A Semantic Similarity Computing Model based on Siamese Network for Duplicate Questions Identification

Zongkui Zhu[1], Zhengqiu He[1], Ziyi Tang[2], Baohui Wang[3], Wenliang Chen[1]

[1] School of Computer Science and Technology,
Soochow University, Suzhou, Jiangsu, 215006, China
[2] College of Electrical and Information Engineering,
Hunan University, Changsha, Hunan, 410082, China
[3] Laiye Inc, Beijing, 100080, China

Abstract. Traditional semantic similarity computing methods mostly regard the text as a set of words, by calculating the number of words occurred in the text to build the feature vector, then using the metrics such as cosine distance between the vectors to calculate the text similarity. However, these methods only consider the word level of the sentence, not the semantic level, which may ignore many important information, including syntax and word order. This paper proposes a new deep learning method, which combines the attention mechanism with Bi-LSTM based on Siamese network to achieve the semantic similarity matching for given question pairs. Experimental results show that our models can make full use of the semantic information of the text, and the F1 value in the dataset provided by the CCKS2018 question-intention matching task is 0.84586, achieving fourth place in the final test.

Keywords: Sentence Matching, Semantic Similarity, LSTM, Attention Mechanism, Siamese Network

1 Introduction

The goal of CCKS2018 question-intention matching (QIM) task is to match the question pairs against Chinese real-world customer service data. In the task, we should design models and algorithms to determine which of the provided pairs of questions contain two questions with the same intention. For example, given a sentence pair: "一般几天能通过审核" and "一般审核通过要多久", we need to determine whether they convey similar meaning (label 1) or not (label 0).

Sentence matching, also called semantic equivalence identification or short text semantic similarity computing, is one of the most basic tasks of NLP. It is the basis of various NLP tasks such as automatic question and answer, chat robot, information retrieval, and machine translation. In recent years, with the rapid development of Internet technology, various data information shows explosive growth, and the demand for processing large amounts of similar data is increasing. Although researchers have advanced the semantic similarity task, there are many challenges remained.

- If the requirements for question matching are different, the definition of matching will be not the same. For example, in classic sentence retelling task, we need to judge whether the two sentences are merely different in expression, but the meaning is the same.
- We often face short sentences and the complex structures in the task. Thus, it is difficult to extract informative features purely based on the sentence itself.
- The data of QIM task comes from the log text of original smart customer service in the field of bank, so there is much dirty data, such as mislabeling, typos, missing words and simplified words, which can easily lead to word segmentation errors, including the identification of the unknown words.
- In addition, the training of word vectors also lacks corpus data for specific financial fields, which associated with this task.

In this paper, we propose a semantic matching method based on Siamese network [1] for this task. For the similarity computing, instead of using the traditional artificial designed features, we adopt the neural network to learn automatically the information contained in the question pairs. In this way, it can mine the potential syntactic and semantic features in the sentences, which effectively compensates for the shortcomings in the manual extraction of features.

2 Related Work

The core of text similarity computing is to compare the differences between two given texts, usually measured by a real value between 0 and 1. In different fields, domestic and foreign scholars have proposed many methods of the similarity computing and applied them in practice. At present, these methods can roughly divide into four categories: 1) string-based method, which is also called “literal similarity method” and the typical methods include edit distance, Hamming distance, Euclidean distance, cosine similarity, longest common substring, Jaccard, and so on. Since the string-based method does not consider the semantic information of the text, the effect of the method is limited. 2) corpus-based method, which uses the information obtained from the corpus to calculate the text similarity, and can subdivide into a word-bag-based model approach, a neural network-based approach, and a search engine-based approach. The first two types use the document collection that will compare as the corpus, and the latter is a web-based corpus. 3) world-knowledge-based method, which makes use of a knowledge base with a standardized organizational system to calculate text similarity. 4) other methods, which include syntax analysis and mixing methods [2].

Among them, the neural network-based method can represent the text semantic information more in line with human cognition, and has become the development trend of the task, which solves many drawbacks of traditional methods. For example, the InferSent model proposed by Conneau et al. [3], which based on a Bi-LSTM architecture with max pooling, and the ESIM model proposed by Chen et al. [4], which based on chain LSTMs and attention, both preform well on natural language inference (NLI) , a task similar to semantic similarity computing.

3 Our Approach

Our work can divide into three parts: data preprocessing, model building and post-processing.

