A Generative Text Summarization Model Based on Document Structure Neural Network

Haihui Huang and Maohong Zha(✉)

School of Software Engineering, Chongqing University of Posts and Telecommunications, Chongqing, China
huanghh@cqupt.edu.cn

Abstract. Aiming at the low accuracy of the automatic generation of text summaries in the field of data mining, as well as the defects of the existing encoder and decoder models, this paper proposes a generative text summarization model based on the document structure neural network. The model introduces the document structure, divides the text into a word encoding layer and a sentence encoding layer, and builds a top-down hierarchical structure to avoid the back propagation error problem caused by the long input sequence in the traditional encoder and decoder model; At each level, an attention mechanism is added, and a multi-attention mechanism is proposed and introduced, which refines the granularity of the attention mechanism, thereby improving the accuracy of text summary generation. Experimental results show that, compared with the original encoder-decoder model, this model can effectively refines the granularity of the attention mechanism and significantly improve the accuracy of text summary generation.

Keywords: Data mining · Text summarization generation · Multi attention mechanism · Document structure neural network

1 Introduction

There are a lot of text data such as news and blogs on the Internet that fill our lives [1]. However, there are often redundant and useless information in these text data. Through a short summary, we can efficiently retrieve text content and mine text information. However, manually writing abstracts for each article, news, and blog requires a lot of manpower and material resources.

Natural language processing is a relatively active processing method in the field of data processing, and it is also an important step for public opinion analysis and data mining [2]. Text summarization is an important field in natural language processing, including extractive text summaries and generative text summaries. Extractive text summaries extract the most important sentences in the original text as abstracts, while generative text summaries automatically generate abstract sentences based on the content of the text. Text summaries can summarize a medium-length text in one sentence, which can greatly improve efficiency compared with manual text summaries. But its accuracy is still relatively low, especially in the capture of key words [3].

M. Mahmud et al. (Eds.): AII 2021, CCIS 1435, pp. 176–187, 2021.
https://doi.org/10.1007/978-3-030-82269-9_14

The traditional encoder-decoder model [4] first encodes the words of the text, then adds the attention mechanism [5] to learn the key words of the article, and then decodes the word encoding to generate a text summary. Compared with the previous rule-based and statistical-based summary generation methods, this type of method has a significant improvement in efficiency, but the granularity of its attention mechanism is relatively rough, and it cannot achieve good attention for long text learning. As a result, it is difficult to capture the key sentences and key words in a medium-length text, resulting in a large deviation in the accuracy of the generated abstract. For example, given a text [In addition, according to the "Business Insider" website, in response to Trump's above remarks, Andrew Bates, the director of rapid response of the Biden campaign team, responded: "Due to the failure of Donald Trump, China's position has become stronger in all aspects, while the US's status has declined." He said, "Trump is the weakest president in American history against China."] Humans can quickly capture the key sentence "Trump is the weakest president in the history of the United States against China." However, the text in this text is too long and the relationship between the characters involved is complex, the traditional encoder-decoder model will produce large deviations in key words and sentence capture. The reason is that although it introduces an attention mechanism, the traditional model processes the entire text sequence and uses a time-series neural network. However, for a long text vector sequence, gradient dispersion or derivative calculation deviation will still occur, resulting in deviations. Introducing the attention matrix on the basis of, will increase the error and cause the final generated summary to have a large deviation. The structure of the document has the following characteristics: sentences are composed of words, and documents are composed of sentences, which a bottom-up hierarchical structure can be constructed. Based on this, this paper proposes a generative text summarization model based on the document structure neural network (DSNN-GSM) to improve the granularity of the attention mechanism and improve the accuracy of the generative text summary.

This paper mainly studies the generation of text generative summaries. Based on the encoder-decoder model based on the attention mechanism, this paper proposes an improved model DSNN-GSM that divides the neural network model into layers. The neural network level is divided into word coding layer and sentence coding layer, which is more in line with the text structure. At each level, attention mechanism and multi-attention mechanism are added to make the attention mechanism more granular and make the model better Understand the meaning of the text. In general, the contribution of this article has the following two points:

1. The original encoder-decoder model is divided into a bottom-up model of word coding level and sentence coding level, which shortens the length of the input sequence of each processing unit, thereby alleviating the back propagation caused by the excessively long sequence Problems with large derivation errors;
2. At each level, an attention mechanism or a multi-attention mechanism is introduced to refine the attention granularity of the model, so that it can more accurately capture the key information in the article, and improve the accuracy of generating abstracts.

