Weighted TextRank-Based Chinese Automatic Text Summarization Postprint

Authors: Huang Bo, Liu Chuancai

Date: 2018-12-13T00:00:00+00:00

Abstract

Existing Chinese automatic text summarization methods primarily exploit information inherent in the text itself, suffering from the limitation that they cannot fully utilize semantic relatedness between words and other such information. In light of this, we propose an improved Chinese text summarization method. This method incorporates external corpus information into the TextRank algorithm in the form of word vectors. Through the combination of TextRank and word2vec, each word in a sentence is mapped to a high-dimensional vector space to form sentence vectors. By fully considering factors such as inter-sentence similarity, keyword coverage, and similarity between sentences and the title, we calculate the influence weights between sentences and select the top-ranked sentences, which are then reordered to form the text summary. The method achieves favorable results on the dataset used in this paper. Experimental results demonstrate that this method outperforms the original method in automatic Chinese summary extraction.

Full Text

Preamble

Vol. 37 No. 2 *Application Research of Computers* ChinaXiv Partner Journal Accepted Paper

Chinese Automatic Text Summarization Based on Weighted TextRank

Huang Bo, Liu Chuancai (School of Computer Science & Engineering, Nanjing University of Science & Technology, Nanjing 210094, China)

Abstract: Existing Chinese automatic text summarization methods primarily utilize the text' s own information, with the defect that they cannot fully exploit semantic relationships between words. To address this limitation, this

paper proposes an improved Chinese text summarization method. This approach incorporates external corpus information into the TextRank algorithm in the form of word vectors. By combining TextRank with Word2vec, each word in a sentence is mapped to a high-dimensional lexicon to form sentence vectors. The method fully considers factors such as similarity between sentences, keyword coverage, and similarity between sentences and titles to calculate influence weights between sentences, selecting the top-ranked sentences as the text summary. This method achieved good results on our dataset. Experimental results demonstrate that this approach is more effective than the original method for automatically extracting Chinese summaries.

Keywords: text summarization; TextRank; word vector; sentence similarity

Chinese Automatic Text Summarization Based on Weighted TextRank

Abstract: The method of Chinese existing automatic text summarization mainly utilized the text' s own information, and its defect was that it cannot make full use of the related semantic information between the words. Therefore, this paper proposed an improved Chinese text summarization method. This method integrated the information of the external corpora into the TextRank algorithm in the form of a word vector. Combined TextRank with Word2vec, it mapped each word in the sentence to the high-dimensional lexicon to form a sentence vector. This method fully considered the similarity between sentences, the coverage of keywords and the similarity between sentence and title to calculate the influence weights among sentences, and choose the top-ranked sentences used as the summarization of the text. This method has achieved good results in the data set of this paper. The results of experiment show that this method is more effective than the original method in extracting Chinese summarization automatically.

Key words: text summarization; TextRank; word vector; sentence similarity

1 Related Work

With the development of computer information technology, various types of information data on the Internet are growing exponentially. How to quickly obtain needed information from massive text has become critically important. Traditional manual information extraction methods can no longer meet demand, while automatic text summarization has attracted increasing attention and holds significant application value.

Text summarization extracts main content by summarizing and generalizing text information. Based on different summarization approaches, automatic summarization techniques can be divided into extractive and abstractive summarization [**?**]. In 1958, Luhn from IBM proposed a text summarization method based on scoring high-frequency terms [**?**], pioneering automatic text summarization research. Ge et al. constructed sentences into an undirected graph, converting

text summarization extraction into node weight calculation in a graph model [?]. Erkan et al. [?] proposed a text summarization algorithm based on the LexRank algorithm, which primarily calculates sentence weights based on term weights or sentence features, represents them as a graph model using vector space models, and extracts sentences with high similarity as text summaries. Li et al. [?] used keyword expansion methods to automatically extract summaries from news texts. This paper focuses on single-document Chinese text, specifically extracting sentences based on sentence weight scoring from individual documents to generate summaries.

