ECNU at 2016 LiveQA Track: A Parameter Sharing Long Short Term Memory Model for Learning Question Similarity

Weijie An, Mengfei Shi, Xin Ouyang, Yan Yang, Qinmin Hu, and Liang He

Department of Computer Science and Technology, East China Normal University, Shanghai, 200062, China
{wjan, mfshi, yxou}@ica.stc.sh.cn, {yanyang, qmhu, lhe}@cs.ecnu.edu.cn

Abstract. In this paper, we present our system which is evaluated in the TREC 2016 LiveQA Challenge. Same as the last year, the TREC 2016 LiveQA track focuses on "live" question answering for the real-user questions from Yahoo! Answer. In this year, we first apply a parameter sharing Long Short Term Memory(LSTM) network to learn a high embedding of question representation. Then we combine the question representation with the key words information to strengthen the representation of semantic-similar questions, followed by calculating the question similarity with a simple metric function. Our approach outperforms the average score of all submitted runs.

1 Introduction

Open-domain question answering is a very important research topic in recent years. The TrecQA task in the previous years mainly dealt with the factual questions. Same as the task in 2015 [1], the LiveQA task in this year deals with the most recent questions submitted on the Yahoo! Answers[1] site that have not yet been answered by humans, and the participants are requested to respond in one minute with a limitation of length less than 1000.

The major challenges of this task are: 1)there is not a big enough corpus to answer any questions from diverse users. 2)the given question dose not directly share the lexical units with its semantic-similar question. 3)the question, especially the question body, contains a large amount of irrelevant information.

Therefore, we search in the community question answering site, such as the Yahoo! answers and Answers.com[2], to collect the semantic-similar question-answer pairs. The subsequent step is to measure the similarity of the original question and the candidate questions. Previous work on question answering task using

[1] answers.yahoo.com
[2] answwers.com

deep learning technologies achieved the state-of-the-art result [2, 4, 5]. These methods have a common purpose to learn the precise semantic representation of questions or answers. We apply a parameter sharing Long Short Term Memory(LSTM) network to learn a highly embedding of question representation, and we combine the question representation with the key words information to strengthen the semantic representation of question. Finally, we use a simple metric function to calculate the question-similarity. Our approach outperforms the average score of all submitted runs.

The rest of the paper is organized as follows: Section 2 describes the whole architecture of our system. Section 3 presents and analyzes the results. The conclusion are drawn in Section 4.

2 System Overview

Figure 1 shows the architecture of our system, which consists of three parts: Online Searcher, Similarity Metric and Answer Re-rank.

The Online Searcher searches the similarity question from specific community question and answering(CQA) site. Then we obtain the set of candidate question-answer pairs. The Similarity Metric measures the similarity of the original question and each candidate question obtained from previous step. The Answer Re-rank part combines the similarity of each question pairs with the external information such as the order in the source CQA site, and then responds the answer of the highest candidate question as the best answer. In this step, we have a hypothesis that the answer of the most similar question is able to better answer the original question than the others.

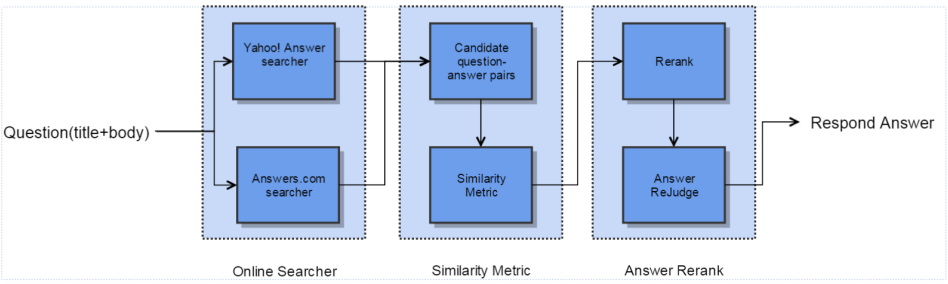

Fig. 1. The architecture of our system.

