Sentence Similarity Computation by Integrating Shallow and Deep information

Pengyuan Liu[1], Zhijun Zheng[3]
School of information science
Beijing Language and Culture University
Beijing, China
liupengyuan@pku.edu.cn, zzj@blcu.edu.cn

Qi Su[2]*
School of Foreign Languages
Peking University
Beijing, China
sukia@pku.edu.cn

***Abstract*—The calculation of sentence similarity has always been a hot topic in the field of natural language processing. The existing methods of sentence similarity computation usually consider either shallow or deep information of sentences. In this paper, we propose a method to integrating the two types of information. In the model, the shallow information is acquired based on word similarity and sentence length. The deep information is extracted via a parallel convolutional neural network. Then we integrate the two parts linearly. Experiments on SemEval 2007 task 5, similarity evaluation between Chinese and English sentences, show that the performance of our integration model is better than the models which treat the two types of information separately. For parallel convolutional neural network, a performance improvement is evident when shared parameters are assigned.**

Keywords-sentence similarity; parallel convolutional neural network; integration model

I. INTRODUCTION

The calculation of the similarity between sentences plays an important role in the field of natural language processing. The general procedure of calculating sentence similarity is: given a pair of sentences, we first represent them as feature vectors. Based on the vectors, a similarity score can then be computed with different methods. Currently there are mainly two scoring systems being developed. One takes a range of values from 0 to 1; the other from 0 to 5. Here 0 means that the two sentences are completely irrelevant in semantics; 1 or 5 means that the pair of sentences shares the same meaning. Most existing methods for sentence similarity calculation consider either shallow or deep information encoded in sentences for the representation of sentences. This paper proposes a sentence similarity calculation model that integrates both shallow and deep information, in which shallow information is obtained based on both lexical similarity and sentence length; deep information is extracted via a parallel convolutional neural network. The model integrates the two types of information in a linear manner.

The remainder of the paper is organized as follows. In section 2 we discuss the related work. Section 3 gives a detailed description of our model. The experimental results and analysis are given in section 4. Finally, section 5 presents our conclusions and future works.

II. RELATED WORK

Based on whether sentences are modeled at the semantic level, the relevant research can be divided into shallow methods and deep methods.

A. Shallow methods

Lexical semantic based sentence similarity computation [5] calculates the semantic similarity of words in two sentences, and then further computes the similarity of the sentences. The key point in the process is the measure of similarity between words. Semantic dictionaries such as WordNet, HowNet and Cilin are commonly used for it. Structure-based sentence similarity calculation methods mainly consider words and their positions in sentences. Different usages of a word in different positions, especially in English, can reflect the internal information of a sentence. Correspondingly, edit distance between sentences are widely used in measuring sentence similarity [6]. Using Chinese words as calculating units, Chen [7] adapted the generalized edit distance to make it more suitable for the similarity calculation of Chinese sentences. In addition to the above-mentioned methods, there are a number of other approaches which are based on keywords ranking, dependency syntax and so on. Methods based on vector space model [8-10] count the frequency (or calculate the TF-IDF) of each word in corpus, and represent a sentence with word frequencies (or TF-IDFs). Then the similarity between two sentences can be calculated by Euclidean distance or cosine measure.

B. Deep methods

Recently, neural network based approach has gained prominence for natural language processing applications. The approach has also been widely adopted in sentence similarity calculation, with the advantage of capturing deep semantic information encoded in sentences by representing them with neural networks. For example, in a CNN-based parallel semantic matching model [11], two sentences are taken as input into two convolutional neural networks individually to derive their semantic representations. Then the two representations are fed into a multi-layer neural network for checking semantic matching. To solve the problem of non-interaction between an interdependent sentence pair in parallel CNN, researchers introduced the mechanism of "attention" [12]. When dealing with long sentences, the problem of long distance dependence still cannot be solved by traditional CNN. Therefore, researchers further designed Siamese Network [13] for complex sentences. Some researchers believe that the semantics of a sentence can be divided into similar parts and dissimilar parts, both of which are helpful for the calculation of sentence similarity [14]. Through the combination of the two parts, sentence similarity can be calculated more effectively.

