Received: 18 December 2017 | Revised: 1 March 2018 | Accepted: 2 March 2018
DOI: 10.1002/cpe.4489

WILEY

SPECIAL ISSUE PAPER

Improving semantic similarity retrieval with word embeddings

Fengqi Yan[1,2] | **Qiaoqing Fan**[1] | **Mingming Lu**[1]

[1]Department of Electronic and Information Engineering, Tongji University, Shanghai, China
[2]Shanghai Heyeah Information Technology Co, Ltd, Shanghai, China

Correspondence
Mingming Lu, Department of Electronic and Information Engineering, Tongji University, Shanghai 200092, China.
Email: 1310500@tongji.edu.cn

Funding information
Science and Technology Commission of Shanghai Municipality, Grant/Award Number: 16511102800

Summary

Word similarity matchmaking is one of the core research areas of information retrieval. The existing methods based on a synonym dictionary would lead to the problem of semantic gap, which could be caused by the absence of synonyms. To address this problem, we improve semantic similarity retrieval by incorporating word embeddings. Especially, word embeddings are trained by Word2Vec and then use them to depict the semantic similarity between words. Experiments are conducted on two different datasets, ie, one is a public long text dataset (ie, Reuters-21578), and the other is a short text dataset (ie, 120ask) collected from a healthcare community. The experimental results on the two datasets show that the proposed method further improves the accuracy of the similarity retrieval.

KEYWORDS

information retrieval, semantic similarity, word embedding

1 | INTRODUCTION

Web search engines, eg, Google and Baidu, have become an important window for people to understand the outside world. They index all of the global Web sites to help users find valuable information. Web search engines mainly use the word form-based retrieval technology.[1] A typical method is to represent documents as vectors by calculating the TF-IDF (Term Frequency-Inverse Document Frequency) value of each term of the documents. Then, the method of cosine similarity between vectors is used to compare user query with documents. Finally, we obtain a list of documents as the retrieval result, which is sorted from high to low by similarity degrees.

Although the word form-based retrieval technology can meet the needs of users to obtain information to a certain extent, users expect to have a higher retrieval accuracy, especially in the cases of exposing the semantic relation between information. Thus, semantic retrieval techniques[2] are considered to be an effective way to solve the problems. For example, when a user searches for "slim," the traditional word form-based retrieval technology cannot return the documents, which include the words of "thin." In contrast, the semantic retrieval technology can return these relevant documents because "slim" and "thin" are semantically similar.

In the past, the research of semantic retrieval was focused on ontology-based semantic reasoning mechanisms.[3-5] Due to the difficulties in building a strong and robust ontology knowledge base, it normally takes lots of human and material resources, so that the ontology-based semantic retrieval technologies have not been widely applied. Later, Huang et al[6] proposed a function focusing on the semantic information of terms. In their work, the synonyms dictionary is used to depict the semantic similarity between terms, and then the keywords of user query are extended to a set of similar terms. Through this method, the precision of the retrieval results is improved. Similar to the ontology knowledge base, the problem with the synonym dictionary is that the words and the relationships between words are limited. For words that do not exist in the dictionary, their semantic similarity cannot be calculated.

Currently, word embeddings have been widely integrated in different natural language processing (NLP) tasks, such as text classification[7] and statistical machine translation.[8] Word embedding, also known as distributed representation of words,[9] maps each word into a fixed-length vector (the dimension usually ranges from tens to hundreds, which is much smaller than the size of the dictionary) and captures both the syntactic and semantic information of words. Bengio et al[10] applied a three-layer feed-forward neural network to train the probabilistic language model, and

the word embeddings are output as the byproduct of the language model. The current mainstream word embedding training methods, CBOW and Skip-gram, were proposed by Mikolov et al.[11,12] An open source implementation is Word2Vec, which is employed in our work.