Text summarization can be achieved using the text's own content and structural features, meeting requirements to a certain extent, with the TextRank algorithm [?] as a typical representative. Additionally, summaries can be extracted through training on large corpora. Unlike traditional simple algorithms, these methods require substantial training data. Traditional algorithms mostly treat documents as simple collections of words, ignoring semantic and grammatical elements, with each word appearing independently without dependencies. Incorporating external knowledge such as corpora into automatic text summarization algorithms could theoretically improve performance. The Word2vec model [?] developed by Google' s research team uses word vectors [?] to represent words and their relationships. This paper integrates Word2vec with the TextRank algorithm with improvements, using high-dimensional lexicon mapping based on word vectors to calculate sentence similarity instead of using co-occurrence frequency of identical words as influence weights between sentences, though it weakens the weighting effect of co-occurring words.

Luhn' s work [?] noted that frequently occurring words have strong associations with article topics, and calculating sentence weights based on term frequency for summary generation achieved higher accuracy than many complex methods [?]. Li et al. [?] achieved better results than the original method using keyword expansion based on TextRank for text summarization. Keywords play a significant role in extracting summary sentences from articles, and increasing keyword coverage—the proportion of keywords among all words after sentence segmentation—serves as a weighted factor for sentences, supplementing the weighting effect of terms. Additionally, article titles often represent main content to some extent. Cheng et al. [?] fully considered features such as term frequency, title, sentence position, and sentence similarity to construct a feature-weighted function for extracting key sentences to generate summaries. The greater the similarity between sentences and the text title, the more likely they are key sentences. Therefore, this paper utilizes sentence similarity, keyword coverage, and similarity between sentences and titles as influence factors for weights between sentences to extract text summaries.

1.1 TextRank Algorithm

The classic TextRank algorithm introduces Google's PageRank [?, ?] algorithmic 思想 into text summarization. The automatic text summarization algorithm

based on TextRank splits text information into sentences as network nodes, forming a graph model of sentence networks to represent structural relationships between sentences, and achieves importance ranking through iterative graph computation. This method requires no prior training on corpora or other related documents, is simple to implement, and performs well, thus gaining widespread application.

The general model of the TextRank algorithm can be represented as a weighted graph model $G = (V, E)$, where V is the set of nodes (i.e., sentences) and E is the set of edges. w_{ji} represents the weight of the edge between any two nodes V_i and V_j, i.e., the similarity between sentence V_i and sentence V_j. For any given node V_i, $In(V_i)$ is the set of nodes pointing to it, and $Out(V_i)$ is the set of nodes that V_i points to [**?**]. The score calculation formula for node V_i is:

Equation (1) is the recursive formula of TextRank [**?**], where d is the damping factor, with a value range between 0 and 1, representing the probability of a node in the graph model pointing to other nodes. A damping factor that is too large will cause a sharp increase in required iterations and make the algorithm' s ranking unstable, while one that is too small will result in no obvious effect from the iteration process. Generally, the value is set to 0.85 [**?**].

The edge weight w_{ji} is represented by sentence similarity, calculated based on the coverage rate of co-occurring words between sentences—by comparing the number of common words between different sentences. For given two sentences S_i and S_j, the following formula is used:

where $S_i = [w_{i,1}, w_{i,2}, \ldots, w_{i,j}, \ldots, w_{i,n}]$ is the set of words after removing stop words from the sentence, and $w_{i,j}$ is the j-th word in the i-th sentence after stop word removal. The PageRank algorithm obtains webpage importance by calculating mutual citation counts between webpages; in the TextRank algorithm, sentence similarity replaces the number of mutual links between webpages [**?**]. For example, for "我/爱/中国" and "你/喜欢/中国/吗", the edge weight is:

This method achieves some effect in calculating sentence similarity but ignores influencing factors such as word grammar and semantics, including relationships between synonyms and antonyms.

When using the TextRank algorithm to calculate scores for nodes in the graph model, arbitrary initial values are first assigned to each node, then recursive iteration is performed based on edge weights until the error rate of any node in the graph model is smaller than a preset threshold. Research shows that when the threshold is generally set to 0.0001 [**?**], recursive computation converges well.

