ISSN 2616-5775 Vol. 6, Issue 9: 1-10, DOI: 10.25236/AJCIS.2023.060901

ASMAM: An Answer Summarization Mechanism Based on Multi-layer Attention Model

Jian Song[1,*], Meixuan Jiang[2]

[1]*Jiangsu Expressway Network Operation & Management Co., LTD, Nanjing, 210049, China*
[2]*The University of Warwick, Coventry, CV47AL, United Kingdom*
[*]*Corresponding author*

Abstract: *At present, deep learning technologies have been widely used in the field of natural language process, such as text summarization. In Community Question Answering (CQA), the answer summary could help users get a complete answer quickly. There are still some problems with the current answer summary mechanism, such as semantic inconsistency, repetition of words, etc. In order to solve this, we propose a novel mechanism called ASMAM, which stands for Answer Summarization based on Multi-layer Attention Mechanism. Based on the traditional sequence to sequence (seq2seq), we introduce self-attention and multi-head attention mechanism respectively during sentence and text encoding, which could improve text representation ability of the model. In order to solve "long distance dependence" of Recurrent Neural Network (RNN) and too many parameters of Long Short-Term Memory (LSTM), we all use gated recurrent unit (GRU) as the neuron at the encoder and decoder sides. Experiments over the Yahoo! Answers dataset demonstrate that the coherence and fluency of the generated summary are all superior to the benchmark model in ROUGE evaluation system.*

Keywords: *answer summarization, attention mechanism, encoder-decoder framework, recurrent neural network*

1. Introduction

We often use a variety of summarization techniques in our daily lives, such as newspaper abstract, TV news title, book review, shopping guide and movie trailer. With the rapid development of online publishing and e-government, users will be overwhelmed by a large amount of text information on the Internet. The automatic text summarization technology comes into being and becomes a new method to ease "information overload".

The automatic text summarization technology can condense the information of the text document into a short and easy-to-read summary [1], which enables users to understand the main idea of the text quickly, thereby saving a lot of time and energy. We could divide it into three steps: analysis, transformation, and synthesis [2]. In general, there are two automatic text summarization techniques: extractive summary and abstractive summary. The former picks important phrases or sentences directly from the source text and then combines them, while the latter rewrites and reorganizes the semantic of the source text and then generates a more concise and generalized text. In CQA, an open question always has multiple answers, but the community only picks one as the best answer according to its own criteria. For factual questions, this operating mechanism will work well. However, for open questions, the best answer may not be complete. According to statistics made by Liu et al. [3], no more than 48% of the 400 best answers in the 4 most popular directories in the Yahoo! Answers community are complete. In the answer list, other answers besides the best answer may contain the information required by the question, so answer summarization can help users find the most appropriate answer.

For this, Scholars' researches have made some progress, but there are still several problems: insufficient understanding of semantics, low sentence fluency, low accuracy and self-repetition of the summary. To solve these problems, we propose a new algorithm called ASMAM based on multi-layer attention mechanism, which uses the encoder-decoder model as the basic framework. And the encoder selects Gated Recurrent Unit (GRU) as the neuron. Moreover, we introduce different attention mechanisms in sentence and text encoding layer to enhance the model's representation ability. The decoder not only selects Gated Recurrent Unit (GRU) as the neuron, but also uses an internal attention mechanism to alleviate self-repetition of the summary. Specifically, we make following contributions:

ISSN 2616-5775 Vol. 6, Issue 9: 1-10, DOI: 10.25236/AJCIS.2023.060901

(1) We use Gated Recurrent Unit (GRU) as the neuron, which overcomes the "vanishing gradient" of Recurrent Neural Network (RNN).

(2) To improve the text representation of the model, we introduce self-attention and multi-head attention mechanism respectively during encoding. To avoid self-repetition of the summary, the decoder uses an internal attention mechanism.

2. Related work

In recent years, scholars from home and abroad have carried out a lot of researches and proposed their own algorithms. In general, algorithms can be divided into two categories: extractive text summarization and abstractive text summarization.