Compared with the TF-IDF model, word embedding represents the word as a low-dimensional real vector, which makes it possible to overcome the problem of feature sparseness and dimension disaster in the TF-IDF model. Furthermore, word embedding captures the semantic information of words through context, which provides a way to solve the problem of semantic gap caused by the absence of synonyms. Therefore, based on the work of Huang et al,[6] this paper presents an improved semantic similarity retrieval algorithm, which incorporates word embeddings. The algorithm employs Word2Vec[11,12] to train word embeddings and based on which to depict the semantic similarity between words. The experiments are conducted on two different datasets. One is the public Reuters-21578 dataset,[13] which is characterized by large texts and the dataset has a semantic gap between the words. Another is from the 120ask* healthcare question answering community, which consists of 10,000 questions. Because the length of text in the 120ask dataset is short, as well as there are many professional vocabulary and colloquial expressions in the sentences, the semantic gap problem is obvious when the synonyms dictionary is used to find semantic similar words. The experimental results on these two datasets show that the proposed semantic similarity retrieval method incorporating word embeddings further improve the retrieval precision.

The rest of this paper is organized as follows. Section 2 reviews the related work for semantic retrieval. Section 3 details the implementation of the proposed algorithm. Section 4 presents and discusses the experimental settings and results, respectively. Section 5 concludes this paper.

2 | RELATED WORK

The research efforts on semantic retrieval are mainly focused on ontology theory, semantic dictionary, and topic model.

The concept of ontology is derived from philosophy, and it studies the basic categories of objects and their relations. In the field of computer science, Sun et al[2] and Studer et al[14] defined four characteristics of ontology as conceptualization, explicit, formal, and shared. Since then, many ontology-based retrieval technologies have been proposed.[3-5] However, the establishment of ontology knowledge base requires the participation of many experts, so that the maintenance would be a serious problem, and the knowledge base could not be updated frequently.

The synonym dictionary maintains the relevant semantic information for each word, such as WordNet[15] and HowNet.[16] Lots of researchers have applied synonym dictionary in query expansion. Liu and Li[17] proposed a similarity calculation method incorporating HowNet, which filled the gap for measuring Chinese words similarity. However, the synonym dictionary is limited to the words included. For the words that do not exist in the dictionary, we cannot calculate their similarity degrees. For example, if the "slim" word is not included in the synonym dictionary, the semantic relationship between "slim" and "thin".

Topic model is a type of latent semantic analysis technologies that uses statistical methods to learn the distribution of the topics of each document and the distribution of the words of each topic from the corpus. Then, each word can be represented as a topic vector, while each entry is the degree of contribution of the words in corresponding topic. Therefore, Sun et al[18] proposed a similarity calculation method using the topic information as features. However, when the corpus grows dynamically, it become challenging to find the appropriate dimensions of the topics.

Recently, state-of-the-art results have achieved many NLP tasks by incorporating word embeddings, eg, text classification,[19,20] text summarization,[21] and information retrieval.[22] Among these works, Kusner et al[23] used word embeddings and term frequency information to represent texts and then calculated text similarity by comparing word mover's distances (WMD) between texts. Although this method achieved best results in text classification task, its computational complexity was very high. Compared to their work, our improved similarity retrieval algorithm has smaller computational complexity.

3 | ALGORITHM IMPLEMENTATION

In this section, we first review two training models of Word2Vec and then describe the similarity function based on the semantic information of terms. The details of our improved algorithm are described in detail.

3.1 | Word2Vec

Word2Vec is an open sourced word embedding training toolkit based on the works of Mikolov et al.[11,12] Two training models of Word2Vec, CBOW and Skip-gram, are shown in Figure 1. In the CBOW model, current word can be represented by its context; on the contrary, the Skip-gram model uses the context to represent current word.

Figure 2 is the detailed structure of the CBOW model. In Figure 2, context(w) represents the context of word w, which contains k words. $\boldsymbol{V}_w$ is the word embedding of w. The input layer maps the context to word embeddings, respectively. Then, all of them are summed in a projection layer as $\boldsymbol{X}_w$. The output layer of CBOW model is a Huffman tree, which predicts the word w. Here, θ_j^w is the coding value of Huffman tree in layer j to word

