ISSN 2616-5775 Vol. 6, Issue 1: 77-82, DOI: 10.25236/AJCIS.2023.060112

Research on keyword extraction based on abstract extraction

Zihao Yan

Nanjing University of Finance & Economics, Nanjing, China, 210023

Abstract: *In order to improve the accuracy of text keyword extraction, this paper combined with the relevant methods of abstract extraction, aiming to extract key sentences through the abstract extraction method, and then optimize the effect of keyword extraction. The experimental results show that the effect of the keyword extraction algorithm on the text after abstract extraction is improved by 6.92 percentage points.*

Keywords: *Keyword Extraction, Abstract Extraction, Supervised*

1. Introduction

The keywords are usually one or more words or phrases that can describe the subject information of the document [1]. For a text, keywords, as a feature that can express the main semantic content of the text, can effectively reflect the user's interest in acquiring the semantic content of the relevant text and understanding the article. With the rapid development of the Internet, text is produced at a very fast speed, which makes the manual annotation of keywords not only slow, but also not efficient. Because of this, automatic keyword extraction technology has emerged at the right moment, which refers to the key words of documents obtained through certain rules or mathematical models. This technology is known as keyword extraction in the field of text mining, and automatic indexing in the field of information retrieval [2].

Abstract is a summary of the contents of the text, including academic papers, and is a short article that concisely and accurately describes the important contents of the article with the purpose of providing a summary of the contents of the document.

For a document, it is composed of multiple sentences. Each sentence contains multiple keywords. The relationship between keywords and sentences is very close. The calculation of keyword weight depends on some features of the keyword itself, such as word frequency and part of speech. On the other hand, the influence of the sentence on the keyword weight cannot be ignored. The importance of the keyword also determines the importance of the sentence. For example, if a keyword appears in an important sentence of a document, the probability of the keyword being more important is relatively high, and the weight value of the keyword should also be increased accordingly.

For the long document, it contains a large amount of information, but some of the information may be the expansion and extension of the key words. These information can enrich the content of the text, make the article look very rich, but will affect the quality of the keyword extraction. For example, an article introducing food may describe the origin and conditions of food. When extracting keywords for this article, if the author spends a large amount of time writing about the place of origin, it is very likely that the extracted keywords will focus on reflecting the place of origin while ignoring the main content of food. Therefore, based on the relevance of the relationship between sentences and keywords, this paper considers using extractive abstract technology to reduce the long text, removing the unimportant sentences and repetitive sentences in the long text, and then preserving the important sentences in the text, and observing the impact of the reduced sentences on keyword extraction.

2. Related work

2.1 Keyword extraction technology

Keyword extraction technology is mainly divided into two types, one is based on unsupervised methods, and the other is based on supervised methods. There are two main unsupervised methods. The

ISSN 2616-5775 Vol. 6, Issue 1: 77-82, DOI: 10.25236/AJCIS.2023.060112

first is based on statistics, such as N-gram [3], TF-IDF [4], word frequency [5], word co-occurrence [6]. The second is based on graphs, such as TextRank [7], TopicRank [8],. For the supervised method, it is mainly necessary to give a labeled data set in advance to train the model, especially with the development of machine learning and deep learning, which adds new vitality to it. The main models used include SVM, RNN, Bert model, etc.

2.2 Extractive abstract technology

The classification of extraction technology is similar to that of keyword extraction technology, which is mainly divided into two types: unsupervised and supervised. Unsupervised is mainly based on statistical information, such as the length and position of sentences. For the supervised method, some scholars expand the relationship between sentences by introducing word nodes, and model the abstract in the form of graph. As early as 2004, graph structure was used for the abstract task: LexRank [9] and TextRank [7] take sentences as nodes, build edges according to the similarity between the features of sentences, sort the importance of nodes in the way of unsupervised iteration, and select the most important nodes as the summary. However, it is not easy to select an appropriate threshold for a graph with edge built by similarity. Recently, some efforts have tried to determine whether the sentence nodes should be connected by manually defined features (such as ADG [10]), or to construct diagrams (such as RST [11]) through rhetorical devices or common reference relationships.Some scholars try to use the full connection graph transformer directly to let the model learn the edge weight by itself. On the other hand, the BERT model is a pre-training model proposed by Google in 2018. It is based on the Transformer mechanism, has strong language representation ability and feature extraction ability, and has achieved excellent performance in many NLP benchmark tasks. Some scholars have built abstract extraction models based on BERT model, such as BertSum [12] model.

