International Core Journal of Engineering Volume 5 Issue 12, 2019
ISSN: 2414-1895 DOI: 10.6919/ICJE.201911_5(12).0024

Document Summary Optimization Generation Based on Joint Information Sharing with Self-Attention

Jian Di [a], Renjie Qi [b]

Department of Computer, North China Electric Power University, Baoding 071003, China.

[a]dijian6880@163.com, [b]861041427@qq.com

Abstract

The extraction method extracts sentences from the source text, which will cause information redundancy. The generator method can generate non-source words, which will cause grammatical problems and poor naturalness. Combining the decimation method and the generative method, the bidirectional context information word vector generated by the pre-training source is extracted from the sentence function to extract the key sentence, and the key sentence is rewritten as a cloze task in the digest generation stage to generate the final digest. The experimental results show that the model has achieved good results in the Sogou data set.

Keywords

Abstractive,Summarization,Self Attention, Recurrent Neural Networks.

1. Introduction

The purpose of the text summary task is to generate a summary from the input document while retaining the key information, which is generally divided into a decimation method and a production method. The extraction method extracts the key sentence generation summary from the source text; the production method reorganizes or rewrites the sentence of the source text to generate a summary. Innovations and improvements to the two method models have been varied. For example, See et al. proposed a summary model based on a sequence-to-sequence framework and introduced a replication mechanism. Paulus et al. proposed introducing a time-attention mechanism in the encoder decoder.

The previous language generation model has the following problems: a) Most models use le ft-to-right one-way decoders, which will lose a lot of context information and cause perfor mance degradation; b) no pre-trained context language model is used in the decoder part, d ecoding It is difficult to learn how to perform summary representation and context correlati on; c) the deprecated model can guarantee the fluency of the sentence, but it will cause inf ormation redundancy; the generated model can use non-original words, the language is strea mlined, but it will generate grammatical problems and generate Language lacks naturalness.

2. Method

2.1 Encoder-Decoder Frame

The encoder-decoder is the most common network for text summarization tasks, accepting a sequence as input and encoding the information in the sequence as an intermediate representation, and finally decoding the intermediate representation as the target language. The Transformer model proposed by Vaswani et al. [9], while greatly improving the accuracy of natural language processing tasks, is a cyclic neural network (RNN), such as the Long and Short Term Memory Network (LSTM).

Gated loop units (GRU) still have their unique advantages when combined with the Transformer model [4]. Ba et al. [5] proposed the use of normalized function stabilization training in each LSTM

ISSN: 2414-1895 DOI: 10.6919/ICJE.201911_5(12).0024

unit; See et al. [2] proposed a pointer generation network based on RNN, and controlled the coverage of source text by overlay vector.

The role of the encoder is to read the input sequence and produce a vector representation of each word. In order to efficiently encode the input sequence, we use an encoder based on the self-attention mechanism. Compared with the cyclic neural network, we do not need to process the input sequence word by word, but calculate each word simultaneously through the self-attention mechanism. Context vector, so it has good parallelism and low computational complexity.

The encoder can be stacked in multiple layers, each layer including a multi-head attention layer and a forward feedback layer. The multi-attention layer is used to calculate the attention weight of each word about other words in the input sequence to get the context representation of each word. "Multi-head" means to map the input to multiple subspaces, and in these sub-spaces The context representation is calculated in space, and finally the calculation results are stitched together.

2.2 Network Design

The input sequence usually contains many words, and only a few of them contain key information about the entire sequence. This key information is also needed by the model to generate the summary. In order for the model to screen the key information of the input sequence.

The gated network is used to control the flow of information from the input sequence to the output sequence, removing unwanted information, allowing the decoder to focus more on generating digests from critical information. Specifically, the gated network structure is as shown. The input is a sentence representation s of the original sequence, and the word of a word in the sequence represents hi; the output is a new vector obtained by filtering hi to represent h. Referring to the work of Devlin [3], etc., it uses h_i (the hidden layer corresponding to the start identifier of the input sequence) as the vector representation of the sentence, so in our work, h0 is also used as the representation of the input sequence vector.

$$g_i = sigmoid(W_g[h_i, s] + b) \tag{1}$$

Where W_g is a linear transformation, b is an offset, and the larger the g, the more critical the word is. Then, the amount of information to the decoder is controlled by g to obtain the filtered vector h as shown in equation. Filtering each word yields a vector representation of the entire sequence (h_0, h_1,..., h_n). This vector sequence is then passed to the decoder for generating a digest.