*http://www.120ask.com/

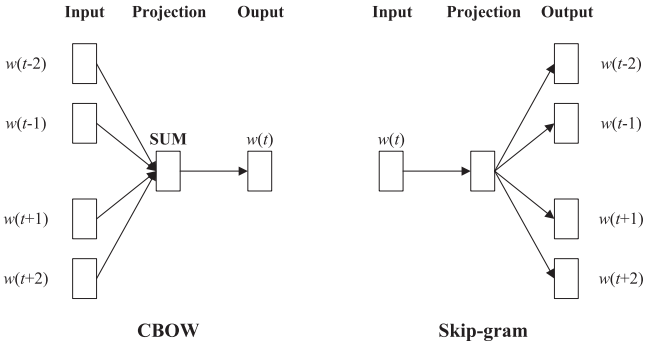

FIGURE 1 Two training models of Word2Vec

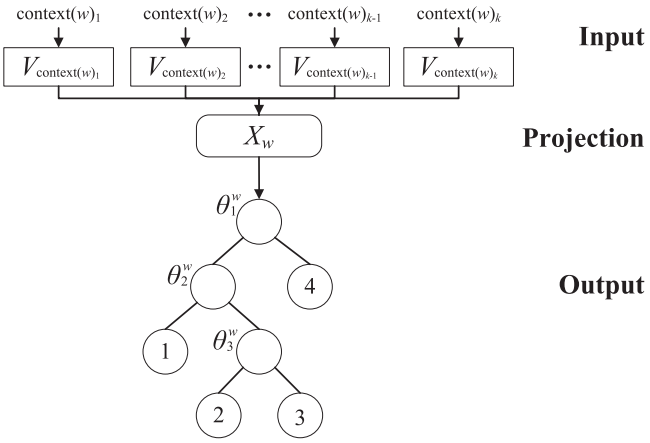

FIGURE 2 The architecture of the CBOW model

w. If $\boldsymbol{X}_w$ predicts the word 3 in Figure 2, then three branches are required to go through to reach it with each branch being a binary classification. In Word2Vec, 1 is defined as a negative class and 0 is defined as a positive class. The probability of being the positive class at the node θ_2^w is defined as

$$p(0|\boldsymbol{X}_w, \theta_2^w) = \sigma(\boldsymbol{X}_w^{\mathsf{T}}\theta) = \frac{1}{\left(1 + \mathrm{e}^{-\boldsymbol{X}_w^{\mathsf{T}}\theta}\right)}. \tag{1}$$

For each word in the dictionary, the output layer must have a binary classification path that directs to the word. Then, the probability language model $p(w|\text{context}(w))$ can be defined as the product of the probability of all nodes in this path

$$p\left(w|\text{context}(w)\right) = \prod_{j=2}^{J}\left(d_j^w|\boldsymbol{X}_w, \theta_{j-1}^w\right), d_j^w \in \{0, 1\}, \tag{2}$$

where J is the number of nodes in the binary classification path.

By maximizing the value of $p(w|\text{context}(w))$, we can get the word embedding $\boldsymbol{V}_w$ corresponding to word w. From the above analysis, we know that Word2Vec can capture the semantic information of the words in the process of training the language model. The semantic environment of the target word is reflected by the surrounding words, and it is represented as a vector.

3.2 | Implementation

To guarantee that the similarity retrieval models can offer an optimal or near optimal performance and predict the performance analytically, Fang and Zhai[1] presented an axiomatic retrieval framework based on direct modeling of relevance at a level of terms. The optimal retrieval function they derived is defined as follows:

$$S(Q, D) = \sum_{t \in Q} c(t, Q) \times \left(\frac{N}{df(t)}\right)^{0.35} \times \frac{c(t, D)}{c(t, D) + 0.5 + \frac{0.5 \times |D|}{\text{avdl}}}, \tag{3}$$

where Q is the user query, D is the document, $c(t, Q)$ is the occurrences of term t in Q, $df(t)$ is the document frequency of term t, N is the total number of documents, $c(t, D)$ is the occurrences of term t in document D, and avdl is the average of the number of terms in all documents.