978-1-7281-1175-9/18/$31.00 ©2018 IEEE

III. OUR MODEL

The sentence similarity calculation model proposed in this paper is shown in Figure 1, which mainly includes three components: shallow module, deep module, and integration module.

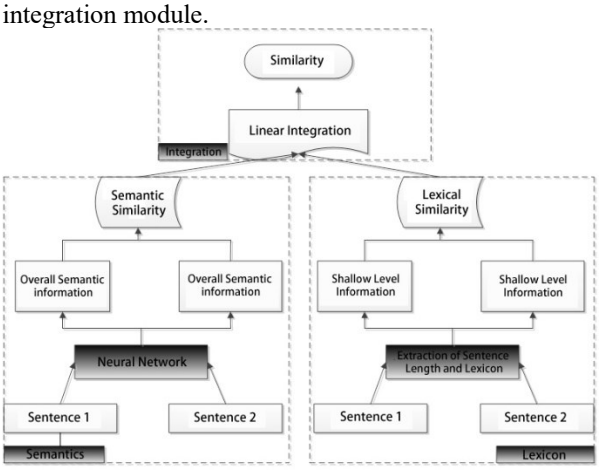

Figure 1. Model of Sentence Similarity Computation by Integrating Lexical and Semantic Information

A. Shallow Module

Figure 2 shows the shallow module, which comprises two parts. The right part calculates the similarity of word embeddings based on keywords; and the left measures the difference in sentence lengths. After the calculations, we compute lexical similarity by a linear subtraction of the outputs of the two parts.

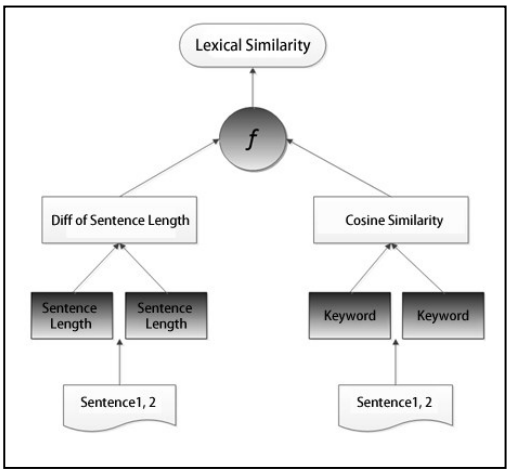

Figure 2. Lexical Similarity

Calculation of Word Embedding Similarity

Firstly, keywords in a sentence pair are extracted. Then, we get the vector representation of each keyword using a pre-trained word embedding. Equation 1 is used to calculate the similarity based on word vectors

$$Wv = Cos(\sum_{i\in n} V(S_i), \sum_{j\in m} V(S_j)) \quad (1)$$

where $V(S_i)$ and $V(S_j)$ represent the word vectors of the i-th and j-th keywords in two sentences, respectively. Cos represents the cosine similarity operation.

Difference of Sentence length

It is generally believed that, for two sentences, the greater the difference between their lengths, the lower their similarity. The reduction of the similarity caused by the length difference between two sentences is shown as:

$$DifLen(A,B) = \frac{|Len(A)-Len(B)|}{(Len(A)+Len(B))} \quad (2)$$

In the equation, $|\mathrm{Len(A)} - \mathrm{Len(B)}|$ represents the absolute value of the length difference between two sentences. $|\mathrm{Len(A)} + \mathrm{Len(B)}|$ is the sum of sentence lengths. $\mathrm{DifLen(A,B)}$ denotes the value of reduced similarity caused by the length difference of two sentences. f shown in Figure 2 is the similarity calculation operation with lexical and sentence length information, as presented in Equation 3, where $\mathrm{Wv(A,B)}$ represents the similarity calculated based on the keyword vector of the sentences, $\mathrm{Sim\_lex(A,B)}$ denotes lexical similarity.

$$Sim\_lex(A,B) = Wv(A,B) - DifLen(A,B) \quad (3)$$

B. Deep module

Figure 3 shows the deep module using parallel CNN [16]. It has both high performance and satisfactory time efficiency even when the sentence length is short and the number of sentences is large. Different from traditional CNN models, the model used in this paper extracts sentence features from both the context of sentences and the whole sentences. The CNN model comprises of four components: input layer, convolution layer, pooling layer, and output layer.

