Text Similarity Algorithm Based on Semantic Vector Space Model

LiHong Xu[1], ShuTao Sun[2], Qi Wang[3]
School of computing
CUC, No.1, Dingfuzhuang Street(E), Chaoyang District
Beijing, China
e-mail: xfdyg1102@163.com

***Abstract*—In this paper, a text similarity computation method named VSM-Cilin which is based on semantic vector space model is proposed in the background of radio station. VSM-Cilin improved the traditional VSM in the following areas. First, consider the semantic relations between words. Second, use semantic resources to reduce dimension. Third, use inverted index to filter out candidate document set. Forth, take the weight of the feature item into consideration when compute the similarity. The experiments show that the accuracy of VSM-Cilin is significantly improved compared with the traditional vector space model and the method of bidirectional mapping based on HITIR-Lab Tongyici Cilin.**

Keywords—VSM; semantic; Tongyici Cilin; text similarity

I. INTRODUCTION

There are vast amount of audio resources in radio station's library. It's a very arduous task for editors to choose the audio they wanted from the library. How to get the audio they need quickly become a more and more important problem to be solved. In traditional retrieval system, a lot of radios that without complete labeling information can't be activated, and lost its value. In this paper, we proposed a solution to this problem. First, the system get the audio corresponding text which we named audio-text through the mature speech recognition technology, then according to the draft edited by editors, the system find out five most similar audio-texts with text similarity matching algorithm, and recommend them to the editors. So, editors need only to focus on these five audios.

In terms of text similarity algorithm, it has been studied for a very long time. VSM is an information similarity computation model which is first proposed by Salton, and it has been used more frequently and more effectively in recent years[1]. The traditional VSM only consider statistical properties of words in context without considering the semantics of words.

Latent semantic analysis is another classic algorithm. This algorithm need to choose some representative words from a large number of text firstly, and to form a word-document matrix. Then, decrease the dimension of the matrix by singular value decomposition(SVD), and find out the potential link between word and word. While this algorithm is dependent on the context information, it can not reflect the text latent semantic when the document data is too sparse, and its computation is complexity[2].

Other algorithms including algorithm based on probability statistical model, algorithm based on ontology, maximal marginal relevance algorithm, algorithm based on edit distance, algorithm based on string matching and penalty mechanism, algorithm based on the surface structure of sentence and common content[3].

This paper used the method based on improved vector space model with HITIR-Lab Tongyici Cilin(Extended) which we named VSM-Cilin to compute text similarity. VSM-Cilin take the correlation between words into consideration, reduce the feature dimension by synonym clustering, filter out candidate documents by inverted index, retain weights in the semantic space. Good results have been obtained through these method.

II. RELATED DESCRIPTION AND DEFINITION

A. HITIR-Lab Tongyici Cilin

Considering functional needs of the algorithm and the complexity of codes of semantic resources, we choose HITIR-Lab Tongyici Cilin to be the semantic knowledge resources of the VSM-Cilin algorithm. Cilin classify word into five levels, and provide a category five coding[1]. The higher the level, the more detailed the meaning of the word. So, the bigger the level of the code of the two words are the same in, the more similar of the two words, when they are the same till the fifth level, they have the biggest similarity remembered to 1, and we consider these two words to be interchangeable.

B. VSM (Vector Space Model)

The fundamental idea of this algorithm is supposing that the words are independent, and each document is expressed in a space vector, each word is a dimension of the space vector, that simplify the complexity relationship of the words and convert the computing of the document similarity into the computing of the angle between vectors[4]. In this model document can be expressed as $(t_1,t_2,...,t_n)$, $(w_1,w_2,...,w_n)$ is the corresponding coordinate value of the n-dimension space which is composed of $(t_1,t_2,...,t_n)$, then the text similarity computation is converted to relationship computation of high-dimensional vectors. Using this model to compute Chinese documents' similarity we have three necessary works to do, including common semantic space construction(Chinese segmentation, term weighting, feature selection), mapping documents to semantic space, computing similarity.

C. Inverted index

Efficiency is an important problem to be considered in the algorithm. A very practical way to improve efficiency of document retrieval is to use the index. Index is a data structure based on text, which is used to speed up the search speed [4]. In this paper, we use inverted index which is the most commonly used index technology to speed up the retrieval. It is a word oriented scheme for indexing documents and speeding up the search tasks. A simple inverted index structure consist of a basic inverted index of document set and an item-document matrix. This storage structure requires too much space, for it's a sparse matrix, each document contains only a small part of the vocabulary words. The solution is to link a list of documents with each word, and replace the item-document matrix with an inverted table. Each record of the inverted index table include a word and a document list.