Extractive text summarization refers to selecting key words or phrases from the source text and combining them. Barzilay et al. [4] propose a new algorithm to calculate vocabulary chains in text, combining several important knowledge sources: WordNet synonyms dictionary, partial speech markers, grammatical parser to recognize noun phrases. The text summary is divided into three steps: cutting the source text, constructing a vocabulary chain, identifying strong vocabulary chains, and extracting important sentences from the text. Conroy et al. [5] regard the text summary as extracting sentences from the source text that summarize the main idea, and use hidden Markov model to evaluate whether the sentences in the source text should be included in the summary. Wan et al. [6] propose conditional Markov random walk and HITS model based on clustering according to the importance of different topics. They first use clustering algorithm to divide the sentences into several different topic groups, secondly use the model to calculate the score of the sentences in the document, and finally remove redundant sentences and select important sentences to form a summary. In order to evaluate the importance of sentences in the graph, Erkan et al. [7] propose the LexRank algorithm based on the adjacency matrix of cosine similarity within the sentence, which constructs a graph by taking sentence in the source text as node and the similarity between sentences as edge. The thicker the edges, the higher the similarity between the two connected sentences. The score of each sentence node in the graph is calculated from the number of edges of the node and the thickness of the edge. Finally, the sentence with the higher score is selected as the text summary. Nallapati et al. [8] propose RNN-based document summary sequence model, SummaRuNNer, which uses Bidirectional Gated Recurrent Unit (Bi-GRU) for word-level and sentence-level text representation. They see text summary as a sequence classification, which accesses each sentence of the document in turn and makes a binary decision based on whether it is placed in the final summary. In addition, a new training mechanism is also proposed, which does not need to extract labels. Zhou et al. [9] proposed a novel end-to-end neural network framework for the text summarization, which combines learning sentence score and sentence extraction. First a hierarchical encoder is used to encode sentences in the document, and then sentences are extracted one by one to generate a summary. The difference from previous work is that the sentence selection strategy is integrated into the scoring model. Fang et al. [10] propose Topic Aspect-Oriented Summarization (TAOS) model and introduce latent variables and group norms into the summary task, using a greedy algorithm to generate the summary. These factors describe different characteristics of a topic, which represents entity with capital letters. Based on this, different feature groups can be extracted, and then a common feature group is selected through a set of norm penalties and potential variables. Yasunaga et al. [11] propose a graph-based neural multi-document summary system using graph convolutional network for the relational graph, and sentence embedding obtained from recurrent neural network (RNN) are used as the feature of input node. Through multi-layer propagation, graph convolutional network generates high-level hidden sentence features. Finally, they use greedy heuristics to extract sentences. Heu et al. [12] propose FoDoSu model for multi-document summary, which mainly uses Flickr tag clusters to select important sentences and WordCluster-based HITS algorithm to calculate the contribution of each word in the frequency table. After analyzing the relevance and contribution of each word, they use the association graph to calculate the score of each sentence. Cao et al. [13] propose a sentence ranking method based on recursive neural network for multi-document summarization. The ordering of sentences is mainly accomplished by hierarchical regression, in which the relevance of sentences in the parse tree is evaluated. Based on word-level to sentence-level supervision, a recursive neural network is used to automatically learn the ranking features in the tree. They select important and non-redundant sentences to form a summary based on the sorted score of words and sentences. Ko et al. [14] propose an effective text summarization method that uses contextual information and statistical methods to extract important sentences. At first, two consecutive sentences were combined through a sliding window mechanism [15] to form a Bi-Gram Pseudo Sentence (BGPS). They then select many related BGPSs from the target document and divide

ISSN 2616-5775 Vol. 6, Issue 9: 1-10, DOI: 10.25236/AJCIS.2023.060901

each BGPS into two single sentences. Finally, the separated sentences are extracted and formed into a summary. Although the extractive text summarization algorithm is relatively mature and has some breakthroughs in terms of syntax and grammar, it lacks the understanding of text semantic information and the quality of the generated summary is generally not high.

Abstractive summarization refers to rewriting and reorganizing the semantics of the source text, which would generate new words or phrases. Banerjee et al. [16] develop an abstractive text digester. They first determine the most important document in a multi-document set, and align sentences in the document with sentences in other documents to form a similar sentence group. Then use the word graph structure to generate the shortest path from the sentences in each group. Finally, a new integer linear programming model is used to select sentences from the shortest path with the goal of maximizing the readability of abstract. The integer linear programming model proposed in this paper represents the path as a binary variable, and fully considers the length of the path, information score, and the quality of the language in the objective function. Moawad et al. [17] propose a method to generate a single document summary using rich semantic graph simplification techniques, which is mainly divided into three steps: creating rich semantic graphs for the source document, making a more abstract representation of the generated semantic graphs, and generating a summary. In the semantic graph, the nodes represent verbs or nouns in the source text and edges represent semantic and topological relationship between words. While abstracting the source semantic graph, heuristic algorithm is used to replace, delete, or merge nodes in the graph.