Next, this article will analyze specific issues. In Sect. 2 we will introduce other processing methods in this field; in Sect. 3, we will focus on the main content, which will

introduce the generative text summarization model based on document structure neural network; Sect. 4 will introduce the evaluation method of the text summary and make a confirmatory comparison between the model in this article and the reference model; Sect. 5 gives the conclusion of this article; Sect. 6 is the part of the cited references.

2 Related Work

2.1 Encoder-Decoder Model Based on LSTM

Generative text summaries are mainly realized by the structure of deep neural networks. The Sequence-to-Sequence sequence proposed by the Google Brain team in 2014 opened up the fiery research on end-to-end networks in NLP. Sequence-to-Sequence is also known as Encoder-Decoder (Encoder-Decoder) architecture. Encoder and Decoder are both composed of several layers of RNN or LSTM. Encoder is responsible for encoding the original text into a vector C; Decoder is responsible for extracting information from this vector C, obtaining semantics, and generating text summaries. However, due to the problem of "long-distance dependence", when the RNN entered the word at the last time step, a large part of the data had been lost. At this time, the vector C generated by the encoder also lost a lot of information, resulting in inaccurate results. Therefore, the LSTM neural network is used, and the Attention mechanism is introduced to capture the key words in the text [6].

2.2 Gated Recurrent Unit (GRU) Neural Network

The structure diagram of GRU neural network [7] is shown in Fig. 1. GRU is a very effective variant of the LSTM network. It has a simpler structure than the LSTM network, and the effect is also very good, so it is also a very manifold network at present. Since GRU is a variant of LSTM, it can also solve the long dependency problem in RNN networks.

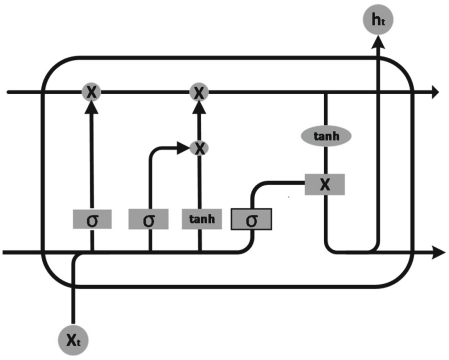

Fig. 1. Neural network structure diagram

Both LSTM and GRU introduce a gating mechanism in the recurrent neural network [8]. In a general RNN recurrent neural network, If the prediction y_t at time t depends on

the input $x_t - k$ at time $t - k$, when the time interval k is relatively large, the problem of gradient disappearance or gradient explosion is prone to occur, then it is difficult for the recurrent neural network to learn such long input information. In this case, when the current forecast requires longer-term information, the problem of long-term dependence will arise. However, if all the information entered at the past moment is stored in order to learn very long information, it will cause the saturation of the stored information in the hidden state h and the loss of important information. To this end, a better solution is to introduce a gating mechanism to control the speed of information accumulation, including selectively adding new information, and selectively forgetting previously accumulated information.

There are only two gates in the GRU model, namely the update gate Z_t and the reset gate R_t The update gate Z_t is used according to formula (2.1) to control how much information the current state h_t needs to retain from the historical state h_{t-1}, and how much new information needs to be received from the candidate state h_t. The larger value of the update gate, the more state information from the previous moment is brought in.

$$Z_t = \delta(W_Z x_t + U_Z h(t-1) + b_Z) \tag{2.1}$$

Then calculate the hidden state h_t according to formula (2.2).

$$h_t = Z_t \odot h(t-1) + (1 - Z_t) \odot \widetilde{h}_t \tag{2.2}$$

The reset gate R_t controls whether the calculation of the candidate state $\widetilde{h}_t$ depends on the state h_{t-1} at the previous moment according to formula (2.3). In other words, it is used to control the degree of ignoring the state information at the previous moment. The smaller the value of the reset gate, the more ignorance.

$$r_t = \delta(W_r x_t + U_r h(t-1) + b_r) \tag{2.3}$$

The candidate state $\widetilde{h}_t$ at the current moment can be obtained by formula (2.4):

$$\tilde{h}^t = \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \tag{2.4}$$

3 Generative Text Summary Model Based on Document Structure Neural Network (DSNN-GSM)

3.1 DSNN-GSM Model Structure

This paper proposes a generative text summarization model DSNN-GSM based on document structure neural network. The model architecture is shown in Fig. 2. It is divided into word embedding layer, word encoding layer, sentence encoding layer and decoding layer.

