A Combined Model for Extractive and Abstractive Summarization Based on Transformer Model

Xin Liu, Liutong Xu
The Academy of Computer Science and Teconology, Beijing University of Post and Telecommunication
Beijing, China
liuxiaoxin@bupt.edu.cn, xliutong@bupt.edu.cn

***Abstract*—Summary generates by summarizing automatically main information from the critical sentences of the article. The traditional method of generating text summarization uses extractive or abstractive algorithm model built based on neural attention sequence to sequence framework. This kind of model has performance bug and weak parallel computing capability when getting summary, which causes the summary doesn't fit the meaning of the original and has no smooth sentences. Therefore, we put up with a joint summary generation model based on improving transformer. This model can put attention on sentence and provide sequence information for periodical transformer model by recurrent neural network. On the other hand, in the generation stage, Transformer model is used to learn the long distance dependence between words, and the summary statement is more consistent with the original meaning by adding pointer mechanism and consistency loss function. Experiments were carried out on three datasets and a manual evaluation was added to verify that the model has good summary significance.**

***Index Terms*—Text Summarization , Transformer, Sequence Information, Joint Model**

I. Introduction

Summary generation aims to get a simplified input text representation to capture the core meaning of the original content. There are two types of methods: extractive and abstractive. Extractive methods usually selects the original sentence or word [1], such as Lead3, Summarunner [2], SwapNet [3] model. The summary obtained by these models are not smooth due to the lack of connectives. Abstractive methods can generate new words and phrases that are not included in the source text.But the summary has incorrect fact details and duplicate information, and words that are out of vocabulary. In recent years, the pointer generator model proposed by see et al [4] which has the ability to extract words from the original text and reduce the repetition rate. Hsu et al. [5] proposed the inconsistent loss function which combines extractive methods and abstractive methods.

The transformer model proposed by Ashish et al [9] is effective for capturing the global context semantic relationship and parallel computing. In this paper, we proposed TP-EABS (Transformer added Pointer and combine the Extractive and Abstractive methods) model. It adopted the advantages of two types of algorithms and transformer model. The model uses the hidden layer information of GRU to supplement the sequence information of transformer position, and dynamically adjusts the attention of words in the second phase through sentence level attention, so as to reduce the probability of words in sentences with lower weight appearing in the abstract. And we add a pointer mechanism to the transformer model, which enables the transformer to copy words from the original text.

In summary, our contributions are as follows:

- We propose a joint model based on improved Transformer.
- We improve the Transformer architecture by adding position information and pointer mechanisms.
- We have made comparative experiments on CNN/Daily, Papers and DUC-2004 datasets, and the results have been improved.

II. Related Work

In recent years, summary generation has been widely studied. Generally,In the extraction method, key sentences or words in the original text are extracted and presented as abstracts. [3] and [4], [6] used recurrent neural network to code the text, and then mark whether the sentence or word belongs to the summary statement. Although some extraction methods [10] can get high Rouge scores, their readability is very low.

Abstrctive methods are mostly based on the sequence to sequence framework based on neural attention [11], [14]. [12] proposed a new model, which first selects key sentences, then rewrites them with abstract algorithm to generate a brief summary of the text. [11] proposed a new model, which can not only retain the ability to generate new words, but also copy words from the original text accurately to reproduce information, and reduce the repetition rate of the generated words in the summary by updating the attention weight. [9] proposed Transformer, which is completely based on attention mechanism and eliminates recursion and convolution. It can solve the problem of long-distance dependence and realize parallel computing. It can obtain the text semantic information and structural information better.

III. Our model

This chapter introduces three aspects: sentence extraction model, generating model, dynamic word-level mechanism. Fig. 1 gives the overview of TP-EABS model.