1.2 Word2vec Model

Word2vec [**?**] is a tool developed by Google in 2013 for word vector calculation. It can efficiently train datasets of millions of levels or more, mainly using two models: CBOW (continuous bags-of-words model) and skip-gram (continuous skip-gram model), both based on Huffman trees. The CBOW model predicts

the probability of the current word based on context, while the Skip-gram model predicts context based on the current word. Both models contain three layers: input, projection, and output [?], as shown in Figures 1 [Figure 1: see original paper] and 2 [Figure 2: see original paper].

Both methods use artificial neural networks to train large batches of text, converting words in the text into word vectors in an N-dimensional vector space, using the similarity of spatial text vectors to measure text similarity. When neural network training is completed, word vectors can be obtained for words in the corpus that appear more than a preset threshold.

2 Research Methodology

The idea of automatic text summarization based on TextRank is to convert the extraction process into a ranking process of sentence importance in the text. First, word vectors are obtained by training the corpus using the Word2vec model to get sentence vectors. Then, similarity between sentences is calculated based on sentence vectors to construct a graph model of candidate sentence networks, i.e., a complete probability transition matrix between sentences. Through iterative computation, node importance is obtained to achieve automatic summarization ranking and extraction.

2.1 Chinese Text Preprocessing and Feature Selection

Using natural language processing (NLP) [?] techniques, text information is processed by first splitting the text body into individual sentences, then using the Chinese Academy of Sciences' NLPIR Chinese word segmentation system (also known as ICTCLAS2013) [?] to segment sentences, filtering out meaningless stop words such as "的, 地, 得, 了" to obtain word sets for each sentence.

2.2 Chinese Text Representation Based on Word Vector Model

Word2vec essentially uses a shallow neural network model (generally three layers) to learn and train the probability of words appearing in a corpus or dataset, representing words in a suitable dimensional space as numerical values—word vectors. The similarity between words can be measured using similarity between word vectors. Compared with traditional sparse matrix-based word representation methods that often encounter the curse of dimensionality and cannot represent semantic, grammatical information, and intrinsic connections between words, Word2vec-generated word vectors not only solve the dimensionality disaster problem—where word vector dimensions would increase unlimitedly with text growth—but also mine associative attributes between words from large-scale corpora, thus improving semantic accuracy of word vectors. For example, in traditional models, the words "好像" and "似乎" have no connection, but in Word2vec word vectors, they have high similarity.

A dimension of 100 is commonly used as the standard for training Word2vec

word vectors. If the dimension is too large, model training complexity increases dramatically. Moreover, the numerical value in each dimension of a word vector only represents the degree of positive or negative correlation of the word in that dimension, and its magnitude does not represent the actual correlation degree with words in the training vocabulary. Therefore, with the default dimension at the hundred level, neither direct accumulation and averaging nor taking the maximum value of each dimension can well represent sentences. This paper designs a sentence vector calculation method using trained word vector models for sentence similarity computation. In daily Chinese usage, a few thousand to tens of thousands of words can express information from most texts. This paper adopts a lexicon mapping method, constructing a high-dimensional lexicon to map text semantic information to a commonly used high-frequency word library, using lexicon mapping to obtain sentence vectors from sentence semantics.

First, a high-frequency common word lexicon with N words is constructed. Combined with word vectors for each word in the lexicon, each text can be mapped into an N-dimensional vector, where each dimension is the maximum similarity between the word corresponding to that dimension in the high-frequency lexicon and each word in the text. The text can be a sentence, article, etc. In terms of word similarity calculation, compared with context-based methods, each word obtains mapping in the high-dimensional word list and is expressed as dense features of the Word2vec model, such as [0.231262, 0.178923, 0.798699, 0.325891, ···], with almost no sparse cases where dimensions are 0. This can endow each word with richer semantic distribution, essentially extending bag-of-words rather than using simple 0 or non-0 values like [0,0,1,0,0,0,0,0,···] to represent absolute relevance or irrelevance, which effectively addresses the sparsity problem of bag-of-words.

Using the Word2vec tool to represent text vectors with high-dimensional lexicon R, assuming there are n words in the high-dimensional lexicon, the word vector representation is:

$R = [r_1, r_2, \dots, r_i, \dots, r_n]$, where r_i represents the word vector of the i-th word in the high-dimensional lexicon.