$$\widetilde{h_i} = g_i h_i \tag{2}$$

2.3 Parameters and Self-Attention

One of the functions of the visible decoder is to encode the generated digest sequence to obtain its vector representation, which is similar to the function of the encoder. The difference is that the encoder encodes the input sequence, and the decoder encodes the part. The function is to encode the summary sequence, but this difference does not affect the functional similarity of the two modules. Based on this similarity, we propose an encoder-sharing decoder, which uses the encoder as one of the decoder modules and passes the encoding task of the generated sequence to the encoder.

$$\alpha_j^i = \frac{\exp(h_t^j)}{\sum_{k=1}^{t} \exp(h_t^k)} \tag{3}$$

The benefits of doing so are as follows. a) Integration of redundant functional modules reduces model parameters and reduces model complexity. It turns out that Vaswani et al. used an extra layer of attention to encode the digest sequence, and we removed the module and handed the coding task to the encoder. b) Equivalent to the training data of the encoder. The training data of the original encoder only has the input sequence. Now it also includes the digest sequence.

Through the training of more data, the encoding ability of the encoder will be enhanced. c) By using the same encoder, the input sequence and the summary sequence can be mapped to the same vector space. The decoder's subsequent module needs to calculate the attention weight between the input sequence and the digest sequence. The vector representation of the two is in the same vector space,

ISSN: 2414-1895 DOI: 10.6919/ICJE.201911_5(12).0024

which is beneficial to the calculation of the dot product attention function, so that the model can mine the input sequence and output more clearly. The relationship between sequences.

2.4 Loss function

During the training phase, the goal of the model is to maximize the probability of the digest sequence y given the input sequence x, so the objective function is to minimize the negative log-likelihood function.

$$L = -\sum_{t=1}^{n} \log \mathrm{P}(y_t = y_t^* | X_i, S) \quad (4)$$

The digest generation phase is a decoding process that regenerates the text digest after the source document information is reduced. The whole process provides a more complete input sequence, which is consistent with their pre-training phase. The model idea of this paper is as follows: Firstly, the pre-trained data is used to obtain the context information. Then, the key sentence extraction stage discards the key sentence with the lowest score according to the source document. Finally, the abstract generation stage is based on the source document and other words in the key sentence, one time step. Concentrate on predicting a word and regenerate the final text summary based on the extracted key sentences.

3. Experiment Analysis

3.1 Data Set

The dataset used in this article is the Sogou dataset. we use a pre-processed version such as Rush. The processing details are as follows: a) All words are lowercase.b) The numbers are replaced by "#" .c) less than 5 times the number of occurrences of the unknown word with a word identifier "<unk>" instead.

The Sogou data set is large and has been divided into three parts: training set, verification set, and test set.

3.2 Evaluation

Following the work already done, we use the ROUGE indicator to assess the quality of the generated summary. The ROUGE indicator mainly evaluates the coincidence between the model generation summary and the standard summary. The higher the coincidence, the higher the ROUGE score. The ROUGE indicator evaluates the coincidence degree from three granularities, namely word, binary phrase, and longest common subsequence, corresponding to three indicators, namely ROUGE-1, ROUGE-2, ROUGE-L, each indicator contains accuracy rate. , recall rate, F1 value; use the F1 value to judge the effect of the model on the Sogou dataset.

3.3 Implementation Details

In terms of model parameters, we build the source dictionary and the target dictionary based on the source text and summary text of the training set, the sizes are 90000 and 68883 respectively; the word vector dimension size is 256; the encoder and decoder layers are set to 2, all The multi-attention layer contains 4 headers, the number of hidden layer neurons is 256; the intermediate layer size of the forward feedback layer is 1024; using position coding [1], to use the sequence information of the sequence, between the various network layers The residual is connected to avoid the gradient disappearing; use the dropout method [6] to avoid overfitting and set the ratio to 0.1.

During the training phase, the training set is randomly disrupted, batch training is used, and the batch size is set to 64. Using the Adam optimizer [7], the initial learning rate is set to 0.001, and after each training on the training set, the learning rate is attenuated to the current Half of it. After training 10 epochs, the model tends to converge.

In the test phase, the beam search method is used to select the candidate summary sequence, and the beam width is set to 5. Beam search tends to select shorter summary sentences. To avoid this shortcoming, we divide the scores of each candidate summary in the search process by its length to encourage the model to generate longer summaries.