Although the abstractive text summary algorithm has improved the quality of the abstract to some extent, there are still problems such as insufficient semantic understanding of text, poor sentence fluency, and self-repetition of the summary.

3. Answer summarization mechanism

In CQA, there are generally multiple answers for an open question. Each answer may explain the question from a different perspective, but it is not complete. We can use the answer summary technology to integrate different answers. The answer summary refers to extracting highly relevant topic, low redundancy and diverse text from all the answers corresponding to the question. In order to facilitate subsequent research, we first merge multiple answers corresponding to each question into a single document and record it as D. Suppose D consists of m sentences, then $D=\{s_1, s_2, \ldots, s_m\}$, where $s_i\,(1<=i<=m)$ represents the i_{th} sentence. Suppose s_i consists of n_i words, then $s_i=\{w_1, w_2, \ldots, w_{n_i}\}$, where $w_j\,(1<=j<=n_i)$ represents the j_{th} word.

3.1 Work principle

In the process of answer summary, we introduce different attention mechanisms based on Seq2Seq model which is originally applied to machine translation and promoted in the fields of natural language and image process. This section first introduces the principle of the basic Seq2Seq model with recurrent neural network (RNN) as a carrier, and then further explains the principle of the Seq2Seq & Attention model [18].

In RNN, the input of a sequence is represented by $X=\{X_1, X_2 \ldots X_t\}$. At each time step t, the update formula of the hidden state $h_{<t>}$ is as follows:

$$h_{<t>}=f(h_{<t-1>}, x_t) \tag{1}$$

Where f is the non-linear activation function. The RNN predicts the next word by learning the probability distribution of the input sequence, and the output probability distribution of each time step t can be expressed as $P(x_t \mid x_{t-1}, \ldots, x_1)$. We use $softmax$ function to transform the output as follows:

$$p(x_{t,j}=1 \mid x_{t-1}, \ldots, x_1)=\frac{\exp(w_j h_{<t>})}{\sum_{j'=1}^{K}\exp(w_{j'} h_{<t>})} \tag{2}$$

ISSN 2616-5775 Vol. 6, Issue 9: 1-10, DOI: 10.25236/AJCIS.2023.060901

Where w_j is the number of rows in the weighted matrix W, and the probability of the input sequence x is calculated by combining these probabilities, as shown in the following formula:

$$p(x) = \prod_{t=1}^{T} p(x_t \mid x_{t-1}, ..., x_1) \tag{3}$$

The basic Seq2Seq model consists of two parts, Encoder and Decoder. When the encoder finishes encoding, it outputs an intermediate semantic vector C which is then decoded. The RNN-based encoder-decoder framework uses another independent recurrent neural network RNN while decoding. At the time t, the update of the hidden state of the decoder is as follows:

$$h_{<t>} = f(h_{<t-1>}, y_{t-1}, c) \tag{4}$$

The conditional probability of predicting the next word can be expressed as:

$$P(y_t \mid y_{t-1}, y_{t-2}, ..., y_1, c) = g(h_{<t>}, y_{t-1}, c) \tag{5}$$

In this basic encoder-decoder model, the maximum likelihood condition probability can be used while designing the loss function, as shown in the following formula:

$$\max_{\theta} \frac{1}{N} \sum_{n=1}^{N} \log p_\theta(y_n \mid x_n) \tag{6}$$

Where θ is the model's parameters and (x_n, y_n) represents input and output sequences. In order to enhance the model's ability to represent and generalize, based on the Seq2Seq model, Bahdanau et al. [18] introduce the attention mechanism. The following uses machine translation task as an example to explain the principle of the Seq2Seq & Attention mechanism, as shown in the following Figure 1.