DOI reference number: 10.18293/SEKE2020-069

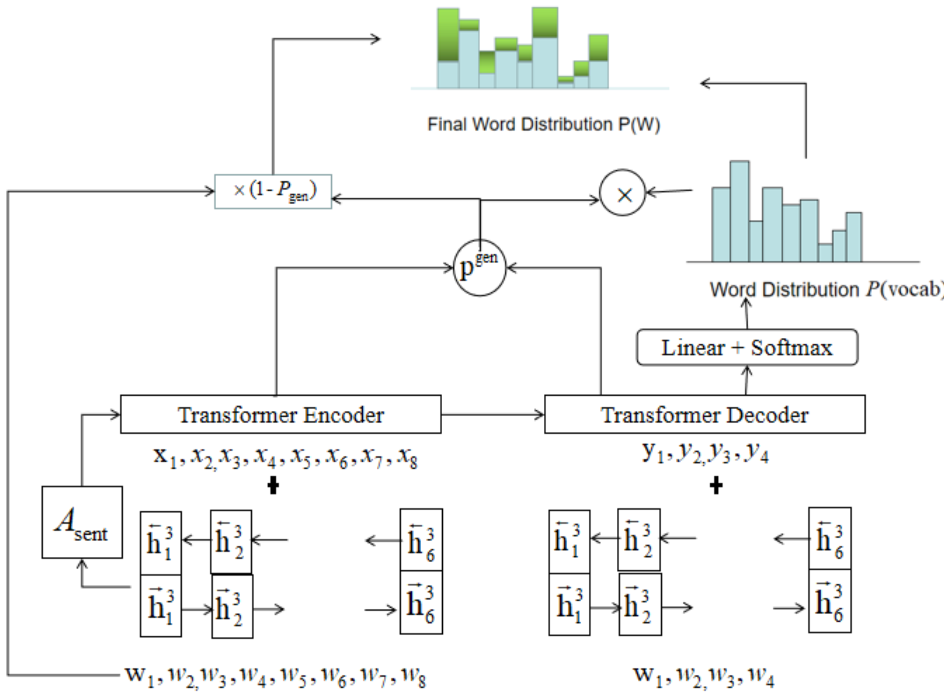

Fig. 1: our model

A. Sentence Extraction Model

The model input is a series of sentences$S = [s_1, s_2, ..., s_m]$, where the representative *m* is the mth sentence and s_i is the ith sentence, expressed as $s_i = [w_1, w_2, ..., w_n]$, the input sentence maps each word to a vector through the language training model, and the i-th sentence is expressed as $s_i = [x_1, x_2, ..., x_n]$, Where *n* represents the nth word embedding vector. we use BiGRU (Bi-directional Gated Recurrent Unit) to process the input word sequence.After reading the words of the sentence, we update its word representation $x_i^j = \left[\overrightarrow{h_i^j}; \overleftarrow{h_i^j}\right]$. We use matrix X to represent the input vector.Get the sentence vector by summing the word vectors in the sentence.

Then the sentence vector is input to the second layer of BiGRU, and then calculated by the sigmoid function to obtain sentence-level weight β_n, The extractor loss is calculated using the following cross-entropy loss.

$$L_{ext} = -\frac{1}{n}\sum_{n=1}^{N}(g_n log\beta_n + (1-g_n)log(1-\beta_n)) \quad (1)$$

In the above formula, g_n is the n-th sentence a summary, the value belongs to 1 and the value does not belong to 0.To get ground truth labels $g = \{g_n\}_n$.We use the unsupervised method proposed by Nillan et al. [5] to get extracted labels.

B. Generating Model

The benchmark model in this paper uses Transformer. We will extract the vector $Z = [z_1, z_2, ..., z_m, ...]$ of the first layer GRU encoding output in the model as the input of the Transformer layer, we think z_i contains word information and location information.

$$[Q; K; V] = W_Z \cdot Z + B_Z \quad (2)$$

$$Attention\,(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (3)$$

Where W_Z and B_Z are training parameters. We obtain the query, key, and value vectors through Equation 10, d_k represents the size of the key value.

$$MultiHead\,(Q, K, V) = Concat\,(head_1, ..., head_h)\,W^O \quad (4)$$

$$where \quad head_i = Attention\left(QW_i^Q, KW_i^K, VW_i^V\right) \quad (5)$$

We represent the output of the Encoder layer as E and denote the output of the Decoder layer as D.We calculate the attention weight between the encoder and decoder and take it out to calculate the probability p_{gen}, which determines whether the word is copied from the original text or generated in the dictionary.Enter the last layer of the decoder into softmax to get the probability of getting words from the vocabulary p_{vocab}.In the end, we use Beam-search with the Beam-size set to 3.

$$P_{voacb} = softmax(V'\,(V \cdot D_4 + b) + b') \quad (6)$$

$$[K_E; V_E] = W_E \cdot E + B_E \quad (7)$$

$$Q_d = W_d \cdot D_1 + B_d \quad (8)$$

$$C_t = MulAttn(Q_d, K_E, V_E) \quad (9)$$

We use the following formula to calculate p_{gen}, where X is the input vector and C is the context text. The pointer generator network is a hybrid between our baseline and the pointer network, because it can both copy words by pointing and generate words from a fixed vocabulary.

$$p_{gen} = \sigma(W_e^T E + W_d^T D + W_C^T C_t + b_{ptr}) \tag{10}$$

Next, we calculate the probability of the final word through the probability p_{vocab} and p_{gen}, where C_t is the output of the encoder and decoder mutual attention matrix at time t.

$$P(w) = p_{gen} P_{vocab} + (1 - p_{gen}) C_t \tag{11}$$

$loss_t$ represents the loss function at time t, and L_{abs}represents the loss function of the generated model part.

$$loss_t = -logP(W_t^*) \tag{12}$$

$$L_{abs} = \frac{1}{T} \sum_{t=0}^{T} loss_t \tag{13}$$

C. Dynamic word-level mechanism

The dynamic word-level mechanism is to reduce the word-level attention through the sentence's attention weight, so that the generative summary can pay more attention to a certain sentence to generate a word, which also makes the sentence weights of the same information different, which reduces the repeatability of the generated words to a certain probability.