ISSN: 2414-1895 DOI: 10.6919/ICJE.201911_5(12).0024

3.4 Results Analysis

In this paper, the comparison model and the model of this paper are unified and listed as follows, and the best-performing models in each column are marked with black bold numbers. The experimental results are shown in Table 1. From the model presented in this paper, it can be clearly seen that in the ROUGE-1 and ROUGE-L evaluations, the method exceeds all the comparative experimental models, indicating that the pre-trained text summary model is effective. of. On the one hand, compared with the current advanced extracted NeuSUM model, the model of this paper increases the scores of ROUGE-1 and ROUGE-L by 0.77 and 0.75 percentage points respectively. On the other hand, compared to the Pointer-generator model that also uses the replication generation mode, ROUGE-1, ROUGE-2.

Table 1. Comparison of experimental results

	Model	1	ROUGE Score 2	L
Extracted model	LexRank	39.12	15.61	36.92
	Pointer-generator	38.53	14.21	37.42
	MMR	41.22	18.72	35.87
Generate model	Refresh	38.96	17.26	35.14
	Textsum	40.29	18.71	36.58
	NeuSum	38.68	19.21	34.69
ours		42.12	18.81	38.92

Both ROUGE-3 and ROUGE-3 increased by 2.83, 1.6, and 2.35 percentage points respectively. Secondly, ROUGE-1, ROUGE-2, and ROUGE-L were improved compared with the well-developed Bottom-Up model, which increased by 1.14, 0.2 and 0.39 percentage points respectively. However, the ROUGE-2 score did not reach the best level, which was 0.13 percentage points lower than the NeuSum model. Based on the above comparison experiments, the proposed text-synthesis model combined with pre-training has a certain improvement in performance compared to the single extraction or generation model, and achieves the best results.

4. Conclusion

In this paper, a generative text digest method based on encoder sharing and gating network is proposed. The encoder is used as part of the decoder, so that the encoder can not only encode the input sequence, but also encode the generated digest sequence. The training of the encoder is also introduced; at the same time, a gated network is introduced to filter the information of the input sequence and retain the key information, so that the model can generate the abstract according to the key information of the original text. Experiments on the abstract data sets Sogou prove that the proposed method based on encoder sharing and gating network can significantly improve the training speed of the model and the quality of the generated summary.

References

[1] Vaswani A, Shazeer N, Parmar N, et al. “Attention is all you need”,31st Conference on Neural Information Processing Systems. Long Beach: MIT Press, 2017: 6000–6010
[2] Sutskever I, Vinyals O, Le Q V. “Sequence to sequence learning with neural networks”,28th Conference on Neural Information Processing Systems. Montreal: MIT Press, 2014: 3104–3112
[3] Bahdanau D, Cho K, Bengio Y. “Neural machine translation by jointly learning to align and translation , Proceedings of 3rd International Conference for Learning Representations. San Diego”, 2015

ISSN: 2414-1895 DOI: 10.6919/ICJE.201911_5(12).0024

[4] Rush A M, Chopra S, Weston J. “A neural attention model for abstractive sentence summarization” , Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Lisbon: ACL Press, 2015: 379–389
[5] Hu Baotian, Chen Qingcai, Zhu Fangze. LCSTS: “a large scale chinese short text summarization dataset” , Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon: ACL Press, 2015: 1967–1972
[6] Chopra S, Auli M, Rush A M. “Abstractive sentence summarization with attentive recurrent neural networks” , The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego: ACL Press, 2016: 93–98
[7] Nallapati R, Zhou Bowe, Ccíero Nogueira et al. “Abstractive text summarization using sequence-to-sequence RNNs and Beyond” , Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning. Berlin: ACL Press, 2016: 280– 290
[8] Gu Jiatao, Lu Zhengdong, Li Hang, et al. “Incorporating copying mechanism in sequence-to-sequence learning” , Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. Berlin: ACL Press, 2016: 1631–1640
[9] Zhou Qingyu, Yang Nan, Wei Furu, et al. “Selective encoding for abstractive sentence summarization” , Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics. Vancouver: ACL Press, 2017: 1095–1104
[10] Lin Junyang, Sun Xu, Ma Shuming, et al. “Global encoding for abstractive summarization” , Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. Melbourne: ACL Press, 2018: 163–169