Although the above retrieval function has achieved good results in practice, Equation 3 does not capture the semantic information of the user query and documents. Therefore, Huang et al[6] improved Equation 3 as follows:

$$S(Q, D) = \sum_{t \in Q} \text{Sim}(t, Q) \times \left(\frac{N}{\text{Sim\_df}(t)}\right)^{0.35} \times \frac{\text{Sim}(t, D)}{\text{Sim}(t, D) + 0.5 + \frac{0.5 \times |D|}{\text{avdl}}}, \tag{4}$$

where $\mathrm{Sim}(t, Q)$ is the occurrences of similar terms corresponding to term t in the user query Q, $\mathrm{Sim\_df}(t)$ is the document frequency of similar terms corresponding to term t, and $Sim(t, D)$ is the occurrence of similar terms corresponding to term t in document D. Huang et al[6] used WordNet to depict the similarity between words, and the similarity threshold was determined experimentally.

WordNet and HowNet are two popular synonym dictionaries in English and Chinese, respectively. Because their contents are maintained manually in a highly professional field, the similarity between a large number of professional vocabulary cannot be calculated because vocabulary absence in the dictionary, eg, "alzheimer" and "dementia" in the field of healthcare. In addition, the problem of polysemy cannot be solved well in a highly professional field. For example, the similarity between "male" and "female" is 0.86 using the method presented in the work of Liu and Li.[17] From the perspective of species, these two words are indeed highly similar. However, they should be processed differently in the field of healthcare, ie, only "male" can have prostate diseases, and "female" can have gynecological diseases. If the score 0.86 is used in the retrieval of "male" as keywords, then the documents about gynecological diseases will also be returned. Absolutely, this is unreasonable. To address the problem of semantic gap, word embeddings are taken into account, as opposed to synonym dictionaries. Therefore, in the word embedding vector space, "gynecology" and "female" are similar.

Based on the above analysis, we use word embeddings to depict the semantic similarity between words. $\mathrm{Vec}_w = [v_1, v_2, \ldots, v_k]$ is the embedding representation of word w and k is its dimension. Then the semantic distance of two words can be represented by cosine similarity which can be calculated using Equation 5

$$\mathrm{SemanticSim}_{ab} = \frac{\sum_{i=1}^{k} \mathrm{Vec}_a[i] \times \mathrm{Vec}_b[i]}{\sqrt{\sum_{i=1}^{k} \mathrm{Vec}_a[i]^2} \sqrt{\sum_{i=1}^{k} \mathrm{Vec}_b[i]^2}}. \tag{5}$$

Given the threshold of word similarity $\mu (0 < \mu \leq 1)$ and the number of documents N to return, our proposed semantic similarity algorithm can be expressed as follows.

Inputs: A user query and documents collection.

Outputs: A list of documents.

Step 1: Prepare the corpus, and initialize the vocabulary;
Step 2: Employ Word2Vec to train the word embeddings for each word w in the vocabulary;
Step 3: Traverse all documents and calculate the document frequency of term t, $df(t)$, without considering the word similarity;
Step 4: Find the documents containing similar terms corresponding to term t by Equation 5, where the similarity between words should be larger than μ;
Step 5: Based on the equation 4, compare user query with documents, and generate a relevance list in the descending order;
Step 6: Return top N of the relevance list as retrieval results.

4 | EXPERIMENTAL RESULTS

Two different datasets are employed to evaluate the proposed algorithm. One is the public Reuters-21578 dataset, and the other is collected from the 120ask healthcare question answering community.

Reuters-21578 is a widely used dataset for text categorization, and it has two dataset splits, ie, ModApte and ModWiener. The ModApte split was used in this work including 9,603 documents in the training set and 3,299 documents in the test set. The documents were classified by topics, and the total number of topic keywords was 135, while there were 92 topic keywords in the test set. In this work, we employed the test set to do retrieval by topic keywords.

120ask is a healthcare question answering community in China. We randomly extracted 10,000 questions from ten categories (eg, Chinese medicine, liver disease, and gastroenteritis) to build the corpus. For each category, we constructed 10 questions (totally 100 test questions), which annotated with a collection of relevant questions. In addition, in order to ensure the correctness of the word segmentation, we collected 132,371 medical proper nouns from Sogou[†], covering items such as drug names and disease names. The Skip-gram model was used to train word embeddings, along with Huffman tree as the output layer. The dimension of the word embeddings is 50, and the window size is 5.

Table 1 is the experimental results, showing a comparison of the precision at top 10 results between the Axiomatic algorithm as presented in Equation 3 and the proposed word embedding-based method.