3.1 Data Preprocessing

Because the data comes from the log text of original smart customer service in the field of bank, many sentences are informal and contain much noise. The main issues are listed as follows:

- Traditional Chinese characters such as “何時會邀請我使用微粒貸” and “为什么我的微信上还没有出现微米粒”, the first one is written in Traditional Chinese and the second is in Simplified Chinese. If the sentences are used directly to train the neural network without preprocessing, it will affect the accuracy of the prediction to some extent.
- Typos, such as “预期罚息和利息” and “逾期利息”. “预期” should be “逾期” in the first sentence of the two sentences.
- Too short, such as “我找” and “你老板呢”, the two sentences are very short and it is not easy to extract useful feature information.
- Redundant Character, such as “用微信都6年，微信没有微粒贷功能 ” and “4。 号码来微粒贷”, there are both extra numbers, punctuation marks such as spaces in the two sentences, which also affect the performance of the model.

Based on the above problems, we carry out the following specific preprocessing on the data: 1) use the open source OpenCC [5] tool to convert the data from traditional to simplified, 2) remove the extra space characters, Chinese and English punctuation marks and numbers, 3) convert all the letters to lowercase. In view of the fact that the word segmentation tool cannot handle the specific domain data well, and the experimental results show that the char level is obviously better than word level when tested. For the lack of training corpus for word vector, we use the training data provided by the task to train embedding at the char-level.

3.2 Model Building

To calculate the semantic similarity of the question pair, that is, whether the question pair belongs to label 1 or 0. Obviously, this is a classification problem in the field of NLP, but it is different from the traditional text categorization method.

In order to represent the feature information of two questions at the semantic level and capture the commonalities between them, we designed the following two models based on Siamese network. The Siamese network is a neural network framework with two identical structures and a shared weight subnetwork for evaluating the similarity of two input samples.

Figure 1 and Figure 2 show the entire frameworks of the semantic similarity computing models. It can be found easily from the figures that the structure of the two models is similar, they both contain Bi-LSTM layer, self-attention layer, non-linear layer, matching layer and MLP layer.

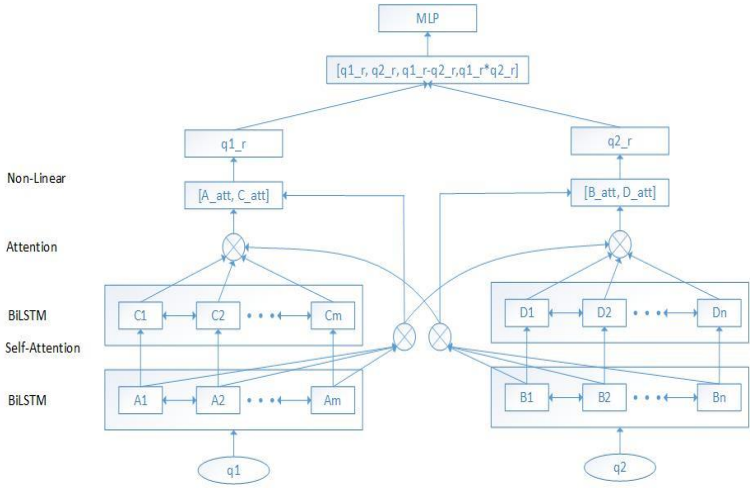

Fig. 1. The overview architecture of semantic similarity model-I.

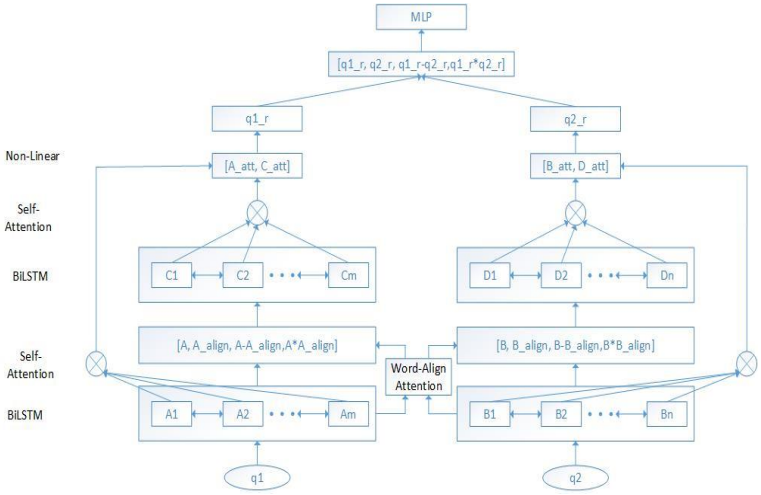

Fig. 2. The overview architecture of semantic similarity model-II.