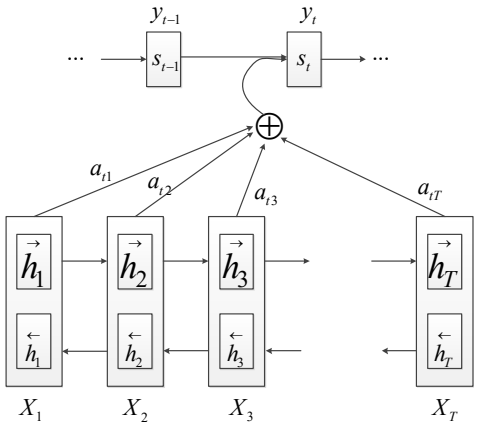

Figure 1: The structure of NMT

After the attention mechanism is introduced into the Seq2Seq model, the output y_i of each time step i can be calculated using the following formula:

$$p(y_i \mid y_1, ..., y_{i-1}, x) = g(y_{i-1}, s_i, c_i) \tag{7}$$

Where s_i is the hidden state of the time step i, which can be calculated by the following formula:

$$s_i = f(s_{i-1}, y_{i-1}, c_i) \tag{8}$$

The context vector c_i depends on the translation sequence $(h_1, ..., h_{T_x})$. Each note h_i contains all the information in the entire input sequence, especially the i_{th} word of the input sequence. The context vector can be represented by the weighted sum of the annotation as follows:

ISSN 2616-5775 Vol. 6, Issue 9: 1-10, DOI: 10.25236/AJCIS.2023.060901

$$c_i = \sum_{j=1}^{T_x} a_{ij} h_j \tag{9}$$

Where the formula for calculating the weight a_{ij} of each annotation h_i is as follows:

$$\begin{aligned} a_{ij} &= \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})} \\ e_{ij} &= a(s_{i-1}, h_j) \end{aligned} \tag{10}$$

Where $e_{ij} = a(s_{i-1}, h_j)$ is used to measure the correlation between the input of position j and the output of position i.

3.2 Encoder

The traditional recurrent neural network (RNN) will forget the previous input information, which is only biased towards the memory of the last moment. To solve this problem, we use bi-directional gated recurrent unit (Bi-GRU) while designing the encoder. Compared with the long short-term memory (LSTM), the gated recurrent unit (GRU) has a simpler structure and fewer parameters, and can alleviate overfitting.

When representing the entire answer text, we draw on the idea of the hierarchical network(HAN) [19]. We first encode the words in the sentence, and then encode the sentences in the text. In order to extract the semantic information of the sentence in parallel, we use both forward GRU and backward GRU to encode the words in the sentence. The input of Bi-GRU is two GRUs with opposite directions at time t, and the output is their splicing. The formula for calculating its hidden state h_t is as follows:

$$\begin{aligned} \overrightarrow{h_t} &= \overrightarrow{f_{GRU}}(v_t, \overrightarrow{h_{t-1}}) \\ \overleftarrow{h_t} &= \overleftarrow{f_{GRU}}(v_t, \overleftarrow{h_{t-1}}) \\ h_t &= [\overrightarrow{h_t}, \overleftarrow{h_t}] \end{aligned} \tag{11}$$

Where f_{GRU} represents the GRU operation, h_{t-1} is the hidden state of the Bi-GRU at time $t-1$, and v_t is the source text sequence data. $\overrightarrow{h_t}$ is the output of forward propagation, and $\overleftarrow{h_t}$ is the output of backward propagation. For the convenience of calculation, the states in the hidden layer are combined, and the hidden layer is represented as follows:

$$H_{ws} = (h_1, h_2, ...h_n) \tag{12}$$

Where $H_{ws} \in \mathbb{R}^{n \times 2r}$, r is the number of hidden units in the unidirectional GRU. Because each word in the sentence contributes differently to the semantic representation of the sentence, we introduce the attention mechanism. In order to fully learn the dependency relationship between words in a sentence, we choose the self-attention mechanism [20], as shown in the following formula:

$$a_{ws} = soft\max(w_2 \tanh(w_1 H_{ws}^T)) \tag{13}$$

Where a_{ws} indicates the importance of the word, w_2 and w_1 are parameters, and $softmax$ is used to normalize weighted sum of the attention. The attention vector a_{ws} and the hidden state h_t in the encoder are summed to obtain a vectorized representation s_i of the sentence at time t, as shown in the following formula:

$$s_i = a_{ws} H_{ws} \tag{14}$$

While encoding at the document level, the vectorized sequence of sentences $S = \{s_1, s_2, s_3..., s_m\}$ is used as input. The encoding process is similar to sentence-level encoding. We again use Bi-GRU to encode sentences in the text and stitch forward and backward output states, which is as follows:

ISSN 2616-5775 Vol. 6, Issue 9: 1-10, DOI: 10.25236/AJCIS.2023.060901

$$\begin{aligned}\overrightarrow{h_i} &= \overrightarrow{f_{GRU}}(s_i, \overrightarrow{h_{i-1}}) \\ \overleftarrow{h_i} &= \overleftarrow{f_{GRU}}(s_i, \overleftarrow{h_{i+1}}) \\ h_i &= [\overrightarrow{h_i}, \overleftarrow{h_i}]\end{aligned} \tag{15}$$

To facilitate calculation, we merge the hidden states in the GRU as shown in the following formula:

$$H_{sd} = (h_{i1}, h_{i2}, ...h_{im}) \tag{16}$$

Where $H_{sd} \in \mathbb{R}^{m \times 2r}$, r is the number of hidden units in the unidirectional GRU. Each sentence in the document shows different importance to the representation of the entire document, so we also introduce an attention mechanism while encoding the document. In order to achieve parallel computing and improve the training speed of the model, we choose to use the multi-head attention mechanism. The main idea of the multi-head attention mechanism is to make the model obtain richer contextual sentence information by using multiple self-attention operations. The calculation formula for a single self-attention is as follows:

$$\begin{aligned}a &= [soft\max(w^T \tanh(H_{sd}))]^T . H_{sd} \\ h &= \tanh(a)\end{aligned} \tag{17}$$

Where w is parameter, and we can concatenate h to get the representation of the answer text, which is as follows:

$$D_{ans} = w_s Concat(h_1, h_2, ..., h_k) \tag{18}$$

3.3 Decoder

The use of hierarchical attention mechanism at the encoder side greatly improves the model's ability to represent. If the recurrent neural network (RNN) or its variant is simply used at the decoder side, the decoder will have the phenomenon of word duplication due to its hidden state. In order to solve this problem, we use the gated recurrent unit (GRU) at decoder side while introducing the internal attention mechanism, which could achieve the integration of previous historical decoding sequence information into the current decoder, and constantly reviews the previous decoding steps. The hidden state s_t of the decoder with attention mechanism at time t is determined by three factors : the output s_{t-1} of the GRU network at time $t-1$, the input y_t of the decoder at time t, and the context vector c_t, which is as follows:

$$s_t = f(s_{t-1}, y_t, c_t) \tag{19}$$

Where the context vector c_t contains the output of all previous moments, but the value of the attention weight is different at each moment. The calculation formula is as follows:

$$c_t = \sum_{i=1}^{t-1} a_{ti} s_i \tag{20}$$

Where s_t represents all the decoding information in the past, and a_{ti} represents the assigned attention weight. The calculation formula is as follows:

$$a_{ti} = \frac{\exp(e_{ti})}{\sum_{i=1}^{t-1} \exp(e_{ti})} \tag{21}$$

Where e_{ti} indicates the correlation between the output of the decoder at time i and the entire decoding process at time t

ISSN 2616-5775 Vol. 6, Issue 9: 1-10, DOI: 10.25236/AJCIS.2023.060901

4. Experimentation and results

4.1 Dataset

The dataset is from the paper [21], which is a benchmark dataset provided by the Yahoo! Answers community. The questions in CQA are broadly divided into two categories: factual and open. Factual questions generally have standard answers, so they are not worth having a research. Tomasoni et al. [21] filtered the benchmark dataset, removed factual questions and kept open questions in the dataset according to certain rules. After processing, the final dataset contains 100 questions and corresponding 361 answers, a total of 2793 sentences and 275 standard answer summaries.

4.2 Evaluation metrics

There is no perfect abstract for a text document, and the judgment of the quality of the abstract is highly subjective, so the evaluation of the abstract is a difficult task. There are two ways to evaluate the text summary: one is external evaluation, the other is internal evaluation. Currently, the ROUGE standard that evaluates the quality of the summary by counting the number of overlapping units between the machine and manual summary is commonly used, which is proposed by Lin et al. [22] in 2004. In this paper, we use ROUGE-N and ROUGE-L to evaluate the quality of the answer summary. The calculation formula of ROUGE-N is as follows:

$$ROUGE-N=\frac{\sum_{gram_N\in Y} C_M(gram_N)}{\sum_{gram_N\in Y} C_Y(gram_N)} \tag{22}$$

Where N is 1 and 2, respectively, the numerator represents the number of co-occurrence of N-grams in the generated answer and reference summary, and the denominator represents the number of N-grams in the reference summary. The formula for calculating ROUGE-L is as follows:

$$R_{lcs}=\frac{\sum_{i=1}^{n} LCS_N(r_i,Y)}{m} \tag{23}$$

Where n represents the number of sentences in the reference summary, m is the number of words, and Y is the generated summary.