Word embedding layer is used to segment the text and convert it into a one-hot encoding, and at the same time do partition processing, and divide each sentence into a processing unit for subsequent processing.

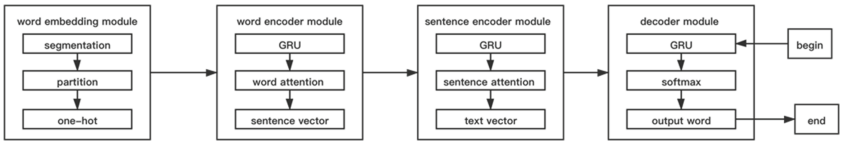

Fig. 2. DSNN-GSM model architecture

Word encoding layer uses the bidirectional GRU neural network to perform word encoding processing on the one-hot vector to obtain the word encode with high representation and add the word attention matrix to obtain the sentence vector.

Sentence encoding layer uses the bidirectional GRU neural network to perform sentence encoding processing on the sentence vector obtained above to obtain a sentence encode with high representation and add the word attention matrix to obtain a text vector.

Decoding layer decodes the obtained text vector, takes the above obtained text vector and BEGIN tag as input to the decoding module, and then performs a softmax calculation to obtain the probability of the next word to be output, and outputs the word with the highest probability. This predicted word will be used as input in the next time sequence, and the weight parameters of the neural network will be updated through the current state, and then the next word to be output will be calculated through softmax. By analogy, a complete text summary is finally generated.

3.2 Algorithm Flow Description

The hierarchical structure diagram of DSNN-GSM is shown in Fig. 3.

The DSNN-GSM algorithm process has the following 6 steps:

1. Split the text into words and perform partition processing to obtain multiple processing units. Convert each word in each processing unit into an embedded representation of a one-hot vector, record it as w_{ij}, and input it to the word-level coding layer. Where i represents the i-th sentence and j represents the j-th word in the i-th sentence.
2. Use each sentence as a processing unit and perform word encoding operations on it. Input the GRU neural network and its variants to perform word encoding processing on the one-hot vector to obtain training parameter matrix and word encoding with high representation. Among them, the training parameter matrix is an incidental product of the neural network model training process, which is used to adaptively adjust the model error.
3. Introduce a random context matrix u_w, do a softmax operation with the word encoding obtained above to obtain the word attention matrix, and then do the dot product and weight the results of the attention matrix and the hidden layer to obtain a highly representative sentence vector S_L. L represents the L-th sentence vector.
4. Input the above sentence vectors into GRU neural network for sentence coding. The sentence vector with high representation is obtained.
5. Introduce a random sentence attention matrix, encode it with the obtained sentence and do a softmax operation to generate a document vector T with high representation.
6. Pass the finally generated text vector as an initialization parameter to the decoder for decoding operation to generate a text summary.

