1.2x times speedup in training and inference. Although an additional RNN based encoder is introduced, the model is still faster and occupies less computing resources.

4.8 Case Study

Table 5: Examples of generated summaries on Gigaword dataset.

Examples
Article 1: the un chief of eastern slavonia , the last serb-held part of croatia , confirmed tuesday that key elections would be held here on april ## as part of local ballots throughout croatia . **Reference:** un confirms elections to be on unk ## in eastern slavonia **Transformer:** eastern slavonia confirms key elections in croatia **RCT:** un chief confirms key elections in croatia
Article 2: the sri lankan government on wednesday announced the closure of government schools with immediate effect as a military campaign against tamil separatists escalated in the north of the country . **Reference:** sri lanka closes schools as war escalates **Transformer:** sri lanka government closes schools **RCT:** sri lanka closes schools as military campaign escalates

We present two examples in Table 5 for comparison. We can observe that: (1) our RCT model is generally capable of capturing the salient information of an article. For example, the subject in Article 1 is "the un chief" which is extracted correctly by our RCT model, but the Transformer model failed. (2) When both models capture the same topic, RCT can generate more informative summary. For Article 2, our model generates "as military campaign escalates" incorporated in the reference, but the Transformer model loses these information.

5 Conclusion

In this paper, we propose a new abstractive summarization model based on Transformer, in which an additional encoder is introduced to capture the sequential context representation. Experiments on Gigaword and DUC2004 datasets show that our model outperforms the state-of-the-art baselines and owns an advantage in speed both on training and inference. The analysis shows that our model is able to generate high quality summaries. Note that we focus on abstractive sentence summarization in this paper. In the future we will investigate the approach of summarizing long documents.

References

1. Cao, Z., Li, W., Li, S., Wei, F.: Retrieve, rerank and rewrite: Soft template based neural summarization. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 152–161 (2018)
2. Dauphin, Y.N., Fan, A., Auli, M., Grangier, D.: Language modeling with gated convolutional networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. pp. 933–941. JMLR. org (2017)

3. Gehring, J., Auli, M., Grangier, D., Yarats, D., Dauphin, Y.N.: Convolutional sequence to sequence learning. In: International Conference on Machine Learning. pp. 1243–1252 (2017)
4. Lin, C.Y.: Rouge: A package for automatic evaluation of summaries
5. Liu, P.J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L., Shazeer, N.: Generating wikipedia by summarizing long sequences. arXiv preprint arXiv:1801.10198 (2018)
6. Nallapati, R., Zhai, F., Zhou, B.: Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In: Thirty-First AAAI Conference on Artificial Intelligence (2017)
7. Nallapati, R., Zhou, B., dos Santos, C., Gulcehre, C., Xiang, B.: Abstractive text summarization using sequence-to-sequence rnns and beyond. In: Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning. pp. 280–290 (2016)
8. Paulus, R., Xiong, C., Socher, R.: A deep reinforced model for abstractive summarization. In: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings (2018)
9. Piji, L., Wai, L., Lidong, B., Zihao, W.: Deep recurrent generative decoder for abstractive text summarization. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 2091–2100 (2017)
10. 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. pp. 379–389 (2015)
11. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). vol. 1, pp. 1715–1725 (2016)
12. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. arXiv preprint arXiv:1803.02155 (2018)
13. Shen, S., Zhao, Y., Liu, Z., Sun, M., et al.: Neural headline generation with sentence-wise optimization. arXiv preprint arXiv:1604.01904 (2016)
14. Sumit, C., Michael, A., M, R.A.: Abstractive sentence summarization with attentive recurrent neural networks. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 93–98 (2016)
15. Takase, S., Okazaki, N.: Positional encoding to control output sequence length. arXiv preprint arXiv:1904.07418 (2019)
16. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems. pp. 5998–6008 (2017)
17. Voita, E., Serdyukov, P., Sennrich, R., Titov, I.: Context-aware neural machine translation learns anaphora resolution. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1264–1274 (2018)
18. Wang, L., Yao, J., Tao, Y., Zhong, L., Liu, W., Du, Q.: A reinforced topic-aware convolutional sequence-to-sequence model for abstractive text summarization. arXiv preprint arXiv:1805.03616 (2018)
19. Zhang, J., Luan, H., Sun, M., Zhai, F., Xu, J., Zhang, M., Liu, Y.: Improving the transformer translation model with document-level context. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 533–542 (2018)
20. Zhou, Q., Yang, N., Wei, F., Zhou, M.: Selective encoding for abstractive sentence summarization. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1095–1104 (2017)