ECNU at SemEval-2016 Task 3: Exploring Traditional Method and Deep Learning Method for Question Retrieval and Answer Ranking in Community Question Answering

Guoshun Wu[1], **Man Lan**[1,2*]
[1]Department of Computer Science and Technology,
East China Normal University, Shanghai, P.R.China
[2]Shanghai Key Laboratory of Multidimensional Information Processing
51141201064@ecnu.cn, mlan@cs.ecnu.edu.cn*

Abstract

This paper describes the system we submitted to the task 3 (Community Question Answering) in SemEval 2016, which contains three subtasks, i.e., Question-Comment Similarity (subtask A), Question-Question Similarity (subtask B), and Question-External Comment Similarity (subtask C). For subtask A, we employed three different methods to rank question-comment pair, i.e., supervised model using traditional features, Convolutional Neural Network and Long-Short Term Memory Network. For subtask B, we proposed two novel methods to improve semantic similarity estimation between question-question pair by integrating the rank information of question-comment pair. For subtask C, we implemented a two-step strategy to select out the similar questions and filter the unrelated comments with respect to the original question.

1 Introduction

The purpose of Community Question Answering task in SemEval 2016 (Nakov et al., 2016) is to provide a platform for finding good answers to new questions in a community-created discussion forum, where the main task (subtask C) is defined as follows: given a new question and a large collection of question-comment threads created by a user community, participants are required to rank the comments that are most useful for answering the new question. Obviously, this main task consists of two optional subtasks, i.e., Question-Comment Similarity (subtask A, also known as *answer ranking*), which is to re-rank comments/answers according to their relevance with respect to the question, and Question-Question Similarity (i.e., subtask B, also known as *question retrieval*), which is to retrieve the similar questions according to their semantic similarity with respect to the original question.

To address subtask A, we explored a traditional machine learning method which uses multiple types of features, e.g., Word Match Features, Translation-based Features, and Lexical Semantic Similarity Features. Additionally, for subtask A, we also built a Convolutional Neural Network (CNN) model and a bidirectional Long Short-Term Memory (BLSTM) model to learn joint representation for question-comment (Q-C) pair. For subtask B, besides IR method and traditional machine learning method, we also proposed two novel methods to improve semantic similarity estimation between question-question (Q-Q) pairs by integrating the rank information of Q-C pairs. Since subtask C can be regarded as a joint work of the two above-mentioned subtasks, we implemented a two-step strategy to first select out similar questions and then to filter out the unrelated comments with respect to the original question.

The rest of this paper is organized as follows. Section 2 describes our system. Section 3 describes experimental setting. Section 4 and 5 report results on training and test sets. Finally, Section 6 concludes this work.

2 System Description

For subtask A, we presented three different methods i.e., using traditional linguistic features, learning a CNN model and a bidirectional LSTM model to represent question and comment sentences. For subtask B, besides traditional methods, we proposed two novel methods to improve semantic similarity estimation between Q-Q pairs by integrating the rank information of Q-C pairs. The first is to adopt general ranking evaluation metrics of Q_0-C and Q_1-C (i.e., Spearman, Pearson, and Kendall Coefficient) as additional ranking scores or features of Q_0-Q_1 where Q_0 and Q_1 represent original question and its related question, respectively. The second is to extract features on Q_0-C and Q_1-C and to regard the cosine values which are calculated on these two feature vectors as additional features for Q_0-Q_1.

2.1 Features Engineering

All three subtasks can be regarded as an estimation task of sentence semantic measures which can be modeled by various types of features. In this work, we employed the following four types of features borrowed from previous work, i.e., Word Match Features, Translation Based Features, Topic Model Based Features, and Lexical Semantic Similarity Features. The details of these four types of features are described as follows. Note that the following four feature types are adopted in both Q-Q and Q-C pairs, here we took the Q-Q pair for example.

Word Matching Feature (WM): This feature records the proportions of co-occurred words between a given sentence pair. Given a Q-Q pair, this feature type is calculated using five measures: $|Q_0 \cap Q_1|, |Q_0 \cup Q_1|/|Q_0|, |Q_0 \cap Q_1|/|Q_1|, |Q_1 - Q_0|/|Q_1|, |Q_0 - Q_1|/|Q_0|$, where $|Q_0|$ and $|Q_1|$ denote the number of the words of Q_0 and Q_1.