The specific algorithm is as follows:

Algorithm 1: Calculating Sentence Vectors for Chinese Text

Input: Sentence text, high-dimensional lexicon word vector set R, word vector model model
Output: Sentence vector

1. Segment the sentence text, remove symbols and stop words to obtain word set D
2. Initialize sentence word vector set T
3. For each d in D:
    - If d is in model: // If word d is contained in model
        - t = model[d] // Get the word vector corresponding to word d

    - t = [0]
  - End if
  - Add t to word vector set T
4. End for
5. Initialize sentence vector set S
6. For each r in R:
  - Initialize word similarity set C
  - For each t in T:
    - Add cos(r,t) to C
  - End for
  - Add the maximum value in set C to set S
7. End for
8. Return S

where step 15 uses equation (4) to calculate similarity between word vectors, and step 19 requires converting the set type to a numerical type.

2.3 TextRank Weight Calculation

The original TextRank automatic text summarization algorithm typically uses equation (2) to calculate similarity between sentences, but simply computes the coverage rate of identical words between sentences as edge weights. To further improve text summarization effectiveness, this paper introduces external knowledge into automatic summarization using the Word2vec tool, making the following improvements to edge weight calculation in the TextRank algorithm. Using relationships between document sentence vectors, a TextRank model is built between sentences, weighting the probabilities between nodes in the TextRank graph model through sentence similarity, keyword coverage, and similarity between sentences and titles to calculate the influence weight of each sentence, ranking them by weight from largest to smallest.

2.3.1 Sentence Similarity After using Chinese word segmentation to remove punctuation and stop words, sentences are represented as word sets. According to equation (3), sentences are mapped to a high-dimensional lexicon and represented in vector form. When calculating similarity between two sentences S_i and S_j mapped using the high-dimensional lexicon, cosine distance is also used (S_i and S_j are sentence vectors mapped to the high-dimensional lexicon):

Let the text have m words after segmentation and removal of special symbols and stop words.

2.3.2 Keyword Coverage The more keywords a sentence contains, the higher its importance. $W_k(S_i, S_j)$ represents the weight transferred from sentence node S_i to sentence node S_j based on keyword coverage, calculated as:

where $p(S_i) = \text{len}(\text{keywords}(S_i))/\text{len}(S_i)$ represents the ratio of the number of

keywords in sentence S_i to the total number of words in the sentence (after removing punctuation and stop words), i.e., the keyword coverage rate in sentence S_i.

2.3.3 Sentence-Title Similarity The higher the similarity between a sentence and the text title, the higher the sentence' s importance. $W_t(S_i, S_j)$ represents the weight transferred from sentence node S_i to sentence node S_j based on title similarity, calculated as:

S_t is the word vector representing the text title mapped to the high-dimensional lexicon, S_i and S_j are sentence word vectors mapped to the high-dimensional lexicon, and Similarity(S_j, S_t) is the similarity between sentence S_j and title S_t. Similarity is generally calculated using cosine distance, i.e., equation (5).

Based on the above formulas, a new weight for influence between sentences is constructed. After normalizing the three weight influence factors respectively, the sentence influence weight formula is constructed as:

Experimental results show that when $a = 0.6$, $b = 0.2$, $c = 0.2$, automatic summarization achieves the best effect. When the value of a gradually approaches 0.6, the evaluation effect generally shows an upward trend, while when $a < 0.5$, the evaluation effect gradually declines. This demonstrates that sentence similarity plays an important role in TextRank weight calculation, while keyword coverage and sentence-title similarity are relatively less important.

Table 1 Weighting coefficient values under different ratio combinations

where a, b, and c represent the respective proportions of sentence similarity, keyword coverage, and sentence-title similarity in weight calculation, i.e., for the constructed graph model, three influence factors for weights between nodes (sentences) are proposed. a, b, and c are the weighted coefficients after normalization of these three sentence weight influence factors, with larger weighted coefficients indicating greater influence of the corresponding factor in weight calculation. Their values range between 0 and 1, and $a + b + c = 1$.