Input layer. Each CNN contains an input layer in which sentences are transformed into a feature matrix composed by word embeddings. Suppose $\mathrm{X_i} \in \mathrm{R^k}$ represents the i-th term in the sentence whose word embedding dimension is k. Then a sentence of n words in the input layer is represented as:

$$\mathrm{X}_{1:n} = X_1 \theta X_2 \theta \cdots \theta X_n \quad (4)$$

Here, θ represents a concatenation operation. The whole sentence is concatenated into a k×n matrix (fill zero for empty cell).

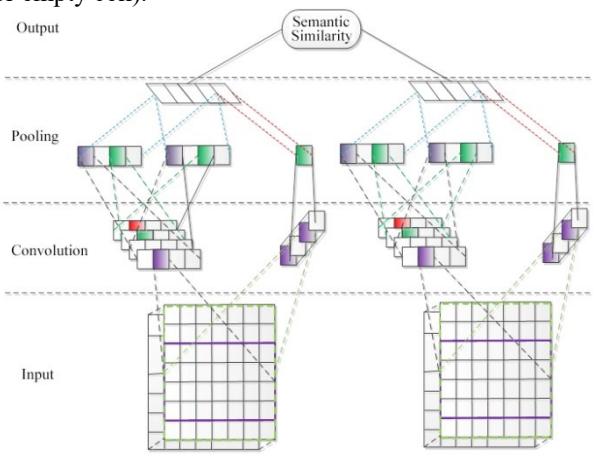

Figure 3. Deep module

Convolutional layer. Convolutional layer is the core layer in convolutional neural network. The operation of convolution is the process of feature extraction. Using a filter $W \in R^{hk}$, C_i is the set of new features generated after h words are convoluted for each time, which is shown in Equation 5:

$$C_i = f(W \cdot X_{i:i+h-1} + b) \quad (5)$$

where f denotes the nonlinear activation function, W denotes the filter, $X_{i:i+h-1}$ is the upper layer features covered by the filter, and b is a bias.

It is difficult for traditional CNNs to encode the entire information of a single sentence, so they usually lack the features at the whole sentence level. To remedy the above-mentioned defects, we propose to convolute information of whole sentences with a conventional convolution (in this case, the height of the filter is sentence length). With the help of the context filter and the filter of whole sentences, we can simultaneously exact the semantic information of sentences from two perspectives.

Pooling layer. The specific operation of pooling can be seen in Equation 6.

$$C' = op\{C_1, C_2 \cdots C_{n-h+1}\} \quad (6)$$

In the equation, $\{C_1, C_2 \cdots C_{n-h+1}\}$ is the feature transmitted by the upper convolution layer. op denotes a specific pooling operation. Average pooling extracts the average information and maximum pooling selects the most obvious information. In order to extract convolutional information from multiple aspects, we combine the two operations of average and maximum pooling together. Equation 7 shows the pooling operation used in the experiments, where $avg_{C'}$ and $max_{C'}$ represent the average and the maximum pooling respectively. ∂ is a hyperparameter.

$$\hat{C} = avg_{C'} + \partial(avg_{C'} - max_{C'}) \quad (7)$$

For the features convoluted from whole sentences, we directly extract their maximum values. The maximum value of concatenating the context pooling results and the results of convoluting the whole sentences is the space vector which represents the semantic information of the sentences.

Output layer. For the sentence vectors from upper layer, the output layer needs to calculate the Euclidean or cosine distance between sentence pairs, which is used to represent the similarity between them (Cosine distance is used in this paper).