Translation Based Feature (TB): The above WM feature only considers the overlapped words between Q_0 and Q_1 and thus it may fail to "bridge the lexical gap" between Q-Q pair. One possible solution is to regard this task as a statistic machine translation problem between question and answer by using the IBM Model 1(Brown et al., 1993) to learn the word-to-word probabilities. Following (Xue et al., 2008; Surdeanu et al., 2011), we regarded $P(Q_0|Q_1)$, i.e., the translation probability of Q_1 when given Q_0, as a translation based feature. The probabilities are calculated as:

$$P(Q_0|Q_1) = \prod_{w \in Q_0} P(w|Q_1)$$

$$P(w|Q_1) = (1-\lambda)P_{tr}(w|Q_1) + \lambda P_{ml}(w|C)$$

$$P_{ml}(w|Q_1) = \sum_{a \in Q_1} P(w|a) P_{ml}(a|Q_1)$$

where $P(Q_0|Q_1)$ is the probability that the Q_0 word w is generated from Q_1, λ is a smoothing parameter, C is a background collection. $P_{ml}(w|C)$ is computed by maximum likelihood estimator. $P(w|a)$ denotes the translation probability from Q_1 word a to Q_0 word w. The GIZA++ Toolkit[1] is used to compute these probabilities.

Topic Model Based Feature (TMB): We used the LDA (Blei et al., 2003) model to transform Q_0 and Q_1 into topic-based vectors and then took the *cosine* value of two topic vectors as feature. We use the *GibbsLDA++* (Phan and Nguyen, 2007) Toolkit to train the topic model.

Lexical Semantic Similarity Feature (LSS): Inspired by (Yih et al., 2013), we included the lexical semantic similarity features in our model. We used three different word vectors to represent LSS feature, i.e., the 300-dimensional version of word2vec (Mikolov et al., 2013) vectors, 300-dimensional Glove vectors (Pennington et al., 2014) and 300-dimensional vectors which are pre-trained with the unsupervised neural language model (Mikolov et al., 2013) on the Qatar Living data [2]. Words not present in the set of pre-trained words are initialized randomly. There are two ways to calculate the LSS features. One is to calculate the cosine similarity by summing up all word vectors in Q_0 and Q_1. Another is to adopt averaged pairwise cosine similarity between each word in Q_0 and Q_1.

Besides above four types of features, for Q-Q pair, we also extracted following two question information features (**QI**) to describe the informativeness of related question Q_1: (1) the number of words in Q_1 (2) the position of Q_1 in all related questions. For Q-C pair, we also extracted following two comment information features (**CI**) to measure the informativeness of a comment text: (1) the number of words in comment (2) the number of nouns, verbs and adjectives in comment.

2.2 Two Methods to address subtask A

2.2.1 Method 1: CNN

We proposed a convolutional neural network to model question-comment sentence. As illustrated in

[1]http://www.statmt.org/moses/giza/GIZA++.html

[2]http://alt.qcri.org/semeval2015/task3/index.php?id=data-and-tools

Figure 1, we first input word embeddings (here we used 300-dimensional Glove vectors in (Pennington et al., 2014)) of question and comment words and then learn the meaning (i.e., feature vector) of question and comment through convolution and pooling operation. After a simple concatenation layer connecting question and comment vectors, we final obtain a relevant score through a softmax operation.

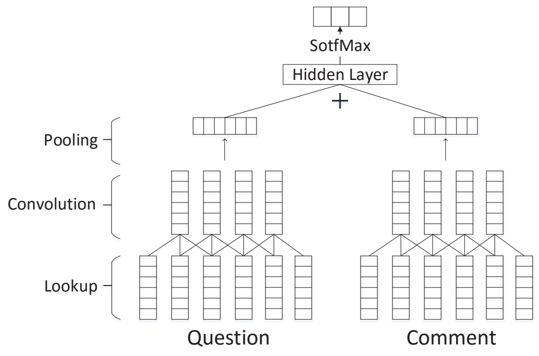

Figure 1: An illustration of CNN for question-comment similarity estimation.

2.2.2 Method 2: BLSTM

Figure 2 shows a multiple-layer BLSTM network model we used for question and comment sentences modeling. The procedure of BLSTM is similar to that of CNN. The words of question and comment sentences are first converted into vectors by looking up publicly available 300-dimensional Glove vectors. Then they are sequentially read by BLSTM from both directions. In this way, the contextual information across words in both question and comment sentences is modeled by employing temporal recurrence in BLSTM. Like CNN, finally it outputs a relevant score between question and comment by a simple concatenation operation on these two output vectors and a softmax operation.