3 Experiments

3.1 Experimental Data and Evaluation Metrics

This paper uses part of the Wikipedia Chinese data released in October 2017 and the Chinese text news dataset launched by Tsinghua University' s Natural Language Processing Laboratory. After data cleaning to filter out punctuation and other irrelevant symbols, the Chinese word segmentation tool NLPIR (also known as ICTCLAS) developed by Dr. Zhang Huaping is used for segmentation, forming a text dataset for training and learning. The Word2vec module in the Python-based natural language processing library Gensim is then used with the CBOW model, dimension 100, window size 5, and other default parameters to train the text dataset and obtain the word vector model file [**?**].

Currently, there is no widely recognized evaluation corpus or standard for Chinese automatic text summarization. The test text dataset in this paper comes from the Chinese text news dataset launched by Tsinghua University' s Natural Language Processing Laboratory, taking 20 articles from each of five categories (sports, finance, technology, politics, and entertainment) for a total of 100 articles as the test corpus. Three graduate students majoring in linguistics manually extracted summaries from the text information in the test corpus. Each independently extracted 8 to 10 summary sentences from each document and ranked them by relevance to the article content. Finally, the summary results from the three annotators were combined, and the three sentences with the highest relevance to the article content that appeared in all three results were taken as the manual summary sentences.

The evaluation method for experimental summary quality adopts the Rouge metric [**?**], the most widely used evaluation metric in the automatic summarization field. Rouge evaluates summaries based on co-occurrence information of n-grams, representing an automated evaluation method oriented toward n-gram recall. The basic idea is to compare system-generated automatic summaries with human-generated standard summaries, evaluating summary quality by counting the number of overlapping basic units (n-grams, word sequences, and word pairs). This paper uses three evaluation metrics: Rouge-1, Rouge-2, and Rouge-L.

3.2 Determination of Weight Coefficients for Influence Factors

To reasonably evaluate automatic summarization quality, this paper adopts the aforementioned Rouge-1, Rouge-2, and Rouge-L metrics as evaluation standards, calculating weighted coefficients for the three influence factors based on sentence similarity, keyword coverage, and sentence-title similarity for each sentence in each text. Combining the weighted coefficients a, b, and c for sentence weight influence factors (where $a+b+c=1$), this paper varies a, b, and c at intervals of 0.05 (increasing or decreasing while ensuring $a+b+c=1$). Through extensive experiments, Rouge-1, Rouge-2, and Rouge-L values under different weighted coefficient combinations were calculated, with some representative experimental data shown in Table 1.

Figure 3 [**Figure 3: see original paper**] Experimental results of different parameter combinations

3.3 Experimental Results and Analysis

Through experiments, this paper obtained an optimal set of weighted coefficients. To verify the effectiveness of the proposed method, experiments were conducted on the 100-article test text summarization dataset using TF-IDF [**?**], LexRank, TextRank, and the improved method proposed in this paper for comparison.

Experimental results are shown in **Figure 4** [**Figure 4: see original paper**].

Figure 4 [**Figure 4: see original paper**] Comparison of the effects of different text summarization methods

For the 10 parameter combinations selected above, automatic summarization results were calculated for each test text and their means taken. Experimental results are shown in **Figure 3** [**Figure 3: see original paper**].

The above experimental data show that the improved method proposed in this paper achieves significant improvements across all three evaluation metrics: Rouge-1, Rouge-2, and Rouge-L. The TF-IDF based method performs worst comparatively. The improved method in this paper outperforms both the LexRank algorithm and traditional TextRank text summarization algorithms. In addition to considering the text' s own features, it also introduces external knowledge bases, increasing influence factors for sentence weights.

4 Conclusion

This paper proposes a Chinese automatic text summarization algorithm based on weighted word vectors, combining graph model-based sentence ranking algorithms with Word2vec model word vectors, fully considering factors such as sentence similarity, keyword coverage, and sentence-title similarity. Experimental results demonstrate that for single-document Chinese automatic summarization, compared with traditional TextRank algorithms, the proposed method achieves better summary extraction results and effectively improves automatic summarization quality. However, text summarization speed needs improvement, which is the target for future work.