3. Method

The abstract extraction model used in this paper is BertSum, which is a supervised abstract extraction model based on the BERT model. The keyword extraction model uses DeepCT [13], which is also based on the feature extraction ability of the BERT model, and the keyword weight constructed on this basis to calculate the weight of words to extract keywords.

3.1 BertSum

BertSum model is mainly composed of sentence coding layer and abstract judgment layer.

3.1.1 Sentence coding layer

The word embedding of each word in the document is obtained through the BERT model. Due to the pre-training mechanism of the BERT model, its output vector is the word embedding of each word. Unlike the text classification task using the BERT model, the input of the abstract extraction task is mainly the article and the category of the article. The abstract extraction task needs to separate multiple sentences and determine whether each sentence is a summary.

3.1.2 Abstract judgment layer

This layer is to embed the words obtained from the sentence coding layer into the code and select and judge them through the Transfomer layer. The purpose is to further extract the document-level features focused on the abstract task, and then connect a full connection layer of the Sigmoid activation function, so as to score each sentence, and finally select the best Top-N sentences as the document abstract. For each sentence, calculate the final prediction score $\overline{Y_i}$ the loss function of the model uses the prediction tag $\overline{Y_i}$ relative to real label Y_i binary cross entropy of.

3.2 DeepCT

Through the abstract extraction in section 3.1, the abstract in the document is obtained. Although the document length is relatively reduced, the important content in the article is still retained. The next step is to calculate the word weight from the text abstract, and use the words with higher weight as the key words of the text.The DeepCT model can be divided into two parts [13]:

The word embedding of upper and lower cultures is generated through BERT model. In order to estimate the importance of a word in a text, the key problem is to generate features that can describe the

ISSN 2616-5775 Vol. 6, Issue 1: 77-82, DOI: 10.25236/AJCIS.2023.060112

relationship between a word and the text context. Here, BERT model is used as the extraction tool of keyword context feature.

Linear regression model is used to predict the weight of words by embedding words from different cultures. The context word embedding generated by the BERT model is a feature vector that contains the syntactic and semantic information of the word in a specific context. The DeepCT model changes these feature vectors linearly to form the importance score of the word weight.

4. Experiment

4.1 Dataset

The dataset used in this paper is a reading comprehension dataset MS-MARCO (Microsoft Machine Reading Comprehension) based on large-scale real scene data proposed by Microsoft. [14] This dataset is generated based on the real search queries in Bing search engine and Cortana intelligent assistant, including 1 million queries, 8 million documents and 180000 manually edited answers. Based on the MS-MARCO dataset, this paper studies the problem of retrieving and sorting documents in all data sets for a given query.

4.2 Evaluation indicators

4.2.1 ROUGE

The evaluation index used in this abstract extraction is ROUGE (Recall-Oriented Understudy for Gisting Evaluation) index, which is a set of indicators for evaluating automatic summary and machine translation. It compares the automatically generated summary or translation with a group of reference summaries (usually manually generated) and calculates the corresponding score to measure the "similarity" between the automatically generated summary or translation and the reference abstract. The ROUGE indicators used in this paper are ROUGE-N and ROUGE-L.

ROUGE-N actually calculates the recall rate after splitting the results generated by the model and the standard results by N-gram. Among them, n-gram splitting refers to dividing the text into a group of n words. The calculation formula of ROUGE-N is as follows:

$$Rough\text{-}N = \frac{\sum_{S \in ReferenceSummaries} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in ReferenceSummaries} \sum_{gram_n \in S} Count(gram_n)} \tag{1}$$

The numerator represents the sum of the number of matches between the real results of all samples and the predicted results after splitting by N-gram; The denominator represents the true result of all samples, and is the sum after splitting by N-gram. The ROUGE-L index is similar to ROUGE-N index, and the longest common subsequence is used. The formula is as follows:

$$R_{lcs} = \frac{LCS(X,Y)}{m} \tag{2}$$

$$P_{lcs} = \frac{LCS(X,Y)}{n} \tag{3}$$

$$F_{lcs} = \frac{(1+\beta^2)\ R_{lcs} P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}} \tag{4}$$

Among them, $LCS(X,Y)$ is the length of the longest common subsequence of X and Y, m and n respectively represent the length of the manual summary and the machine-generated summary (generally refers to the number of words contained), R_{lcs} and P_{lcs} respectively represents recall rate and accuracy rate, and finally R_{lcs} it means ROUGE-L, β it is a super parameter, because the recall rate is often more important when judging the quality of the machine summary β the value is generally set to a large number.