2.3 Two Methods for subtask B

To calculate semantic similarity between Q_0 and Q_1 pair, previous work extracted features only from the Q-Q sentence pair. We stated that the comment set C and its rank with respect to Q_1 also provide useful information for question-question similarity. To address it, we propose two novel methods to improve

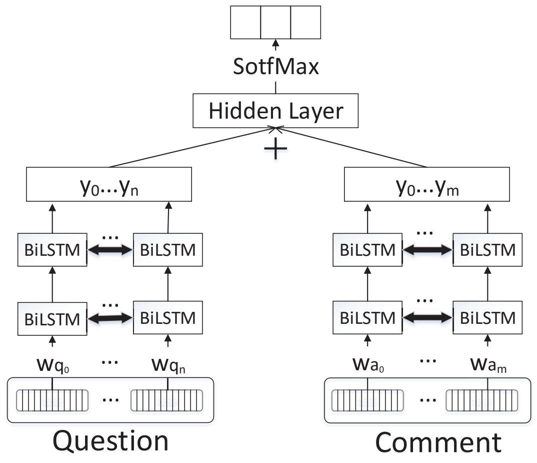

Figure 2: An illustration of BLSTM model for question-comment similarity estimation.

semantic similarity estimation between Q-Q pair by integrating the rank information of Q-C pair.

2.3.1 Method 1: adopt Q-C Ranking Evaluation Metrics as Similarity Score

The first method is to adopt rank evaluation metrics, i.e., *Spearman*, *Pearson*, and *Kendall* Ranking Coefficient directly as similarity scores for question similarity estimation.

Generally, these three nonparametric metrics are to measure the statistical dependence between two variables and to assess how similar between two variables. In comment ranking, they are used to measure how similar the two rankings Q_0-C and Q_1-C are. Based on our consideration, given one comment set C, if the two ranks of Q_0-C and Q_1-C are similar, the semantic similarity between Q_0-Q_1 is high. These three ranking correlation coefficients (i.e., *Spearman*, *Pearson*, and *Kendall* Coefficient) can be used directly as question similarity scores or used as additional ranking scores in combination with other features (described in Section 2.1) extracted from Q_0-Q_1 pair.

2.3.2 Method 2: add new features extracted from Q-C pair

We presented two methods to add new features extracted from Q-C pair. The first is to extract the features from Q_0-C and Q_1-C pair and then use the *cosine* scores calculated on the two feature vectors as additional features for Q_0-Q_1. We extracted traditional NLP features described in Section 2.1 from Q_0-C and Q_1-C pairs, respectively, denoted as two feature vectors, i.e., F_0 and F_1. Then we calculated the *cosine* similarity on these two vectors respectively, and obtain two *cosine* scores, i.e., $cos(Q_0\text{-}C)$, and $cos(Q_1\text{-}C)$. After that, we calculated the absolute difference between these two *cosine* scores. Finally, the obtained scores (denoted as $[cos_1, cos_2, ...]$) are ranked as additional features.

The second is to calculate the ranking scores of Q_0-C and Q_1-C by using comment ranking model firstly, then use the *Manhattan Distance* of two lists of ranked scores as an additional feature.

2.4 A Two-Step Filtering for Subtask C

To overcome the error propagation from question-question similarity step to question-comment similarity step, we employed a two-step filtering strategy for subtask C. The first step is to choose the top *N* similar questions with the aid of the Q-C ranking. The second step is to re-rank the comment and choose the top *M* comments with integration of the previous Q-Q results.

3 Experimental Setting

3.1 Datasets

Table 1 shows the statistics of training, development and test data sets, where the #_original, #_related, and #_answers represent the number of original questions, related questions and answers, respectively. The types of comments with respect to original question and related question fall into three classes: $Good$, $PotentiallyUseful$ and Bad. The types of related question with respect to original question fall into three classes: $PerfectMatch$, $Relevant$ and $Irrelevant$.

Subtask	Data	#_original	#_related	#_answers
A	train	–	5,898	37,848
	dev	–	500	5,000
	test	–	327	3,270
B	train	267	2,669	26,690
	dev	50	500	5,000
	test	70	700	7,000
C	train	267	2,669	26,690
	dev	50	500	5,000
	test	70	700	7,000

Table 1: Statistics of datasets.

3.2 Preprocessing

We first removed stop words and punctuation, and changed all words to lowercase. After that, we performed tokenization and stemming using NLTK[3] Toolkit.