III. VSM-CILIN

A. Introduction of VSM-Cilin

The text similarity computation algorithm based on VSM consider that the words are independent. While in real condition, there are a lot of polysemants and synonyms[6]. So when using the traditional VSM to compute the document similarity, the document that has the same topic with the target document but expressed in other polysemants and synonyms words can't be found. To solve this problem, we add Cilin into the model as semantic dictionary. Another problem in the normal algorithm of similarity computing of texts is, once the features are selected in the stage of feature selection, they get the same weight[7]. In this paper, we take the weight of the feature into account in the stage of mapping documents to semantic space. On one hand, higher weight of the feature means more contribution to the semantic space, these information can't be ignored, on the other hand , considering the weight of the feature means considering all the documents, the effect will be better if there are more documents in the sets. Also, we improved this model in other aspects, which will be introduced in the following. Fig. 1 shows the processes of the similarity computation module.

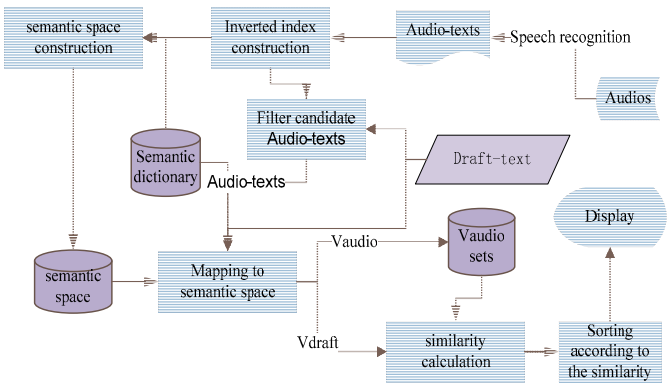

Fig. 1. Processes of the similarity computation module

B. Text similarity computation based on VSM-Cilin

1) Text pre-processing

Chinese word segmentation. Chinese is different from English for that English have a space to delimit different words, but Chinese have not. To count, analyses Chinese word, we should do Chinese segmentation firstly. In this paper, word segmentation is completed by NLPIR segmentation system. Its principle is from the long text content, discover new language features automatically based on cross entropy of information, and adapt to the test corpus linguistic probability distribution model to realize adaptive segmentation[8]. In this step, we get a word set V, it contains all the words that appeared in the document data sets.

Remove stop words. Stop words refer to the words that have high frequencies of occurrences in text, but have no actual meaning. This category words include the particles, adverbs, preposition, conjunction, etc., usually there is no clear meaning itself, only put it in a complete sentence, then it will have a certain role. Extensive using of stop words in documents will cause noise interference to effective information easily[9]. So, it's important to eliminate the noises before feature weighting and feature selection. Reducing the frequency of the occurrence of stop words appropriately can help us to improve the density of keywords effectively and to reduce the number of the words and amount of computing. In this step, we remove all stop words from V.

2) Build inverted index table

For every word in V, record the document that the word appears in and record its appears times. The structure of the inverted index table is shown in Table. 1.

TABLE I. STRUCTURE OF INVERTED INDEX TABLE

Word flag	index term	document-word frequency set
Flag1	TS_1	[Doc,Freq][Doc，Freq]...
Flag2	TS_2	[Doc,Freq][Doc，Freq]..
...	...	...

"Doc" is represent the document that contains the word, "Freq" is the times that the word appears in Doc.

3) Filter candidate document set

For a query draft-text, which have a limited number of words, the number may even be very small, and its dimension values which is relative to the vocabulary table is sparse. According to the characteristics, we use the inverted index to filter out a large part of irrelevant documents.

- For a audio-text, we get its words and calculate the word weight by its appear position of the audio-text, and store the word and its weight in a array Draft[k].

$$Draft[k] = \{[t_1, tw_1],[t_2, tw_2],...[t_k, tw_k]\} \quad (1)$$

- For every t_i in the draft-text, we check all "index term" of the inverted index table, while t_i is contained in the index terms, we get its "document-word frequency set".
- The document weight "DocWeight" which is represent the relevance of the document and the draft-text is compute by (2). The formula means that if t_i is appears in this document, increase its "DocWeight" of dw_i which is compute by (3). "DocLen" is the length of the

document which is computed by the number of the words. The reason that "freq" to be divided by "DocLen" is that longer documents contain more index information which may result in a larger correlation. To salve this problem, document length normalization is necessary.

$$DocWeight = \sum_{i=1}^{m} [\mathrm{dw_i}] \tag{2}$$

$$dw_i = w_i \times \mathrm{freq/DocLen} \tag{3}$$

- Ranked documents by "DocWeight", and take the top N/2 (N is the total document number) documents as candidate document set. The number of documents is reduced by half.