4.3 Experimental results

In our experiments, we select ADAM as optimizer and set the learning rate to 0.001 while updating parameters. In order to verify the performance of our proposed model, we conducted experiments over the Yahoo! Answers dataset.

(1) The analysis of neuron

In order to verify the effect of the gated recurring unit (GRU) used in the model, we replace the neuron with LSTM and RNN, respectively. The experimental results are shown in the following Figure 2:

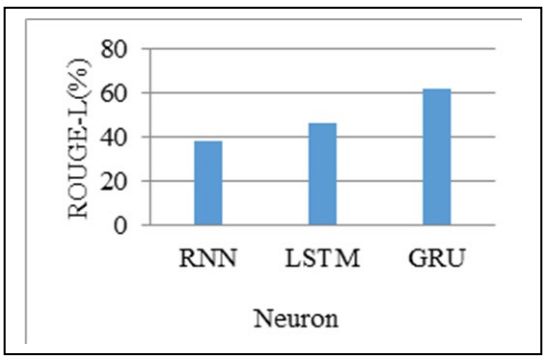

Figure 2: Experimental results of different neurons

After using GRU, the performance of the model has been significantly improved, indicating that the

ISSN 2616-5775 Vol. 6, Issue 9: 1-10, DOI: 10.25236/AJCIS.2023.060901

neural network unit prioritizes GRU while performing answer summary. LSTM is more effective than traditional RNN in dealing with "vanishing gradient", but it has too many parameters and is easy to appear overfitting. The GRU further optimizes the structure of the LSTM, making the model parameters less and alleviating the problem of overfitting.

(2) The analysis of the effect of multi-layer attention mechanism

Figure 3 is a comparative analysis of the effect of the multi-layer attention mechanism. In order to analyze the effectiveness of the mechanism, the answer summary algorithm based on the multi-layer attention mechanism (ASMAM) proposed in this paper is compared with the simple GRU-based answer summary algorithm (AS). The results are shown in Figure 3 below:

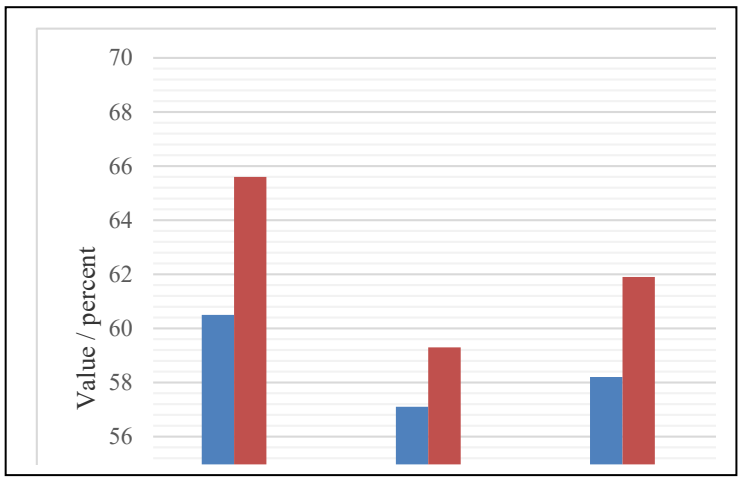

Figure 3: Comparative analysis of the effects of the multi-layer attention mechanism

As can be seen from Figure 3, the performance of the ASMAM model is better than AS in ROUGE indicator system. This is because ASMAM uses different attention mechanisms in the sentence and text encoding layer, which not only effectively eliminates most of the noise in the text, but also further enhances the model's representation capabilities. The internal attention mechanism is introduced to avoid self-repetition of the abstract at the decoder side.