Let $q1 = [a_1, a_2, \dots, a_m]$ and $q2 = [b_1, b_2, \dots, b_n]$ be the given two sentences with length m and n respectively, where $a_i, b_j \in \mathbf{R}^d$ is a char embedding of d-dimension vector. The task is to predict a label y that indicates the similarity relationship between q1 and q2.

For **model-I**, we apply bidirectional long short-term memory network (Bi-LSTM) [6] as the basic component of our model, since LSTM unit can effectively extract the dependence among long-distance and non-continuous words in sentences, which is more suitable for the semantic representation of the word sequence in the sentence. And the bidirectional structure can provide complete past and future context information of each node in a sentence for the output layer. Firstly, we use two layers of Bi-LSTM to

encode the word vector input for the given two sentences q1 and q2, which combines the forward and backward outputs to obtain a textual representation of each sentence. The outputs of the first and second layers of the two sentences as follows:

$$\begin{aligned} A &= \mathrm{BiLSTM}(\mathrm{q1}) \\ \mathrm{B} &= \mathrm{BiLSTM}(\mathrm{q2}) \\ \mathrm{C} &= \mathrm{BiLSTM}(\mathrm{A}) \\ \mathrm{D} &= \mathrm{BiLSTM}(\mathrm{B}) \end{aligned} \tag{1}$$

where $A, C \in \mathbf{R}^{m*2d}$ and $B, D \in \mathbf{R}^{n*2d}$ are the outputs of the two layers of Bi-LSTM respectively. Moreover, the two questions share the parameters of Bi-LSTM.

Secondly, we employ the self-attention mechanism [7] to focus on the key parts of each output of the first layer with Equation (2). The attention mechanism is an additional MLP, which determines which words should put more attention on than other words over the sentence.

$$\begin{aligned} \alpha_A &= \mathrm{softmax}(\tanh(A)\, W_A) \\ A_{att} &= \textstyle\sum_{i=1}^{m} \alpha_{Ai} * A_i \\ \alpha_B &= \mathrm{softmax}(\tanh(B)\, W_B) \\ B_{att} &= \textstyle\sum_{i=1}^{n} \alpha_{Bi} * B_i \end{aligned} \tag{2}$$

where α_A and α_B are attention weights, A_{att} and B_{att} are the attention outputs of the question pair, which are a weighted sum that combining each word.

Thirdly, we utilize the attention mechanism to select the more important parts of each for the output of the second layer with the following Equation (3). Different from the above, we directly use the self-attention output of the first layer.

$$\begin{aligned} \alpha_C &= \mathrm{softmax}(B_{att} C^T) \\ C_{att} &= \textstyle\sum_{i=1}^{m} \alpha_{Ci} * C_i \\ \alpha_D &= \mathrm{softmax}(A_{att} D^T) \\ D_{att} &= \textstyle\sum_{i=1}^{n} \alpha_{Di} * D_i \end{aligned} \tag{3}$$

where α_C and α_D are attention weights, C_{att} and D_{att} are the attention outputs.

Then, we concatenate the two attention outputs $q1_r = [A_{att};\ C_{att}]$ and $q2_r = [B_{att};\ D_{att}]$, and feed them to a nonlinear layer, so that we can get the deeper and more abstract semantic representation of the two sentences.

In order to represent more intuitively the similarity between the two questions, we calculate the difference: $q1_r - q2_r$ and elementwise product: $q1_r * q2_r$, which denote the distance and angle information respectively between the two questions [8].

Finally, we feed the question representations, distance and angle information to a multi-layer perceptron (MLP) layer for classification.

$$o = \text{MLP}([\, q1_r, q2_r, q1_r - q2_r, q1_r * q2_r]) \tag{4}$$

For **model-II**, what it differs from model-I is that we adopt the word-alignment attention to obtain the inter-relationship between each word of two questions. Equations (5-7) provide formal and specific details of this procedure.

$$e_{ij} = v^T \tanh\big(W\big[A_i; B_j\big] + b\big), i \in [1, m], j \in [1, n] \tag{5}$$

$$A_{align\ i} = \sum_{j=1}^{n} \frac{\exp(e_{ij})}{\sum_{k=1}^{n} \exp(e_{ik})} B_i,\ \ i \in [1, m] \tag{6}$$

$$B_{align\ j} = \sum_{i=1}^{m} \frac{\exp(e_{ij})}{\sum_{k=1}^{m} \exp(e_{kj})} A_j,\ \ j \in [1, n] \tag{7}$$

where e_{ij} is the attention score as the similarity between the words of the question pair, and A_{align} represents the extracted relevant information of B while B_{align} represents the extracted relevant information of A.