C. Integration of Modules

The above two modules calculate the similarity of sentence pairs from shallow and deep levels respectively. The integration of the two modules can calculate the correlation between sentence pairs more accurately. Therefore, we further integrate the two parts linearly.

$$sim(A, B) = (1 - \alpha) * sim\_sem + \alpha * sim_{lex} \quad (8)$$

$(1 - \alpha)$ and α indicates the proportion of semantic similarity and lexical similarity. sim_sem is the semantic similarity and sim_lex denotes the lexical similarity.

IV. EXPERIMENTS

A. Dataset and Evaluation Metric

For the experiments, we use the fifth subtask of SemEval2017 task1, which focuses on the similarity evaluation of English sentences. A sample from the testing set is shown in Table 1. There are a total of 250 pairs of sentences in the set. Each pair has been assigned with a similarity score in an interval of [0, 5], with 5 being the most similar and 0 being the least similar. There is no training set being provided by SemEval2017. So, the training set used in this paper is the English sentence pairs distributed by SemEval for similarity evaluation during the period of 2012-2015.

TABLE I. SAMPLE FROM TESTING SET

Sentence 1	*Sentence 2*	*Similarity*
a person is on a baseball team.	a person is playing basketball on a team	2.4

The evaluation metric is Pcc [1] (Pearson Correlation Coefficient), which measures the correlation between two variables. The larger the Pcc, the stronger the correlation, and vice versa. Equation 9 shows the detail of it:

$$\rho_{xy} = \frac{Cov(X,Y)}{\sqrt{D(X)}\sqrt{D(Y)}} = \frac{E((X-EX)(Y-EY))}{\sqrt{D(X)}\sqrt{D(Y)}} \quad (9)$$

where E is the mathematical expectation, D is the variance. $\sqrt{D(X)}$ and $\sqrt{D(Y)}$ are the standard deviations. $E((X - EX)(Y - EY))$ is the covariance of the random variables X and Y, which is also denoted as $Cov(X, Y)$.

B. Experimental results and analysis

This paper designs three deep models based on parallel CNN: shared parameter parallel CNN, partially shared parameter parallel CNN, and non-shared parameter parallel CNN. From Table 2, we can see that with the increase of shared information, the performance of model is getting better. The main reason is that the pairs of sentences used in the experiments are from the same language. The features of two sentences distribute in the same space. Therefore, using shared parameters can better fit their distribution.

TABLE II. EXPERIMENTAL RESULTS WITH DEEP MODEL

Experiments	*Results(Pcc)*
Shared parameter parallel CNN	**79.4%**
Partially shared parameter parallel CNN	78.2%
Non-shared parameter parallel CNN	66.5%

[1] https://en.wikipedia.org/wiki/Pearson_correlation_coefficient

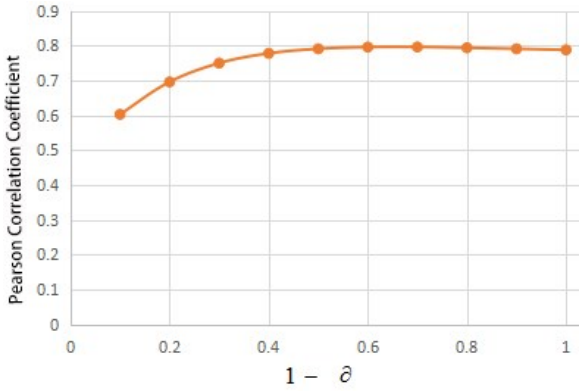

Figure 4. Interaction of Shallow and Deep Information in the Calculation of Sentence Similarity

Shallow and deep information are integrated in a linear way. By adjusting parameter α in equation 8, the experimental results shown in Figure 4 are obtained. From Figure 4, it can be seen that the shallow and the deep information are interacting during the calculation of similarity. The integration of semantic information and lexical information is beneficial to the calculation of similarity. Pearson coefficient gets the best result of 79.93% at 1-α=0.8. This result is 0.53% better than the single shared parameter parallel CNN (1-α=1) (see Table 3). The main reason for the result is that the shared parameter parallel CNN only calculates sentence similarity from a deep level, while the integration model simultaneously calculates sentence similarity from both deep and shallow aspects.