2.1 Online Searcher

Given each question $Q = \langle title, body, category, qid \rangle$, we first search the web resources to generate the candidate question-answer pairs set. In this step we want to get the similar question-answer pairs as much as possible. A high quality and comprehensive candidate set determines the answer quality. At the beginning, we have selected more than a dozen CQA sites. With considering the following reasons:

- **General answer quantity and quality**
- **Access to data easily**
- **Less time consuming**
- **Network stability**

We mainly construct two specific searchers, one for Yahoo! Answer and the other for Answers.com. Due to the different structure of the meta QA pair data in the two site, we define a common container for the QA pair. It mainly contains the following parts:

- **Titles**, the question titles
- **Body**, the question body, usually the description of question
- **Best Answer**, the best answer component in the Yahoo! answer question page, the only one answer in the answers.com question page
- **Source**, which site the question-answer pair from
- **Weight**, the original order in the search result list, more specifically, when we search a question, the site return a list of similar question, it is more like a location weight information

We do not pre-process the original question such as stemming and removing the stop words in this step. We find that if we remove the stop words in the question, the semantic features will be lost. For example, for the question "Do you feel Obama made the economy better or worse off?", if we remove the stop words "do", "you", "the", "or" and "off", it becomes "feel Obama economy better worse". It could be more useful for the key word matching, but meaningless for the similar question retrieval in the CQA site.

2.2 Similarity Metric

Given the original question $Q = \langle title, body, category, qid \rangle$ and the candidate QA pairs set $QA = \{C_1, C_2, ..., C_n\}$, $C_i = \{title, body, BestAnswer, Source, Weight\}$, we measure the similarity of Q_{title} and $C_{i-title}$ for each C_i in QA.

2.2.1 Parameter Sharing LSTM

Long Short Term Memory(LSTM): Recurrent Neural Networks(RNN) have been widely used in modeling the variable length sequence. In order to deal with the vanish of the gradient during the long distance transmission, [3] proposed the

LSTM model. Given an input sequence $x = \{x_1, x_2, ..., x_n\}$, at each time step the LSTM cell updates the hidden vector $h(t)$ using the following equations:

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\ \widetilde{c}_t &= tanh(W_c x_t + U_c h_{t-1} + b_c) \\ c_t &= i_t * \widetilde{c}_t + f_i * c_{t-1} \\ h_t &= o_t * tanh(c_t) \end{aligned} \tag{1}$$

LSTM uses these gate operation to control the flow of information through the cell. Usually the last hidden unit output can be regarded as the representation of the whole sentence.

Parameter Sharing LSTM(PS-LSTM): In our work, we build two LSTM networks, one for modeling the original question and the other for modeling the candidate question. We simply use a last pooling layer acting on the hidden output, which means we select the output of the last step as the representation of the question. When we get both representations, we calculate the similarity with the cosine metric. During the training step, we use the same parameters in both LSTM networks such as the weight matrices $W_i, W_f, W_o, W_c, U_i, U_f, U_o, U_c$ and the bias vectors $b_i, b_f, b_o, b_c,$. The model is shown in Figure 2.

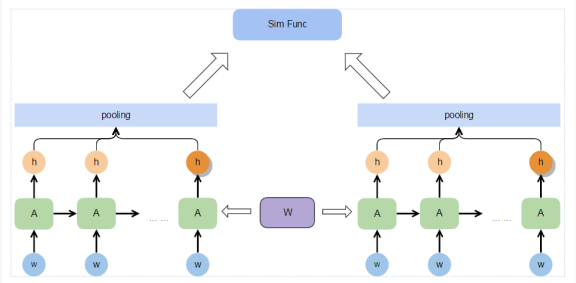

Fig. 2. An illustration of parameter sharing LSTM model

Suppose that h_l is the representation of the original question title(Q_{title}), h_r is the representation of a specific candidate question title($C_{i-title}$). We define the loss function with the maximum cross entropy as:

$$\begin{aligned} L &= -\frac{1}{n}\sum_x [log(p_x) * y + log(1 - px) * (1 - y)] \\ p_x &= 0.5 + 0.5 * cosine(h_l, h_r) \end{aligned} \tag{2}$$

where p_x is the predictive similarity, y is the real label in the training data, it

can either be 1 for similarity or 0 dissimilarity.

2.2.2 Combining the Key Word Information

As shown in [6], the key word information improves the answer selection result. The key word in the question characterized the main topics. Although two question shared almost same representation, they may focus on different topics, because of the shortcomings of word representation. For example, "Japan" and "China" are two words very close in embedding space, but when these two words appear in the question, they may focus on different place. In order to mitigate this weak point, we employ the BM25 retrieval model for calculating the key word matching score.

We build the index in both question title and body in order to capture more information. In the PS-LSTM model, we concern the similarity between the titles. So in this step we consider the information of question body and title to make up the deficiencies of the previous step.