Then, we calculate the distance and angle information of the alignment output of the two questions respectively with A and B, as the input of the second Bi-LSTM layer.

$$\begin{aligned} C &= \text{BiLSTM}\big(\big[A, A_{align}, A - A_{align}, A *\ A_{align}\big]\big) \\ D &= \text{BiLSTM}\big(\big[B, B_{align}, B - B_{align}, B *\ B_{align}\big]\big) \end{aligned} \tag{8}$$

Finally, the self-attention is used to calculate the key parts of each question.

$$\begin{aligned} \alpha_C &= \text{softmax}(\tanh(C)\, W_C) \\ C_{att} &= \sum_{i=1}^{m} \alpha_{Ci} * C_i \\ \alpha_D &= \text{softmax}(\tanh(D)\, W_D) \\ D_{att} &= \sum_{i=1}^{n} \alpha_{Di} * D_i \end{aligned} \tag{9}$$

The latter part is the same as model-I.

3.3 Post-processing

In order to better utilize the characteristics of different models and make full use of the information provided by the training data, we use K-fold cross-validation to train each model, and then obtain the final outputs by voting.

4 Experiments

Datasets and evaluation metrics. The training set consists of 100,000 question pairs, which include labels and where the ratio of positive and negative samples is close to 1:1. The development set consists of 10,000 unlabeled question pairs. And the last test set consists of 110,000 unlabeled question pairs. Submissions are finally evaluated on the test set. Four metrics Precision, Recall, F1-score and Accuracy are used for the task.

Results and Analysis. Table 1 shows the experimental results on the development set, because the test set has only one submission opportunity. When we test and submit with the best training model, we find that the recall rate was significantly lower, as shown in the third and fourth rows in Table 1. This indicates that many positive examples in the data set cannot recognize. Thus, we use K-fold cross-validation method to get different experimental results, and post-process on them by voting, that is, as long as M models consider the question pair to be positive, then the question pair is marked as a positive example. Besides, we use the state-of-the-art ESIM model as our baseline system, and we train the system based on char-level and word-level inputs respectively. From Table 1, we find that the model with char-level input is better than the one with word-level input. Our two models are greater than ESIM, whose results have obvious improvement after 10-fold cross-validation.

Table 1. Results of various models.

	Precision	Recall	Accuracy	F1
ESIM(word)	0.8210	0.7366	0.7880	0.7765
ESIM(char)	0.8265	0.7714	0.8046	0.7980
Model-I	0.8477	0.7726	0.8169	0.8084
Model-II	0.8467	0.78	0.8194	0.8119
Model-I (10-fold)	0.8455	0.8448	0.8452	0.8451
Model-II (10-fold)	0.8304	0.8622	0.8431	0.8460
Model-I/II(10-fold)	0.8364	0.8670	0.8487	0.8514

5 Conclusion

This paper describes two semantic similarity models for the task of CCKS2018 QIM. Our solution is based on Siamese network with the attention factors without additional feature engineering. Finally, our system achieves fourth place in the test data with F1 0.84586.

Acknowledgments

The research work is supported by the National Natural Science Foundation of China (61572338) and the Natural Science Foundation of the Jiangsu Higher Education Institutions (Grant No. 16KJA520001).

References

1. Neculoiu P, Versteegh M, Rotaru M. Learning Text Similarity with Siamese Recurrent Networks[C]// Repl4nlp Workshop at ACL. 2016.

2. 陈二静, 姜恩波. 文本相似度计算方法研究综述[J]. 数据分析与知识发现, 2017, 1(6):1-11.
3. Conneau A, Kiela D, Schwenk H, et al. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data[J]. 2017:670-680.
4. Chen Q, Zhu X, Ling Z, et al. Enhanced LSTM for Natural Language Inference[J]. 2016:1657-1668.
5. OpenCC Homepage, https://github.com/BYVoid/OpenCC
6. Hochreiter S, Schmidhuber J. Long short-term memory.[J]. Neural Computation, 1997, 9(8):1735-1780.
7. Luong M T, Pham H, Manning C D. Effective Approaches to Attention-based Neural Machine Translation[J]. Computer Science, 2015.
8. Ghaeini R, Hasan S A, Datla V, et al. DR-BiLSTM: Dependent Reading Bidirectional LSTM for Natural Language Inference[J]. 2018.