TABLE III. COMPARISON OF EXPERIMENTAL RESULTS

Experiments	*Results(Pcc)*
CNN with Shared parameters +Shallow information	**79.93%**
CNN with Shared parameter	79.4%

V. CONCLUSION

This paper proposes a sentence similarity calculation model that integrates shallow and deep information encoded in sentences. The shallow part is based on both sentence length and lexical similarity, and the deep part is based on parallel convolutional neural network. Therefore, this model can extract features of sentences from both shallow and deep aspects. Experiments on SemEval2017 task5 showed that, shallow information and deep information are interacting in the calculation of sentence similarity. Integrating the two types of information can improve the performance of sentence similarity calculation. The performance of parallel CNN is higher when shared parameters are assigned.

Currently shallow information of sentences was extracted based on the similarity of sentence lengths and that of lexicons. Deep information was extracted via double convolutional neural networks. A simple linear integration operator was used to combine the two parts of information. In future works we will try to develop different methods of extracting the two parts of information and integrate it in more effective ways for better performance.

VI. ACKNOWLEDGEMENTS

This work was supported in part by MOE (Ministry of Education in China) Project of Humanities and Social Sciences (No. 18YJA740030), and Beijing Social Science Foundation (No.14WYC041). Qi Su is the corresponding author of this paper.

REFERENCES

[1] S Niwattanakul. Using of Jaccard Coefficient for Keywords Similarity[J]. Proceedings of Lecture Notes in Engineering & Computer Science, 2013.

[2] Grzegorz Kondrak . N-Gram Similarity and Distance[J]. Proceedings of LNCS, 2005.

[3] Guo S and Xing D. Sentence similarity calculation based on word vector and its application research [D]. Modern Electronics Technique, 2016. (in Chinese)

[4] Metzler D. Generalized inverse document frequency. ACM Conference on Information and Knowledge Management. New York, 2008: 399–408.

[5] Li G. Research and Application of Similarity Calculation in Chinese Texts [D]. Sichuan: University of Electronic Science and Technology of China, 2011. (in Chinese)

[6] Zhou F. and Yang B. New method for sentence similarity computing and its application in question answering system [J]. Computer Engineering and Applications, 2008. (in Chinese)

[7] Chen W. Liu T. Qin B. Li S. Similar Chinese Sentence Retrieval based on Improved Edit-Distance [J]. High Technology Letters, 2004. (in Chinese)

[8] Landauer T K, Foltz P W, Laham D. Introduction to latent semantic analysis. Discourse Processes, 1998, 25(3): 259–284.

[9] Lund K, Burgess C. Producing high-dimensional semantic spaces from lexical co-occurrence. Behavior Research Methods, 1996, 28(2): 203–208.

[10] Allan J, Wade C, Bolivar A. Retrieval and novelty detection at the sentence level. Proceedings of Association for Computing Machinery Special Interest Group on Information Retrieval Conference on Research and Development in Information Retrieval. Dresden, 2003: 314–321.

[11] Hu B, Lu Z, Li H, et al. Convolutional neural network architectures for matching natural language sentences[C]. Advances in Neural Information Processing Systems, 2014.

[12] Wenpeng Yin. Attention-Based Convolutional Neural Network for Modeling Sentence Pairs [J]. Proceedings of TACL, 2016.

[13] Jonas Mueller. Siamese Recurrent Architectures for Learning Sentence Similarity [J]. Thirtieth AAAI Conference on Artificial Intelligence, 2016.

[14] Zhiguo Wang. Sentence Similarity Learning by Lexical Decomposition and Composition [J]. Computation and Language. 2017.

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In Proceedings of NIPS, pages 3111–3119.

[16] Omer Levy. Neural word embedding as implicit matrix factorization. Proceedings of Advances in Neural Information Processing Systems, 2014.