The value in the brackets in Table 1 is the threshold of word similarity. These thresholds were chosen based on the fact that, under these values, the precisions on two datasets were maximums, which were illustrated in Figures 3 and 4, respectively. From the above experimental results, the precision of the retrieval results has improved in both datasets by our proposed semantic similarity retrieval method based on word embeddings. In Reuters-21578 dataset, the precision has improved by 0.3%, which is mainly affected by the size of the corpus, and the expression ability of word embeddings trained from the corpus is limited. In addition, we find that, for the "Mergers/Acquisitions (ACQ)" keyword, the precision in Axiomatic

[†] http://pinyin.sogou.com/dict/

TABLE 1 Top 10 results on two datasets

Algorithm	Dataset	
	Reuters-21578	120ask
Axiomatic	0.354	0.580
Proposed Method	**0.357 (0.97)**	**0.754 (0.95)**

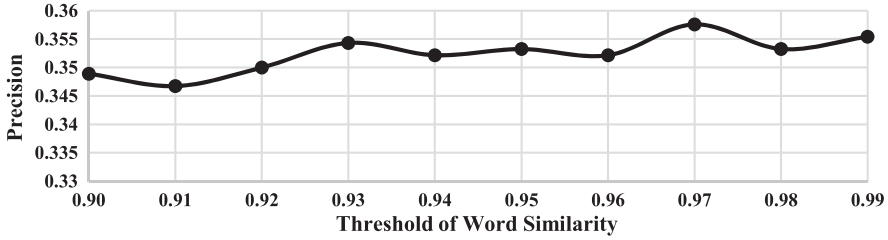

FIGURE 3 The precision of the Reuters-21578 dataset under the threshold of word similarity from 0.90 to 0.99

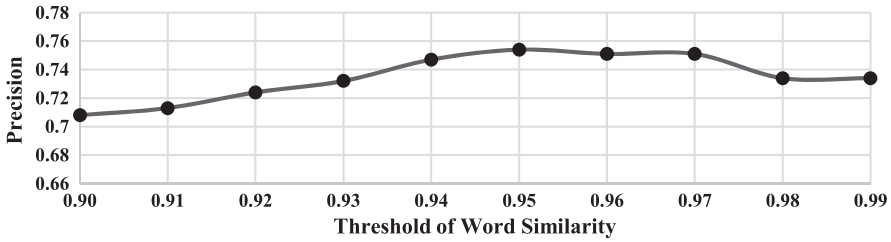

FIGURE 4 The precision of the 120ask dataset under the threshold of word similarity from 0.90 to 0.99

TABLE 2 The Sim_df(t) values calculated by Word2Vec and HowNet, respectively

Word	Word Similarity Algorithm	Sim_df(t)
body	Word2Vec	222
	HowNet	3020
chest	Word2Vec	40
	HowNet	1195

algorithm is 20%, but the precision in word embedding-based method is 60%. It can be explained that word embeddings can improve semantic similarity retrieval.

In the 120ask dataset, word embedding is effective due to the fact that the corpus for training is rich, and the expression ability of word embeddings is strong. Another reason is that word embeddings are learned from the corpus, avoiding the vocabulary absence in the synonyms dictionary. For example, although "acne" is not included in the synonyms dictionary, it can also be represented by the context words, which greatly improves the retrieval results.

It should be noted that the synonym dictionary has the problem of missing vocabulary, but we also find that even if a word is included in the dictionary, the word similarity calculated based on the synonym dictionary does not well characterize the differences between words. For Equation 5, we do an in-depth analysis of Sim_df(t) (the number of documents, which contain similar terms corresponding to term t). Table 2 shows the Sim_df(t) values calculated using HowNet and the open sourced xsimilarity package[17] using the "body" and "chest" as examples in the 120ask dataset.

From Equation 4, we can conclude that the larger the Sim_df(t) value is, the smaller the similarity $S(Q, D)$ would be, and the precision of retrieval results would decrease. From Table 2, the HowNet-based algorithm tends to find more similar words, which leads to the poor performance on the 120ask dataset.