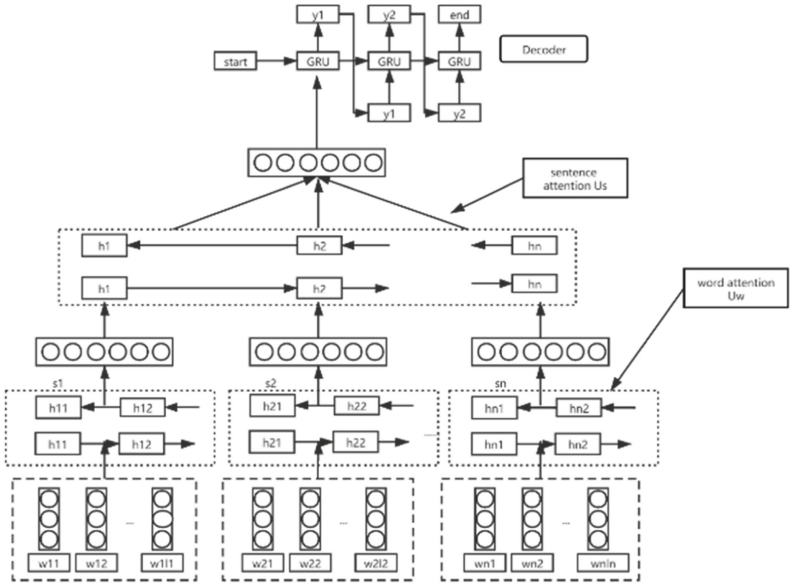

Fig. 3. SNN-GSM hierarchy structure diagram

Among them, the steps of the decoding operation in step 6 are as follows:

a. Input the text vector T as an initialization parameter to the decoder, and pass the label 'begin' as an input parameter to the initialized decoder;
b. The initialized decoder module runs time step once, and uses softmax to calculate the next word with the highest probability and output it.
c. Use the word output at the previous moment as the input at the current moment, and the neural network will adaptively update the weight of the neural network according to the error value of the back propagation process, run time stpe again, calculate the next word with the highest probability through softmax and output it.
d. Repeat the iterative process of c until the 'end' tag is decoded, then end the iterative process, and get the complete summary of the text.

The specific algorithm implementation process is described as follows:

First, word embedding layer performs word segmentation processing on the input sample data, and partitions the set of words in each sentence into a processing unit to obtain the original word sequence $(x_{11}, x_{12}, \cdots, x_{1m}, x_{21}, x_{22}, \cdots, x_{2m}, \cdots, x_{nm})$, where x_{ij} represents the jth word of the i-th sentence.

Then it is transformed into a one-hot vector $(x_{11}, x_{12}, \cdots, x_{1m}, x_{21}, x_{22}, \cdots, x_{2m}, \cdots, x_{nm})$. After that, the one-hot vector is used as the input of the word encoding module. It should be noted that each partition is processed as an independent module, that is, there is no relationship between sentences at this time, and only the relationship between words within each sentence is considered.

Adopt GRU neural network model based on time series. The feature of GRU neural network is that it has update gate and reset gate. It is a variant of long and short memory neural network. The update gate is used to control the extent to which the state information from the previous moment is brought into the current state. The larger the value of the update gate, the more state information from the previous moment is brought in; the reset gate is used to control ignoring the previous moment. The degree of status information, the smaller the reset gate value, the more ignored. Using this feature can solve the problem of gradient dispersion of long text sequences in the neural network training process. Through the two-way GRU model, the new word vector u_{ij} of each word can be mapped.

At the same time, the bidirectional GRU splices the forward and backward states, as shown in formula (3.1):

$$h = \left(h_{forward}\, h_{backward}\right) \tag{3.1}$$

Among them, h represents the state vector of the hidden layer after forward and backward propagation, $h_{forward}$ represents the state vector of the hidden layer forward propagation, and $h_{backward}$ represents the state vector of the hidden layer backward propagation.

Then, the word context matrix u_w is randomly initialized, and the attention matrix is obtained according to formula (3.2):

$$\partial_{ij} = \frac{\exp\left(u_{ij}^T u_w\right)}{\sum_L \exp\left(u_{ij}^T u_w\right)} \tag{3.2}$$

Where L represents the L-th partition.

Then, take the weighted dot product of ∂_{ij} and the hidden layer value h to obtain the sentence vector. After that, each obtained sentence vector $\boldsymbol{s}_i$ is used as the input of the sentence encoding module, and the Bidirectional GRU is used to encode the sentence, and the forward and backward state splicing $h = (h_{forward}, h_{backward})$ is obtained. Then, the sentence context matrix u_s is initialized, and the sentence attention matrix is obtained according to formula (3.3).

$$\partial_{ij} = \frac{\exp\left(u_{ij}^T u_s\right)}{\sum_s \exp\left(u_{ij}^T u_s\right)} \tag{3.3}$$

Among them, S indicates that the scope is the entire text.

Then do a weighted dot product of ∂_{ij} and the hidden layer value h to get the final text vector. The context matrix is learned through the network in the training process. Finally, the last state of the encoding process, that is, the last generated text vector, is used as the initialization parameter of the decoder to be passed to the decoder to be decoded to obtain a generative summary of the result text.