3.3 Evaluation Metrics

To evaluate performance of the tasks, the $Mean$ $Average\ Precision$ (MAP) is adopted as official evaluation measure by the organizers which the MAP is defined as the mean of the averaged precision scores for queries.

3.4 Learning Algorithm

We compared two ranking strategies in traditional method. One is to train a pairwise-based ranking model, i.e., *Learning-to-rank* (Trotman, 2005), and use the output of model as a ranking score directly. Another is to first train a supervised classification model and then use the confidence score of probability as a ranking score. To train a supervised classifier, two algorithms implemented in SKLearn[4] have been examined, i.e., Logistic Regression (LR) and Support Vector Machine (SVM). Finally, Logistic Regression classifier (penalized argument c = 1) is adopted for all three subtasks for its good performance in preliminary experiments.

4 Experiments on Training Data

4.1 Results on Subtask A

For the experiments of subtask A, the hyper-parameters of CNN model are set as follows: the number of filter windows is 2, feature maps are set to 100, learning rate is set to 0.01. And the hyper-parameters of BLSTM model are set as follows: memory size is set to 500 and the learning rate is 0.01. Table 2 shows the results of subtask A with three different methods.

Firstly, all CI, TB, TMB, and LSS features significantly over WM baseline. Since CI is a measure of the informativeness of comment text, this indicates that users trend to choose the comment with more information. TB can learn word alignment between different words. Unlike the surface word matching features which only consider the surface word, the

[3]http://www.nltk.org/

[4]http://scikit-learn.org/stable/

Methods	Features	MAP(%)
Traditional NLP Features	WM	57.13
	.+TB	58.91
	.+TMB	61.37
	.+CI	63.03
	.+LSS	**65.37**
CNN	–	65.04
BLSTM	–	65.13
Tra + CNN + BLSTM	–	**66.84**

Table 2: Results of subtask A using different methods. ".+" means to add current feature to the previous feature set.

LSS features are obtained by integrating context of the word. Therefore, the LSS features show that this particular word embedding seems to complement the surface word matching information. Secondly, the combination of five types of features achieve the best performance for traditional method. Thirdly, the model based CNN and BLSTM achieve comparable performance with traditional method. Finally, the combination of three methods achieve the best performance which shows that CNN and BLSTM catch complementary information for Q-C pair with traditional method.

4.2 Results on Subtask B

Table 3 summarizes the results of subtask B with NLP features and integrating the rank information of Q-C pair. Here *Lucene* represents using Lucene

Method	Features	MAP(%)
Lucene	BM25	69.95
Traditional NLP Features	WM	69.91
	.+TB	70.72
	.+TBM	71.05
	.+LSS	72.13
	.+QI	**74.03**
Method 1	*Pearson*	61.18
	Spearman	62.86
	Kendall	62.95
	Pearson + NLP	68.15
	Spearman + NLP	68.49
	Kendall + NLP	**68.95**
Method 2	NLP	74.03
	.+ARC	74.25
	.+ARR	**75.04**

Table 3: Results of subtask B using different methods.

Toolkit[5] with the original question as query with BM25 ($K_1 = 1.2$ and $B = 0.75$). *ARC* and *ARR* are the first and second methods presented in Section 2.3.2. According to the results of Table 3, we can make following three observations:

(1) Traditional NLP features significantly improve the performance of question-question similarity over *Lucene* baseline.

(2) The *Pearson*, *Spearman*, and *Kendall* get similar performance and do not perform well versus traditional NLP method. The three rank correlations all take down the performance of traditional NLP method when combined with it. The possible reason is that the ranked scores of comments are obtained by pre-trained comment ranking model which has a limitation of performance.

(3) Both *ARC* and *ARR* make contributions to the performance which means combining the information of Q-C pair is helpful to find related questions.

4.3 Results on Subtask C

Table 4 depicts the results on subtask C, where *WMQ*, *TMBQ* and *TBQ* represent extracting word matching, topic model based and translation based features on original question and related question. From Table 4, we observe the similar results with those in subtask A, i.e., traditional features make contribution to comment ranking. Moreover, the performance is improved by adding features extracted from Q-Q pair, which indicates that the information extracted from Q-Q pair makes significant contribution to answer ranking subtask.

Method	Features	MAP(%)
Lucene	BM25	24.59
Traditional NLP Features	WM	30.15
	.+CI	33.13
	.+TB	34.27
	.+TMB	35.80
	.+LSS	**36.82**
Q-Q Features	NLP	36.82
	.+WMQ	38.14
	.+TMBQ	38.51
	.+TBQ	**39.39**

Table 4: Results of subtask C with different traditional NLP features.