This article uses a BiGRU to obtain sentence-level weights in the sentence extraction model. It needs to be added to the word-level weights. The obtained sentence-level weights are the matrix A_{sent} formed by β, and a sentence vector $\tilde{E}_4^t$ obtained by multiplying the word-level attention matrix and value according to the multi-head attention mechanism , And then obtain the updated encoding matrix using the following formula.

$$E_4^t = (W_{sent} A_{sent} + B_{sent})) \tilde{E}_4^t + B_E \tag{14}$$

In order to ensure that the two levels of attention can be kept consistent during the training process, a unified loss function is added here. We use the L_{sw} loss function to represent the error function calculated between sentences and words. Where $m(n)$ is a mapping relationship between words and sentences.

$$L_{sw} = -\frac{1}{T} \sum_{T}^{t=1} log(\frac{1}{\kappa}) \sum_{n \in \kappa} E_n^t \times A_{m(n))} \tag{15}$$

Where κ is the set of the first κ participating words and t is the number of words in the abstract. The loss of inconsistency helps our unified model for end-to-end training benefit both the extractor and the abstractor, and also helps to generate longer digest lengths. Through sentence-level extraction, an improved Transformer generation layer, and a swap mechanism, we finally generate a training loss function L_{sum}

$$L_{sum} = \varepsilon_1 L_{ext} + \varepsilon_2 L_{abs} + \varepsilon_3 L_{sw} \tag{16}$$

where ε_1, ε_2, ε_3 are hyper-parameters. In our experient, we give L_{ext} a bigger weight (e.g., $\varepsilon_1 = 5$) when end-to-end training with L_{sw} since we found that L_{sw} is relatively large such that the extractor tends to ignore L_{ext}.

IV. Experiments

A. Datasets and Settings

We use three datasets for model evaluation: CNN / Daily Mail, Papers, and DUC-2004.The Papers dataset is a private dataset that we build. We use the introduction of the article as the original information and the multi-sentence abstract of the article as the abstract.

In preprocessing, we use byte pair coding (BPE) algorithm [13] to segment words. In this model, we set the vocabulary size to 50,000. The baseline Transformer model is trained using the same hyperparameters as the basic model in Ashish [14]. The number of heads in the Transformer is 8, the size of the feedforward network is 2048, and the training batch size is 8. We use 256-dimensional in BiGRU, which are stitched into 512-dimensional inputs to the Transformer. During the test, we used a beam search with a size of 3 to generate summary, performed 100,000 iterations, saved a basic model every 1000 times, and dropout is 0.5. We limit the maximum output length to 20 and 30, respectively.

B. CNN/Daily dataset

As shown in Table I, we divide the model into Transformer, TPABS (Transformer added pointer mechanism), and TP-EABS. On CNN/Daily dataset, we can see that our model TP-EABS is about 1.7 percentage points higher than Pgen at ROUGE-1. Later, we compared this model with our own baseline model. It was found that before these mechanisms were added, the evaluation index obtained by our model Transformer was not higher than the baseline model, indicating that these mechanisms have improved the summary quality to a certain extent. The ROUGE score is affected by the short length of the generated summary. We are difficult to know why the model gets a low summary score. To evaluate this hypothesis, we randomly select 40 pairs (articles, abstracts) from a fixed test set for manual evaluation. Articles and model-generated summary were submitted to three relevant professionals for evaluation. Each worker has two model-generated summaries, one from the TP-EABS model and one from the PGen model. Workers were asked to choose a better summary according the four different quality metrics from Celikyilmaz et al. [11]. The results are shown in Table II. Interestingly, compared to the PGen model, the summary of TP-EABS is more favored by humans.

In the experiment, we tried two strategies of location information superposition(1) Direct superposition form (ADD): The bidirectional GRU encoding layer information is directly added on the input vector of the Transformer. (2)Learning Strategy (MLP): Add a layer of neural network to let this layer of neural network learn how to superimpose position information and word vector information.As shown in Table I, it is found that the two superimposed effects have slightly higher learning strategies, but the difference is not large. To reduce the amount of calculation, we add the position vector directly to the model.

TABLE I: ROUGE F1 results for various models and ablations on the CNN/Daily Mail test set.