(3) Head count analysis of multi-head attention mechanism

Figure 4 shows the performance of different number of heads in multi-head attention mechanism. As can be seen from it, the performance is best when $k=6$. In general, when the value of k is greater than 6, the performance of the model starts to decline. The reason is that when the value of k is greater than the threshold, the model will appear overfitting, so we set k to 6.

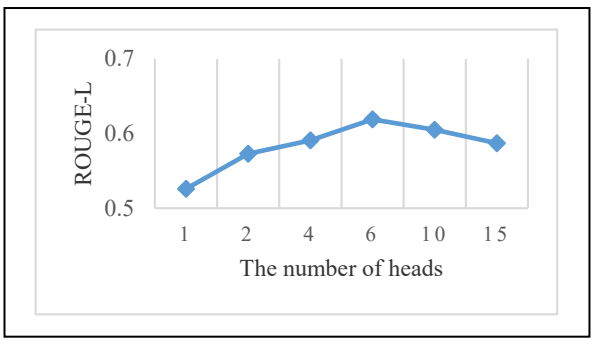

Figure 4: ROUGE-L value for different head counts

(4) Time overhead of different models

In Figure 5, we compare the average time cost of three different algorithms. The ASMAM has the smallest time overhead while the time overhead of TextRank is the largest. The reason is that the TextRank uses a double loop mechanism to speed up the convergence. The first cycle traverses all connected nodes of the node in the graph; the second cycle calculates the out-degree of the connected node . Therefore, this algorithm requires a lot of time. ASMAM takes a lot of time for model training. But after the model is trained, it can be used directly.

ISSN 2616-5775 Vol. 6, Issue 9: 1-10, DOI: 10.25236/AJCIS.2023.060901

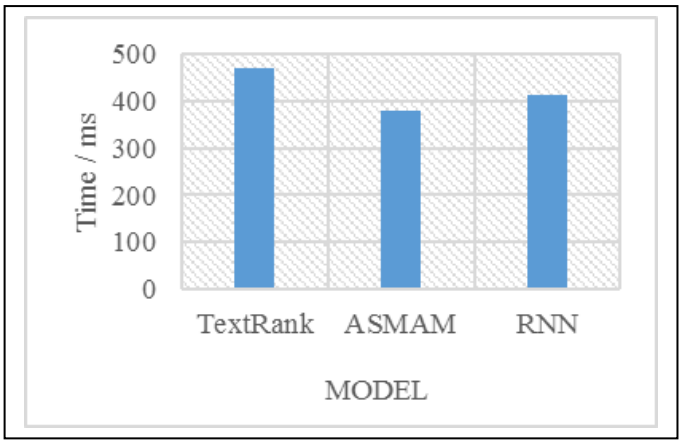

Figure 5: Time overhead of different models

5. Conclusion and future work

In this paper, we propose a seq2seq model based on multi-layer attention mechanism to generate answer summary. In order to solve the gradient disappearance of the traditional recurrent neural network (RNN) and the over-fitting of the Long Short-Term Memory (LSTM), we adopt a bidirectional GRU while encoding the source answer text. Drawing on the idea of hierarchical attention network, according to the characteristics of sentence and document coding, self-attention mechanism [20] and multi-head attention mechanism [23] are used at the sentence and document level, respectively.

During decoding, we introduce an internal attention mechanism based on single-layer GRU to enhance the generalization ability of the model and reduce the redundancy of the text in the generated summary. When encoding answer text, we use hierarchical network and introduce attention mechanism, which greatly enhances the model's representation ability to a certain extent. However, the semantic dependencies between the answer texts are not fully considered. Our proposed model is based on encoder-decoder framework and selects GRU as the neuron. While training the model, a large amount of sample data is required. In the future, we could try to enhance the model's text semantic representation capabilities with the syntactic and grammatical analysis techniques in natural language process, thereby reducing the amount of training data. At present, the Rouge indicator is commonly used in the field of automatic text summarization. But it has some limitations, which cannot evaluate the quality of the summary from the perspective of semantic similarity.