4) Common semantic space construction

The system constructs a common semantic space vector according to the audio-texts in the library. The semantic space contains a feature set named TS and a weight set WS which is corresponding to TS. To construct this common semantic space vector we need to follow these steps.

a) Feature weighting based on TF-IDF[3]

For N is the total number of documents in document set, df_i is the number of documents that contains the word V_i, f_i is the times the word t_i appears in V. So, the regularized word frequency is computed by (4).

$$tf_i = \frac{f_i}{\max\{f_1, f_2, ... f_{|V|}\}} \tag{4}$$

Formula (4) used the TF normalization method based on maximum, due to it's more likely that a word appear repeatedly in a long document. So, words in long document have a tendency to take a larger frequency. This will obviously result in the long text gain a bigger weight, and the main effect of the maximum TF normalization is to mitigate the impact of this unfairness.

The inverse document frequency is computed by (5) .

$$idf_i = \log \frac{N}{1 + df_i} \tag{5}$$

As can be seen from (5), if a word appears in a lot of documents of the data sets, then it may not be very important, because it can not distinguish between documents.

The value of tf-idf of t_i is computed by (6).

$$TI_i = tf_i \times idf_i \tag{6}$$

TI_i which is calculated by (6) is quite different from each other, to eliminate the impact of this numerical size, it's necessary to standardize the data, and control the degree of similarity between the range of (0,1). Finally, the weight of t_i is computed by the (7).

$$W_i = \log_{10} TI_i \div \log_{10} \max\{TI_1, TI_2, ... TI_{|V|}\} \tag{7}$$

b) Using Cilin to reduce the dimension

If all words in V are selected as features, the dimension of the common vector will be too high, this will result in more computation time and memory requirement[7]. Hence we iterate through V to find the word which there is another word in the collection that has a similarity of 1 with it. Then we remove the word from V that have a smaller W_i .

c) Feature selection

After step c), |V| is still too high, we sorted V according to W, and set a variable θ(θ>0) to represent the number of features we selected. In the following, this paper will show how to determinate the variable θ. Finally, we get V(|V|=θ) as TS, and W as WS.

5) Mapping to common vector space

After the construction of a semantic space, every audio-texts in candidate document set and draft-texts should be mapped into the semantic space and be represented in a vector of the space.

For a audio-text, it original structure is as follows:

$$\mathrm{Audio[m]} = \{[\mathrm{t_1, tw_1}], [\mathrm{t_2, tw_2}], ... [\mathrm{t_m, tw_m}]\} \tag{8}$$

Initialize a vector named V_{audio}, it's one of the vectors in the semantic space.

$$V_{\mathrm{Audio}} = (Va_1, Va_2, ... Va_\theta)(\mathrm{Vaj} = \sigma, 0 < j < \theta) \tag{9}$$

σ is a constant, it often takes a relatively small value.

Traversing each word in Audio[m] to find the a word in TS_j which has a max similarity with it, then change the value of Va_j use (10) and (11).

$$TempWeight_i = Max\{Sim\mathrm{Word}(t_i, TS_j)\} \times tw_i \times WS_j + \mathrm{Va}_j (0 < j < \theta) \tag{10}$$

$\mathrm{sim}(t_i, TS_j)$ is compute by Cilin.

$$\mathrm{Vd_j} = TempWeight_i \tag{11}$$

We can map the draft-text into the semantic space in the same way, and get the V_{draft} by (12).

$$V_{draft} = (Vd_1, Vd_2, ... Vd_\theta)(\mathrm{Vd_j} = \sigma, 0 < \mathrm{j} < \theta) \tag{12}$$

6) Similarity measure

Similarity measure is to calculate the degree of similarity between texts, the smaller the value, the smaller the degree of similarity, the greater the difference is. We use the cosine of the angle between the two vectors to measure the difference between the texts[3], which is shown in (13).

$$\mathrm{SimText}(V_{Audio}, V_{Draft}) = \frac{< V_{Audio} \bullet V_{Draft} >}{\|V_{Audio}\| \times \|V_{Draft}\|} \tag{13}$$

IV. EXPERIMENT RESULT

The audio-texts are selected and built in the corpus, which consist of 10 types of news, including entertainment, music, politic, military, economic and so on. Each type has 50 audio-texts, a total 500. The experiment includes 80 draft-texts, including entertainment, politics, military and sports draft. Every draft has its corresponding audio-text, while the corresponding audio-text is ranked in the top five, we consider it to be accurate.

A. Experiments on variable choices

The recommendation accuracy is effected by the variable θ and σ. The smaller of θ, the less words will be considered in, which will result in less of computation, but the accuracy may decline . Fig. 2 shows the change trends of accuracy with θ when σ=0. From the figure, we can see that to get less computation time and memory requirement with high accuracy, set θ=40%×|V| is a better choice. That is to reduce the dimension of public space, and to ensure a more stable rate of recommendation accuracy.