2.3 Re-Ranking the results

After getting the candidate relevance score, we re-rank the candidate question-answer pairs by the Weight attributes and rejudge the answer of each C_i. Generally, according to the manual observation, we consider that the candidate question-answer pairs from Yahoo! Answer site are more valuable than Answers.com. Also we set a higher weight to the front question-answer pair according to the display order in the search result list. As mentioned in Section 2.1, the Source and Weight attributes characterize these impacts. Then we define a simple re-rank scoring function as shown in Equation 3. As we make a hypothesis that the answer of the most similar question is able to better answer the original question than the others, we judge the answer whether it is eligible. More specifically, we restrict the answer length less than 1000 and remove unreadable characters in the answer text.

$$score = (\alpha S_{PS-LSTM} + (1 - \alpha) S_{KWM}) * weight * source \tag{3}$$

where α is a tuned value. Because the PS-LSTM model predict a value $S_{PS-LSTM} \in [0, 1]$, we normalized the S_{KWM} score into $[0, 1]$ too.

3 Result and Evaluation

Network Setup We use the pre-trained word embedding released from the glove project[3] for projecting the word into 300 dimensions space. And the network parameters are randomly initialized using a Gaussian distribution($\mu = 0$, $\sigma = 0.2$).

[3] http://nlp.stanford.edu/projects/glove/

The hidden layer size is tuned to 128. We train our model using a batch size 512, and the maximum sentence length is set to 40. In order to adapt different sentence lengths in each batch, we use the mask strategy in each batch train step.

Evaluation Metric For the final test of Live QA Track, the results are judged by TREC editors firstly using the 4-level scale as follows:

- **4: Excellent** a significant amount of useful information, fully answers the question
- **3: Good** partially answers the question
- **2: Fair** marginally useful information
- **1: Bad** contains no useful information for the question
- **-2:** the answer is unreadable

The performance measures are:

- **avg-score(0-3)** average score over all queries (transferring 1-4 level scores to 0-3, hence comparing 1-level score with no-answer score, also considering -2-level score as 0)
- **succ@i+** number of questions with i+ score (i=2..4) divided by number of all questions
- **prec@i+** number of questions with i+ score (i=2..4) divided by number of answered only questions

Table 1 shows the official result of our system and the average score of all runs in Trec LiveQA 2016 track. Our result outperform the average score in the 4 metric aspects.

Table 1. Official TREC 2016 LiveQA track evaluation results.

RunID	Answers	avgScore(0-3)	succ@2+	succ@3+	succ@4+
ECNU-ECNU	834	0.8365	0.4108	0.2906	0.1350
AVG_SCORE	771.0385	0.5766	0.3042	0.1898	0.0856

4 Conclusion

This paper describes our system architecture and three key components which are evaluated in the TREC 2016 LiveQA Challenge. We apply a PS-LSTM to learn a high embedding of question representation. Also we combine the key word information to enhance the semantic similarity matching. Finally, we define a simple score function that combines some manual discovered weight information.

In the future, we will focus on the question representation with the attention method. Also we will continue on the answer selection and auto-generation methods, such as merging the candidate answers.

Acknowledgment

This research is funded by the National Natural Science Foundation of China (No. 61602179) and the Science and Technology Commission of Shanghai Municipality (No.15PJ1401700).

References

1. Eugene Agichtein, David Carmel, Donna Harman, Dan Pelleg, and Yuval Pinter. Overview of the trec 2015 liveqa track. In *The Twenty-Fourth Text REtrieval Conference (TREC 2015) Proceedings. National Institute of Standards and Technology (NIST)*, 2015.
2. Cicero dos Santos, Luciano Barbosa, Dasha Bogdanova, and Bianca Zadrozny. Learning hybrid representations to retrieve semantically equivalent questions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 694–699, Beijing, China, July 2015. Association for Computational Linguistics.
3. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
4. Tao Lei, Hrishikesh Joshi, Regina Barzilay, Tommi Jaakkola, Kateryna Tymoshenko, Alessandro Moschitti, and Lluís Màrquez. Semi-supervised question retrieval with gated convolutions. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1279–1289, San Diego, California, June 2016. Association for Computational Linguistics.
5. Ming Tan, Bing Xiang, and Bowen Zhou. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*, 2015.
6. Di Wang and Eric Nyberg. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 707–712, Beijing, China, July 2015. Association for Computational Linguistics.