References

[1] M. Gambhir and V. Gupta (2017). Recent automatic text summarization techniques: a survey. Artificial Intelligence Review, vol.47, no.1, pp.1-66.
[2] U. Hahn and I. Mani (2000). The challenges of automatic summarization. Computer, vol.33, no.11, pp.29-36.
[3] Y. D. Liu, J. Bian and E. Agichtein (2008). Predicting information seeker satisfaction in community question answering. Proceedings of The 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.483-490.
[4] R. Barzilay and M. Elhadad (1997). Using lexical chains for text summarization. Proceedings of The ACL/ EACL 1997 Workshop on Intelligent Scalable Text Summarization, pp. 10-17.
[5] J. M. Conroy and D. P. O'Leary (2001). Text summarization via hidden Markov models. Proceedings of The 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.406-407.
[6] X. J. Wan and J. W. Yang (2008). Multi-document summarization using cluster-based link analysis. Proceedings of The 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.299-306.
[7] G. Erkan and D. R. Radev (2004). Lex Rank: graph-based lexical centrality as salience in text summarization. Journal of Artificial Intelligence Research, vol 22, no.2004, pp.457-479.
[8] R. Nallapati, F. F. Zhai and B. W. Zhou (2017). Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. Proceedings of The 31st AAAI Conference on Artificial Intelligence, pp.3075–3081.

ISSN 2616-5775 Vol. 6, Issue 9: 1-10, DOI: 10.25236/AJCIS.2023.060901

[9] Q. Y. Zhou, N. Yang, F. R. Wei, S. H. Huang, M. Zhou and T. J. Zhao (2018). Neural document summarization by jointly learning to score and select sentences. Proceedings of The 56th Annual Meeting of the Association for Computational Linguistics, pp.654-663.

[10] H. Y. Fang, W. M. Lu, F. Wu, Y. Zhang, X. D. Shang, J. Shao and Y. T. Zhuang (2015). Topic aspect-oriented summarization via group selection. Neurocomputing, vol.149, no.2015, pp.1613–1619.

[11] M. Yasunaga, R. Zhang, K. Meelu, A. Pareek, K. Srinivasan and D. Radev (2017). Graph-based Neural Multi-Document Summarization. Proceedings of the SIGNLL Conference on Computational Natural Language Learning, pp.452-462.

[12] J. U. Heu, I. Qasim and D. H. Lee (2015). FoDoSu: Multi-document summarization exploiting semantic analysis based on social Folksonomy. Information Processing & Management, vol.51, no.1, pp.212-225.

[13] Z. Q. Cao, F. R. Wei, L. Dong, S. J. Li and M. Zhou (2015). Ranking with recursive neural networks and its application to multi-document summarization. Proceedings of The 29th AAAI Conference on Artificial Intelligence, pp.2153–2159.

[14] Y. Ko and J. Seo (2008). An effective sentence-extraction technique using contextual information and statistical approaches for text summarization. Pattern Recognit Letters, vol.29, no.9, pp.1366-1371.

[15] Y. Ko and J. Seo (2004). Learning with unlabeled data for text categorization using a bootstrapping and a feature projection technique. Proceedings of The 42nd Annual Meeting of the Association for Computational Linguistics, pp. 255-262.

[16] S. Banerjee, P. Mitra and K. Sugiyama (2015). Multi-document abstractive summarization using ILP based multi-sentence compression. Proceedings of The 24th International Conference on Artificial Intelligence, pp.1208-1214.

[17] I. F. Moawad and M. Aref (2012). Semantic graph reduction approach for abstractive Text Summarization. Proceedings of The 7th International Conference on Computer Engineering & Systems (ICCES), pp.132-138.

[18] D. Bahdanau, K. Cho and Y. Bengio (2014). Neural Machine Translation by Jointly Learning to Align and Translate. Computer Science.

[19] Z. Yang, D. Yang, C. Dyer, X. D. He, A. Smola and E. Hovy (2016). Hierarchical Attention Networks for Document Classification. Proceedings of The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp.1480-1489.

[20] Z. H. Lin, M. W. Feng, C. N. Santos, M. Yu, B. Xiang, B. W. Zhou and Y. Bengio (2017). A Structured Self-attentive Sentence Embedding. Proceedings of The 5th International Conference on Learning Representations(ICLR), pp.1-15.

[21] M. Tomasoni and M. Huang (2010). Metadata-aware measures for answer summarization in community Question Answering. Proceedings of The 48th Annual Meeting of the Association for Computational Linguistics, pp.760-769.

[22] C. Y. Lin (2004). ROUGE: A Package for Automatic Evaluation of summaries. Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004).

[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser and I. Polosukhin (2017). Attention is all you need. Proceedings of Advances in Neural Information Processing Systems, pp.5998–6008.