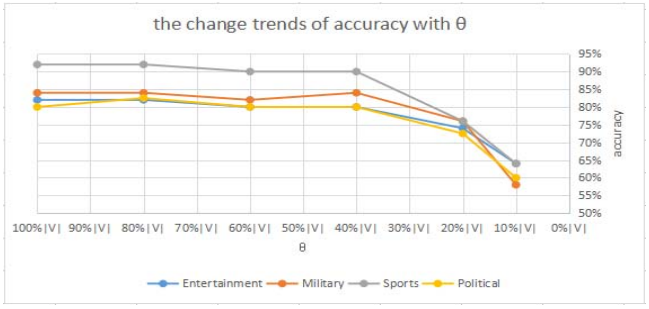

Fig. 2. Accuracy changed along with θ

Fig. 3 shows the change trends of accuracy with σ when θ=40%×|V|, from where we can see σ=0.0000001 is the better choice.

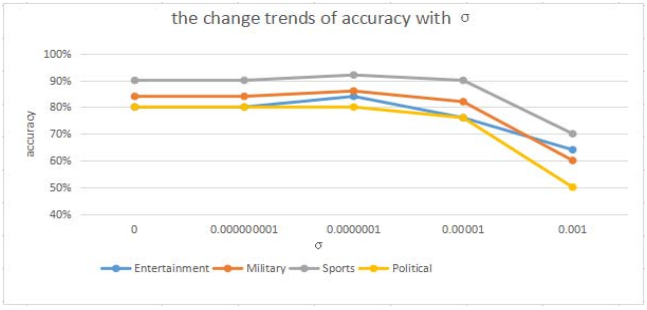

Fig. 3. Accuracy changed along with σ

B. Comparative Experiment

The results of the VSM-Cilin comparing with the traditional text similarity algorithm based on VSM and the text similarity algorithm based on bidirectional mapping with Cilin(BM-Cilin) are shown in Fig. 4. The results show that the accuracy of the system which uses VSM-Cilin algorithm is obviously higher than text similarity algorithms of others. This shows that VSM-Cilin plays an important role in recommending the most similar audio-text for the draft-text.

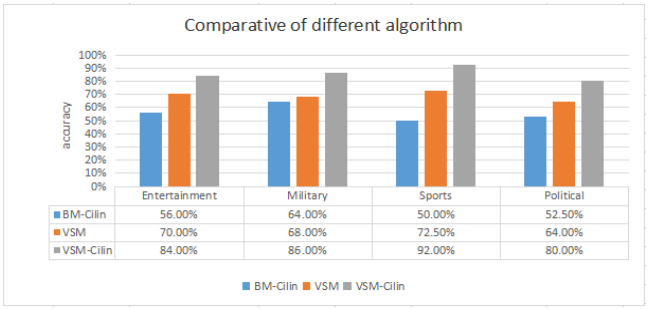

Fig. 4. Comparative of different algorithm

V. CONCLUSION

In this paper, the similarity computation module is proposed to recommend five most similar audios to the editors immediately when he completed a draft-text. Based on VSM-Cilin, the similarity computation module obtained a satisfied performance.

ACKNOWLEDGMENT

The research work was supported by the engineering planning project of Communication University of China under Grant No. 3132015XNG1523.

REFERENCES

[1] L. Xu, D. Wang, and M. Huang, "Improved Sentence Similarity Algorithm based on VSM and its application in Question Answering System." Intelligent Computing and Intelligent Systems (ICIS), 2010 IEEE International Conference on IEEE, 2010, pp. 368 - 371.

[2] Peter W. Foltz, Walter Kintsch, and Thomas K Landauer, "The measurement of textual coherence with latent semantic analysis." Discourse Processes 25.2-3(1998), pp. 285-307.

[3] W. Song, "Applications of short text similarity assessment in user-interactive question answering." , 2010.

[4] B. Liu, Web Data Mining. Springer Berlin Heidelberg, 2011.

[5] Knight, Jon P., and Martin Hamilton, "A File System Based Inverted Index.", 1995.

[6] Y. Ma, J. Liu, and Z. Yu, "Concept name similarity calculation based on WordNet and ontology." Journal of Software 8.3 (2013), pp. 746-753.

[7] Q. Guo, "The similarity computing of documents based on VSM." Network-Based Information Systems. Springer Berlin Heidelberg, 2008, pp. 142-148.

[8] H. Gong, and C. Zhou, "Chinese word segmentation system research." Journal of Beijing Institute of Machinery 3 (2004): 011.

[9] T. Xia, Y. Chai, H. Lu, "E-learning support system aided by VSM based Question Answering System." Computer Science & Education (ICCSE), 2013 8th International Conference on. IEEE, 2013, pp. 1281-1285.