References

[1] Gambhir M, Gupta V. Recent automatic text summarization techniques: a survey [J]. Artificial Intelligence Review, 2016, 47(1): 1-66.

[2] Luhn H P. The automatic creation of literature abstracts [J]. IBM Journal of Research & Development, 1958, 2 (2): 159-165.

[3] Ge Bin, Li Fangfang, Li Fu, et al. Subject Sentence Extraction Based on Undirected Graph Construction [J]. Computer Science, 2011, 38(5): 181-185.

[4] Erkan, G, Radev D R. LexRank: graph-based lexical centrality as salience in text summarization [J]. Journal of artificial intelligence research, 2004, 22 (1): 457-479.

[5] Li Feng, Huang Jinzhu, Li Zhoujun, et al. Automatic summarization method of news texts using keywords expansion [J]. Journal of Frontiers of Computer Science and Technology, 2016, 10 (3): 372-380.

[6] Yu ShanShan, Su Jindian, Li Pengfei, et al. Towards high performance text mining: a textrank-based method for automatic text summarization [J]. International Journal of Grid & High Performance Computing, 2016, 8 (2): 58-75.

[7] Goldberg Y, Levy O. word2vec Explained: deriving Mikolov et al.' s negative-sampling word-embedding method [EB/OL]. (2014) [2018-09-13]. https://arxiv.org/abs/1402.3722v1.

[8] Yang Zhitong, Zheng Jun. Research on Chinese text classification based on Word2vec [C]//Proc of the 2nd IEEE International Conference on Computer and Communications. 2017: 1166-1170.

[9] Kumar Y J, Goh O S, Halizah Basiron, et al. A review on automatic text summarization approaches [J]. Journal of Computer Science, 2016, 12 (4): 178-190.

[10] Chen Yuan, Wushouer Silamu, Maimaitiyiming Hasimua, et al. Automatic test summarization based on comprehensive characteristics of sentence [J]. Computer Science, 2015, 42(4): 226-229.

[11] Haveliwala T H. Topic-sensitive PageRank: a context-sensitive ranking algorithm for Web search [J]. IEEE Trans on Knowledge & Data Engineering, 2003, 15 (4): 784-796.

[12] Chen Ningyuan, Nelly Litvak, Mariana Olvera-Cravioto. Generalized PageRank on directed configuration networks [J]. Random Structures & Algorithms, 2017, 51 (2): 237-274.

[13] Xia Tian. Study on Keyword Extraction Using Word Position Weighted TextRank [J]. New Technology of Library and Information Service, 2013, 29(9): 30-34.

[14] Mihalcea R, Tarau P. TextRank: bringing order into texts [C]// Proc of Conference on Empirical Methods in Natural Language Processing. 2004: 404-411.

[15] Brin S, Page L. The anatomy of a large-scale hypertextual web search engine [J]. Computer networks and ISDN systems, 1998, 30 (1-7): 107-117.

[16] Lilleberg J, Zhu Yun, Zhang Yanqing. Support vector machines and Word2vec for text classification with semantic features [C]//Proc of the IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing. Beijing: IEEE, 2015: 136-140.

[17] Sun Shiliang, Luo Chen, Chen Junyu. A review of natural language processing techniques for opinion mining systems [J]. Information Fusion, 2017, 36: 10-25.

[18] Chen Xinchi, Qiu Xipeng, Zhu Chenxi, et al. Long Short-term memory neural networks for chinese word segmentation [C]//Proc of Conference on Empirical Methods in Natural Language Processing. 2015.

[19] Lin C Y, Eduard H. Automatic evaluation of summaries using N-gram co-occurrence statistics [C]//Proc of Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology. Stroudsburg: Association for Computational Linguistics, 2003: 71-78.

[20] Paik Jiaul H. A novel TF-IDF weighting scheme for effective ranking [C]//Proc of International ACM SIGIR Conference on Research and Development in Information Retrieval. New York: ACM Press, 2013: 343-352.

Note: Figure translations are in progress. See original paper for figures.