3.3 Multiple Attention Mechanism

The attention mechanism introduced by the word encoding layer and sentence encoding layer in the model is a single attention mechanism. This paper also proposes a multiple attention mechanism. Since the introduction method of the attention mechanism of the word encoding layer is the same as that of the sentence encoding layer, only the word coding layer module is taken as an example here. The multiple attention mechanism is changed to randomly initialize n context matrices u_{wk} based on the original attention mechanism, and a single attention matrix is calculated according to formula (3.4).

$$\partial_{ijk} = \frac{\exp\left(u_{ij}^T u_{wk}\right)}{\sum_L \exp\left(u_{ij}^T u_{wk}\right)} \tag{3.4}$$

Then use formula (3.5) to weight all its attention matrices to get the final attention matrix.

$$\partial = \sum_n^1 \frac{\exp\left(u_{ij}^T u_{wk}\right)}{\sum_L \exp\left(u_{ij}^T u_{wk}\right)} \tag{3.5}$$

Among them, $k \in (1, n)$. The selection of n depends on the number of nodes in the calculation unit, and the maximum number of nodes in the calculation unit cannot be exceeded. The best selection of n can be obtained by formula (3.6).

$$n = N_{node} * U_{use} * (1 + W/C) \tag{3.6}$$

Among them, * in the equation is a multiplication operator, W/C is the ratio of idle time to computing time, N_{node} is the number of nodes, and U_{use} is the utilization rate of all nodes. That is, the higher the proportion of node idle time, the larger n can be set. The higher the proportion of node calculation time, the lower n, but the total number of n cannot exceed the total number of nodes N.

Using multiple attention matrices to replace a single attention matrix can superimpose the attention effect of a single matrix and strengthen the attention effect of attention.

4 Experiment

4.1 Text Summary Evaluation Method

Text summary evaluation methods are divided into two categories. One is internal evaluation methods, which provide reference abstracts and evaluate the quality of text abstracts on the basis of reference abstracts. It is the most commonly used text summary evaluation method in the industry. The second is an external evaluation method, which does not provide a reference abstract, and uses the document abstract to replace the original document to execute a document-related application. This paper adopts the Edmundson evaluation method [9] of the internal evaluation method, which is to objectively

evaluate the text summary by comparing the overlap rate of the text summary w_{match} generated by the model and the target text summary (expert summary) w_{total} . Calculate the coincidence rate p_i of each text summary by formula (4.1).

$$p_i = \frac{w_{match}}{w_{total}} * 100\% \tag{4.1}$$

This paper then uses the ROUGE (recall-oriented understudy for gisting evaluation) index proposed by Lin et al. to compare and evaluate each model [10]. This indicator evaluates the pros and cons of the summary model based on the number of n-ary common subsequences of the generated summary in the standard summary, where R-1 and R-2 refer to 1-element and 2-element subsequences, and RL means the longest Common subsequence.

4.2 Experimental Parameter Settings

This article uses the public Chinese text abstract data set Test Data of NLPCC 2017 Task1 of the NLPCC 2017 conference organizer to conduct experiments. In the experiment, the data set is preprocessed by keras [11], word segmentation is used by hanLP, converted into one-hot vector input, and word2vec matrix is obtained using word2vec [12] to training. The output dimension of the word embedding module is set to 200, and the output dimension of the word encoding module is set to 100. The GRU hidden state vector dimension is set to 200, the activation function uses softmax [13], the batchsize is set to 64, and the learning rate is set to 0.05. Among them, the weight parameter matrix in the GRU and softmax classifiers is determined by the model itself, and the gradients of all parameters are calculated through back propagation, and the parameters are updated adaptively. At the same time, in order to prevent overfitting, this paper introduces the Dropout technology [14] and sets its parameter ratio to 0.5 to reduce the overfitting phenomenon that occurs on the training set.

4.3 Activation Function Selection Analysis

The core of the DSNN-GSM model is the activation function selection. Generally, a nonlinear function is introduced as the activation function, which can make the expressive ability of the deep neural network more powerful. This paper selects softmax function, Sigmoid function, Relu function and tanh function [15], and compares and analyzes different activation functions under the same data conditions, and finally obtains the activation function with higher summary accuracy and less time.