4.2.2 BM25

BM25 is an algorithm used to calculate the correlation between a sentence and a document in the

ISSN 2616-5775 Vol. 6, Issue 1: 77-82, DOI: 10.25236/AJCIS.2023.060112

field of information indexing. Its principle is very simple: divide the input sentence into words, then calculate the correlation between each word in the sentence and the document separately, and then carry out weighted summation to get the correlation score between the sentence and the document. This score is BM25 value. When actually calculating the relevance of a word to a document, it is usually regarded as a sentence with only one word, and then its BM25 value is calculated. The calculation formula is as follows:

$$Score(Q,d)=\sum_{i}^{n}W_iR(q_i,d) \tag{5}$$

In the above formula, W_i represents the weight, that is, the inverse document frequency value of the word. The calculation formula is as follows:

$$W_i=IDF(q_i)=\log(\frac{N-n(q_i)+0.5}{n(q_i)+0.5}) \tag{6}$$

Where, N represents the total number of texts, and $n(q_i)$ is the number of documents containing the word q_i. From the formula, we can see that the larger n(qi) is, the larger the denominator is, and the smaller the numerator is, that is, the smaller the W_i value is. Because if a word appears in multiple documents, it can be said to a certain extent that it should be a relatively common word. It exists in any document and cannot reflect the particularity of the word in a specific document. Therefore, it is given a smaller W_i value.

$R(q_i,d)$ represents the correlation score between word q and document d. The calculation formula is as follows:

$$R(q_i,d)=\frac{f_i\cdot(k_1+1)}{f_i+K}\cdot\frac{qf_i\cdot(k_2+1)}{qf_i+k_2} \tag{7}$$

$$K=k_1\cdot(1-b)+b\cdot\frac{dl}{avgdl} \tag{8}$$

Among them, k1, k2 and b are adjustment factors, which are generally set according to experience, f_i expressions q_i frequency of occurrence in document d, qf_i for words q_i the frequency of occurrence in the sentence. dl is the length of document d, and avgdl is the average length of all documents in document set D.

4.2.3 MRR

The evaluation index used in the retrieval task in this paper is Mean Reciprocal Rank (MRR), which is an index to evaluate the search algorithm. For the first result matching, the score is 1, the second matching score is 0.5, and the nth matching score is 1/n. If the score of the unmatched sentence is 0, the final score is the sum of all scores. The calculation formula is as follows:

$$MRR=\frac{1}{Q}\sum_{i=1}^{|Q|}\frac{1}{rank_i} \tag{9}$$

Among them, $|Q|$ is the number of users, $rank_i$ for the ith user, the first item in the recommendation list appears in the real result.

4.3 Abstract extraction experiment

Because of the large number of documents in the MS-MARCO dataset, this paper selects the first 700000 documents of the MS-MARCO dataset and the retrieval problems containing these documents as the experimental dataset for subsequent experiments.First, the MS-MARCO document will be extracted through the trained BertSum model. In order to improve the efficiency of model prediction, the document with less than 3 sentences will be kept unchanged.

4.4 Keyword extraction experiment

In order to verify whether abstract extraction can improve the effect of keyword extraction, this paper adopts the method of comparative test. The original MS-MARCO text and the MS-MARCO text

ISSN 2616-5775 Vol. 6, Issue 1: 77-82, DOI: 10.25236/AJCIS.2023.060112

after abstract extraction are respectively put into the DeepCT model to predict the weight value of each text word.

Then, the correlation between the text and the search problem is calculated according to the weight of the obtained text keywords. The calculation method is BM25 algorithm. It can be seen from Section 4.2.2 that the BM25 algorithm needs to know the frequency of the words appearing in the search problem in the text.While the predicted text weight is mostly between 0-1. Therefore, in order to ensure that the frequency value is an integer, this paper will amplify the predicted weight value by 100 times and then round it to the whole number to simulate the effect of the frequency value to facilitate the calculation of correlation. Finally, the text corresponding to the search question is sorted according to the relevance score from high to low, and the MRR index value is finally calculated. There are two parameters k and b in the calculation of correlation indicators that need to be specified manually. In this paper, k values are 11, 12, 13, and b values are 0.8, 0.9. The MRR values are shown in Table 1:

Table 1: Analysis of related experimental results evaluation indicators

Parameter	k	11	12	13	11	12	13
	b	0.8	0.8	0.8	0.9	0.9	0.9
MRR@10	DeepCT	0.224	0.224	**0.225**	0.221	0.222	0.220
	BertSum+DeepCT	0.234	0.236	0.237	0.239	0.239	**0.240**

It can be found that no matter how k and b change, the MRR value of the text after summary processing is higher than that of the text without any processing. Among them, the maximum MRR of the text after summary processing is 0.240, which is 6.66% higher than the maximum MRR of the text without processing is 0.225. The reason may be that abstract extraction can reduce the interference of non-key content on the text to a certain extent, thus improving the effect of keyword extraction. It shows that refining key sentences can improve the effect of keyword extraction.