5 | CONCLUSIONS

In this paper, we have presented an improved similarity retrieval using word embeddings to fill semantic gaps caused by missing synonyms. We employed Word2Vec to train the word embeddings and then applied them to depict the semantic similarity between words. Experimental results on the two datasets showed that the proposed algorithm further improves the retrieval precision.

In future works, we will focus on the training of word embeddings, comparing the effects of representative word embedding algorithms on retrieval precision, and trying to get more complete word embeddings with external corpus.

ACKNOWLEDGMENT

Our work was supported by Science and Technology Commission of Shanghai Municipality under grant 16511102800.

ORCID

Mingming Lu http://orcid.org/0000-0002-4762-1280

REFERENCES

1. Fang H, Zhai CX. An exploration of axiomatic approaches to information retrieval. Paper presented at: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval; 2005; Salvador, Brazil.
2. Sun ZJ, Zheng Q, Yuan J, Liu H, Wang S. Semantic retrieval based on shallow semantic analysis technology. *Comput Sci*. 2012;39(6):107-110.
3. Chen Y, Lin SP. Technology of semantic search based on ontology. *Comput Eng Appl*. 2006;S1:78-80
4. Bhogal J, MacFarlane A, Smith P. A review of ontology based query expansion. *Inf Process Manag*. 2007;43(4):866-886.
5. Yong LI, Zhang ZG. Semantic retrieval research based on ontology. *Comput Eng Sci*. 2008;30(4):17-18.
6. Huang CH, Yin J, Lu JY. An improved retrieve algorithm incorporated semantic similarity for Lucene. *Acta Sci Nat Univ Sunyatseni*. 2011;50(2):11-15.
7. Kim Y. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882; 2014.
8. Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473; 2014.
9. Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. *Nature*. 1986;323(6088):533-536.
10. Bengio Y, Ducharme R, Vincent P, Jauvin C. A neural probabilistic language model. *J Mach Learn Res*. 2003;3:1137-1155.
11. Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781; 2013.
12. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J. Distributed representations of words and phrases and their compositionality. Paper presented at: 2013 Advances in Neural Information Processing Systems; 2013; Lake Tahoe, Nevada.
13. Apté C, Damerau F, Weiss SM. Automated learning of decision rules for text categorization. *ACM Trans Inf Syst (TOIS)*. 1994;12(3):233-251.
14. Studer R, Benjamins VR, Fensel D. Knowledge engineering principles and methods. *Data Knowledge Eng*. 1998;25(1-2):161-197.
15. Miller GA. WordNet: A lexical database for English. *Commun ACM*. 1995;38(11):39-41.
16. Dong ZD, Dong Q. HowNet - a hybrid language and knowledge resource. Paper presented at: Proceedings of the 2003 International Conference on Natural Language Processing and Knowledge Engineering; 2003; Beijing, China.
17. Liu Q, Li SJ. Word similarity computing based on how-net. *Comput Linguist Chin Lang Process*. 2002;7(2):59-76.
18. Sun CN, Zheng C, Xia Q. Chinese text similarity computing based on LDA. *Comput Technol Dev*. 2013;23(1):217-220.
19. Lai SW, Xu LH, Liu K, Zhao J. Recurrent convolutional neural networks for text classification. Paper presented at: Proceedings of the 29th AAAI Conference on Artificial Intelligence; 2015; Austin, TX.
20. Li SH, Chua TS, Zhu J, Miao CY. Generative topic embedding: a continuous representation of documents. Paper presented at: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics; 2016; Berlin, Germany.
21. Kobayashi H, Noguchi M, Yatsuka T. Summarization based on embedding distributions. Paper presented at: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing; 2015; Lisbon, Portugal.
22. Zamani H, Croft WB. Embedding-based query language models. Paper presented at: Proceedings of the ACM on International Conference on the Theory of Information Retrieval; 2016; Newark, DE.
23. Kusner M, Sun Y, Kolkin N, Weinberger K. From word embeddings to document distances. Paper presented at: 32nd International Conference on Machine Learning; 2015; Lille, France.

How to cite this article: Yan F, Fan Q, Lu M. Improving semantic similarity retrieval with word embeddings. *Concurrency Computat Pract Exper*. 2018;30:e4489. https://doi.org/10.1002/cpe.4489