Figure 4 compares four different activation functions in terms of accuracy and time consumption. In terms of accuracy, the softmax activation function has the highest accuracy rate of 91.4%, the relu function is the closest to softamax, the accuracy rate reaches 84.3%, and the sigmod function has the lowest accuracy rate, only 49.0%. In terms of time, the softmax function, relu function and tanh function are relatively close, and softmax takes the least time. From the comparison results; it can be seen that the softmax function is most suitable for this model.

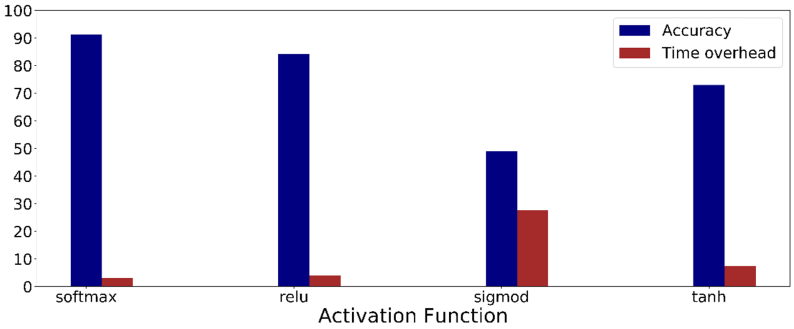

Fig. 4. Activation function analysis

4.4 Comparative Analysis of Methods

In order to prove the advantages of the proposed model, the DSNN-GSM model, BiLSTM [16] and RNN-context [17] model are used to compare the coincidence rate on the "test data of nlpcc 2017 task 1" of the nlpcc 2017 conference. The results are shown in Table 1.

Table 1. Comparison of three methods

Sample category	method	DSNN-GSM	BiLSTM	RNN-context
Training sample	Coincidence rate (%)	91.40	81.00	81.67
Test sample	Coincidence rate (%)	86.65	72.71	75.56

Experimental results show that the coincidence rate of BiLSTM under the test sample is 72.71%, and the coincidence rate of RNN-context is about 75.56%. In contrast, the overlap rate of abstracts generated by the DSNN-GSM model can reach 86.65%, which is better than the former.

Figure 5 shows the performance of DSNN-GSM, BiLSTM and RNN-context. Under the same data set, DSNN-GSM maintains a stable accuracy rate of about 91.4% after 10 rounds of training. BiLSTM maintains a stable accuracy rate after 15 rounds of training, about 81.67%. The RNN-context shows that the rate of change is unstable.

In addition, this article compares RNN-context, Cover-5 [18], DRGD [19], LEAD [20] and other various neural network models on the data set for experimental comparison of ROUGE indicators. It can be seen from the results that the DSNN-GSM model proposed in this paper has a certain degree of improvement in these three ROUGE indicators (Table 2).

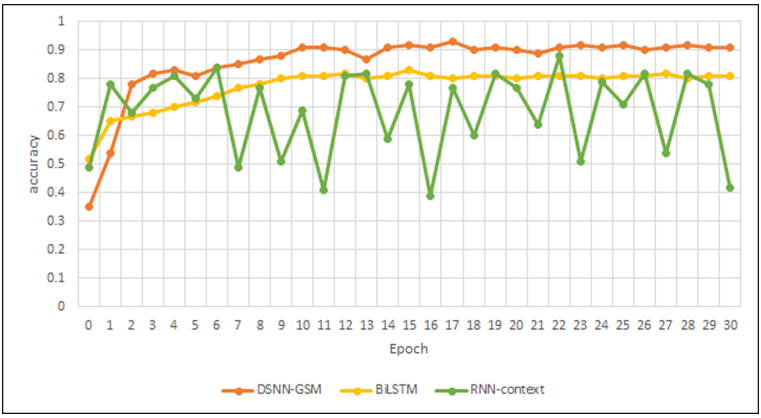

Fig. 5. Model performance comparison

Table 2. Comparison of rouge evaluation results

Methods	R-1	R-2	R-L
RNN-context	30.1	17.3	27.1
Cover-5	36.5	21.0	31.2
DRGD	37.2	24.1	34.3
LEAD	29.9	14.5	28.6
DSNN-GSM	38.9	25.8	34.8