Model	***ROUGE-1***	***ROUGE-2***	***ROUGE-L***
Attn-S2S [6]	32.75	12.21	29.01
PGen [4]	36.44	15.66	33.42
PGen+Cov [4]	39.53	17.28	36.38
Key information [10]	38.95	17.12	35.68
Transformer	35.26	14.12	31.08
TPABS	36.75	15.89	33.15
TP-EABS(MLP)	38.12	17.36	36.45
TP-EABS(ADD)	38.14	17.29	36.52

TABLE II: Head-to-head comparison between test set outputs of PGenand TP-EABS. Analyses done on summaries for Papers.

Model	***PGen***	***same***	***TP-EABS***
Non-redundancy	65	62	182
Coherence	180	42	145
Focus	140	36	176
Overall	160	41	170

C. Papers and DUC-2004

In Papers data set, the length of the article is long, and its summary is mostly not in the original text. We mainly test our improved Transformer model, as shown in Table III. After adding the pointer mechanism, the model obtains The result is 2 percentage points higher than the Pgen model, which proves that the Transformer model is richer in obtaining context information than the recurrent neural network in the language model. On this data set, we set the parameters of Pgen from See et al[4].We did ablation experiments to evaluate the contributions of different mechanismsTransformer, TEABS (Transformer on the extraction model), TPABS (Transformer with a pointer mechanism added), and TP-EABS.The experimental results of the four models are shown in the Table III.

As shown in Table III, our method has made some progress on the current benchmark on DUC-2004 dataset, and ROUGE-1 and ROUGE-L scores have improved the RAS-LSTM model by absolute 0.3 and 1.5 percentage points, respectively. We also compare the model with Feats. We can see that our model still performs better without introducing external information and reinforcement learning. TPEABS improves the data set by 0.13 percentage points and 0.4 percentage points over TPABS. Considering the sequence context information, our model can capture important information and generate high-quality abstracts.

TABLE III: ROUGE F1 results for various models and ablations on the Papers and DUC-2004.

Model	***R-1***	***R-2***	***R-L***	***R-1***	***R-2***	***R-L***
ABS [15]	24.32	7.32	19.24	26.55	7.06	22.05
ABS+ [15]	25.24	7.64	20.32	28.18	8.49	23.81
FeatS2S [7]	26.89	9.52	23.76	28.61	9.42	25.24
RAS-LSTM [8]	28.42	10.77	22.46	28.97	8.26	24.06
PGen [4]	29.70	13.19	27.32	–	–	–
Transformer	28.54	12.04	24.92	–	–	–
TPABS	29.32	12.58	25.45	–	–	–
TEABS	29.46	13.21	26.15	–	–	–
TP-EABS	31.69	14.52	28.18	29.27	9.95	25.54

V. Conclusion

We propose a joint abstract generation model based on improved Transformer. Most importantly, we improved the Transoformer model so that it has the ability to copy words from the original text. After adding sequence information and extraction stages, the model in this paper can obtain more complete summary information in the uniformly distributed original text, and it will not ignore its importance because the key information is located later. Through end-to-end training of our model, we conducted experiments on three datasets and conducted reliable human evaluation on private datasets, proving that the model has good summary information significance.

References

[1] Piotr Bojanowski, Edouard Grave,Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5:135–146, 2017.

[2] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In Proceedings of Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17), pages 3075–3081.

[3] Jadhav A , Rajan V . Extractive Summarization with SWAP-NET: Sentences and Words from Alternating Pointer Networks[C]// Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2018.

[4] Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointergenerator networks. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics.

[5] Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. A unified model for extractive and abstractive summarization using inconsistency loss. arXiv preprint arXiv:1805.06266,2018.

[6] Cao Z, LiW, Li S et al (2016) Attsum: joint learning of focusing and summarization with neural attention[J].arXiv preprint arXiv:1604.00125

[7] Nallapati, R., Zhou, B., dos Santos, C., Gulcehre, C., Xiang, B.: Abstractive text summarization using sequence-to-sequence RNNs and beyond.

[8] Sumit, C., Michael, A., Rush, A.M.: Abstractive sentence summarization with attentive recurrent neural networks. Human Language Technologies, pp. 93–98 (2016)

[9] 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, pages 5998–6008.

[10] Ramesh Nallapati, Bowen Zhou, and Mingbo Ma. 2016a. Classify or select: Neural architectures for extractive document summarization. arXiv preprint arXiv:1611.04244

[11] Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In EMNLP

[12] Yen-Chun Chen and Mohit Bansal.Fast abstractive summarization with reinforce-selected sentence rewriting. arXiv preprint arXiv:1805.11080,2018.

[13] Sebastian Gehrmann, Yuntian Deng, and Alexander M Rush. Bottom-up abstractive summarization. arXiv preprint arXiv:1808.10792, 2018.

[14] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. Incorporating Copying Mechanism in Sequence-to-Sequence Learning. In ACL, 2016.

[15] 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)