In order to show the validity of the word weight obtained by the DeepCT model, this paper uses different methods to calculate the word weight, such as TF, TextRank, Doc2Query, and calculates the correlation between the text and the search problem according to the different word weight, and obtains the MRR index. The specific results are shown in Table 2:

Table 2: MRR indexes of retrieval problems using three traditional word weights and DeepCT word weights

Word weight calculation method	TF	TextRank	Doc2Query	DeepCT	BertSum+DeepCT
MRR@10	0.191	0.13	0.221	0.225	**0.241**

It can be found that the word weight obtained by using the DeepCT method is superior to the traditional word weight calculation method in the MS-MARCO dataset retrieval problem, and the MRR effect is 15.7%, 73.0% and 1.8% higher than that obtained by using the three weights.

5. Conclusion

This paper proposes a method of extracting abstract first, summarizing long text concisely, and then extracting keywords. Taking 700000 pieces of data from the MS-MARCO dataset as the verification set, and using the above method to extract keywords from the text, we can find that the article after abstract extraction can achieve better results. Compared with several benchmark models, the overall effect is better.

In fact, this paper only uses one abstract extraction model and one keyword extraction model. Later, we will consider the impact of different abstract extraction models on keyword extraction models.

References

[1] Turney, P.D. (2000) Learning algorithms for keyphrase extraction. Information Retrieval, 2(4), 303-336.
[2] Zhang, C.Z. (2007) Review and Prospect of Automatic Indexing Research. New Technology of Library and Information Service, 11, 33-39.
[3] Cohen, J.D. (1995) Highlights: language- and domain-independent automatic indexing terms for abstracting. Journal of the American Society for Information Science, 46(3), 162-174.
[4] Salton, G., Yang, C.S., and Yu, C.T. (1975) A Theory of Term Importance in Automatic Text

ISSN 2616-5775 Vol. 6, Issue 1: 77-82, DOI: 10.25236/AJCIS.2023.060112

Analysis. Journal of the American Society for Information Science, 26(1), 33-44.
[5] Matsuo, Y., and Ishizuka, M. (2008) Keyword Extraction from a Single Document Using Word Co-occurrence Statistical Information[J]. International Journal on Artificial Intelligence Tools, 13(1), 157-169.
[6] Barker, K., and Cornacchia, N. (2000) Using noun phrase heads to extract document keyphrases. Springer Berlin Heidelberg.
[7] Mihalcea, R., and Tarau, P. (2004) Textrank: bringing order into texts. Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, 404-411.
[8] Bougouin, A., Boudin, F., and Daille, B. (2013) TopicRank: Graph-Based Topic Ranking for Keyphrase Extraction. International Joint Conference on Natural Language Processing, 543-551.
[9] Radev, D.R. (2004) Lexrank: graph-based lexical centrality as salience in text summarization. Journal of Qiqihar Junior Teachers College, 22, 457-479.
[10] Yasunaga, M., Zhang, R., Meelu, K., Pareek, A., Srinivasan, K., and Radev, D. (2017) Graph-based Neural Multi-Document Summarization. Proceedings of the 21st Conference on Computational Natural Language Learning, 452-462.
[11] Xu, J., Liu, J., Cheng, Y., and Gan, Z. (2020) Discourse-Aware Neural Extractive Text Summarization. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 5021-5031.
[12] Liu, Y., and Lapata, M. (2019) Text Summarization with Pretrained Encoders. Association for Computational Linguistics, 3728-3738.
[13] Dai, Z., and Callan, J. (2020) Context-Aware Term Weighting For First Stage Passage Retrieval. SIGIR '20: The 43rd International ACM SIGIR conference on research and development in Information Retrieval. ACM.
[14] Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., and Majumder, R., et al. (2016) Ms marco: a human generated machine reading comprehension dataset, In CoCo@ NIPs.