5 Conclusion

This paper studies the traditional encoder-decoder model based on LSTM and analyzes its pros and cons: Although a long and short memory neural network is used to memorize the input content before the current input, if the input sequence is too long, it will still cause errors in the back propagation derivation; and the problem of coarse granularity of the attention mechanism. Based on the analysis of the above problems, a generative text summarization model based on the neural network of the document structure is proposed. It divides the complete text input sequence into a word encoding layer and a sentence encoding layer. Input one-hot code to Bidirectional GRU to generate word code, word code forms sentence code, sentence code generates text vector, and finally decodes. In attention, DSNN-GSM alleviates the problem of large derivative error in the back-propagation caused by long sequence; it introduces attention mechanism or multi attention mechanism in each level, which refines the attention granularity of the model, so that it can capture the key information in the article more accurately, and improve the accuracy of generating summary.

References

1. Jing, C.: Development and application of data science in the Internet plus big data Era. Civil Mil. Integr. **6**, 17–20 (2019)
2. Mahmud, M., Kaiser, M.S., McGinnity, T.M., et al.: Deep learning in mining biological data. Cogn. Comput. **13**, 1–33 (2021)
3. Mahmud, M., Kaiser, M.S., Hussain, A., Vassanelli, S.: Applications of deep learning and reinforcement learning to biological data. IEEE Trans. Neural Netw. Learn. Syst. **29**(6), 2063–2079 (2018)
4. Lin, J., Sun, X., Ma, S. and Su, Q.: Global encoding for abstractive summarization. Comput. Lang. **19**(6) 17 (2017)
5. Vaswani, A., et al.: Attention is all you need. In: 31st Conference on Neural Information Processing Systems (NIPS 2017) (2017)
6. Song, S., Huang, H., Ruan, T.: Abstractive text summarization using LSTM-CNN based deep learning. Multimedia Tools Appl. **78**(1), 857–875 (2018). https://doi.org/10.1007/s11042-018-5749-3
7. Dey, R., Salem, F.M.: Gate-Variants of Gated Recurrent Unit (GRU) Neural Networks. In: 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS) (2017)
8. Zhang, Y., Liu, Q., Song, L.: Sentence-state LSTM for text representation (2018)
9. Edmundson, H.: New methods in automatic extracting. J. Assoc. Comput. Mach. **16**(2), 264–285 (1969)
10. Lin, C.: ROUGE: a package for automatic evaluation of summaries. In: Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, Pennsylvania, ACL Press, pp. 74–81 (2004)
11. Gulli, A., Pal, S.: Deep Learning with Keras. Packt Publishing Ltd., Birmingham, United Kingdom (2017)
12. Rong, X.: word2vec parameter learning explained. arXiv preprint arXiv:1411.2738 (2014)
13. Hinton, G.E., Salakhutdinov, R.R.: Replicated softmax: an undirected topic model. Adv. Neural Inf. Process. Syst. **22**, 1607–1614 (2009)
14. Baldi, P., Sadowski, P.J.: Understanding dropout. Adv. Neural Inf. Process. Syst. **26**, 2814–2822 (2013)
15. Agarap, A.F.: Deep learning using rectified linear units (relu). arXiv preprint arXiv:1803.08375 (2018)
16. Wang, H.-C., Hsiao, W.-C., Chang, S.-H.: Automatic paper writing based on a RNN and the TextRank algorithm. Appl. Soft Comput. **97**, 106767 (2020). https://doi.org/10.1016/j.asoc.2020.106767
17. Sun, M.C., Hsu, S.H., Yang, M.C., Chien, J.H.: Context-aware cascade attention-based RNN for video emotion recognition. In: 2018 First Asian Conference on Affective Computing and Intelligent Interaction (ACII Asia) (2018)
18. Gong, Y., et al.: Research on text summarization model with coverage mechanism. J. Front. Comput. Sci. Technol. **13**(2), 205–213 (2019)
19. Li, P., et al.: Deep recurrent generative decoder for abstractive text summarization. In: Proceedings of the 22th Conference on Empirical Methods in Natural Language Processing, Pennsylvania, ACL Press, pp. 2091–2100 (2017)
20. Wasson, M.: Using leading text for news summaries: evaluation results and implications for commercial summarization applications. In: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, vol. 2, pp. 1364–1368 (1998)