However, above results are evaluated using MAP on top 10 comments. Therefore the errors introduced in question retrieval (subtask B) would be prorogated to answer ranking (subtask A) and finally reduce the whole performance of CQA (subtask C).

[5]https://lucene.apache.org/

To solve this problem, we investigated a two-step method to first filter unrelated comments and then filter unrelated answers. Figure 3 shows the results of two filtering methods on MAP metric, where N represents the number of top related questions and M represents the number of top ranked answers.

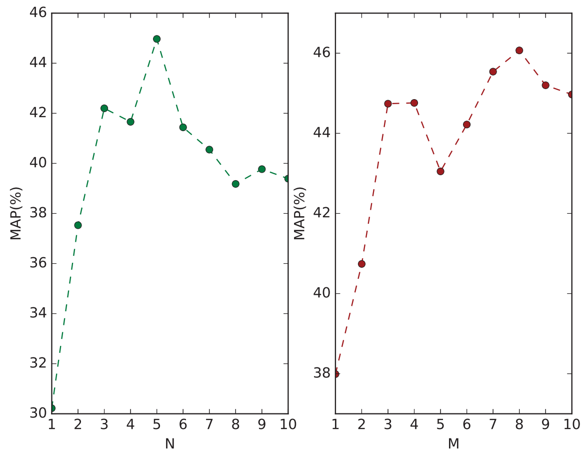

Figure 3: The results of subtask C using two-step filtering operation.

From left subplot of the Figure 3, we see that the best performance with filtering operation are much higher than the best score ($MAP = 39.39\%$) without any filtering. The best performance 44.97% is obtained when $N = 5$ and $M = 10$. The reason may be that the filtering operation of unrelated questions can take away many unrelated comments for original question. The right subplot of the Figure 3 shows the performance curve ($N = 5$) when increasing the values of M. Similarly, the performance increases with M increasing from 7 to 9 and it achieves the best score of 46.07% when $N = 5$ and $M = 8$.

4.4 System Configuration

Based on above experimental analysis, the three system configurations are as followings:

(1) subtask A: We used the combination of traditional method, CNN and BLSTM as the primary run in the test set. Traditional method and BLSTM serve as contrastive1 run and contrastive2 run.

(2) subtask B: Traditional method with Method 2 is used as primary run in the test set. The combination of traditional method with Method 2 and *Lucene* is contrastive1 run and traditional method alone is contrastive2 run.

(3) subtask C: The two-step filtering operation with $N = 5$ and $M = 8$ serves as primary run in the test set. The two-step filtering operation with $N = 4$ and $M = 7$ is contrastive1 run. Traditional features adding Q-Q pair information is used as contrastive2 run.

5 Results on Test Data

Table 5 shows the results on test set which are released by the organizers.

subtask	run(rank)	MAP(%)
A	ECNU-primary(4)	77.28
	ECNU-contrastive1	71.34
	ECNU-contrastive2	75.71
	Kelp-primary(1)	79.19
B	ECNU-primary(7)	73.92
	ECNU-contrastive1	73.25
	ECNU-contrastive2	71.62
	UH-PRHLT-primary(1)	76.70
C	ECNU-primary(7)	46.47
	ECNU-contrastive1	48.49
	ECNU-contrastive2	47.24
	SUper team-primary(1)	55.41

Table 5: Our results and the best results on three subtask test sets.

From the results, we find: (1) In subtask A, the combination of three methods significantly improve the performance over the traditional method and BLSTM, which is consistent with the results on training data as our expectation. (2) In subtask B, the result using traditional features is higher than *Lucene* but still has a certain gap with the best result. The possible reason may be because several traditional features do not work well in the test set. (3) In subtask C, beyond our expectation, the method using two-step filtering operation does not make obvious contribution. The possible reason may be that the values of M and N are not suitable for test set.

6 Conclusion

In this paper, we proposed multiple strategies (i.e., traditional method of extracting features and deep learning models) to address Community Question Answering task in SemEval 2016. For subtask A, we trained a classifier and learned the question-comment representation based CNN and BLSTM. The combination of three models obtains the best results. For subtask B, we proposed two novel methods to improve semantic similarity estimation between Q-Q pairs by utilizing the information of Q-C ranking. For subtask C, we employed a two step filtering strategy to reduce the noise which taking from unrelated comments. The results on test set show the effectiveness of our methods.

Acknowledgments

This research is supported by grants from Science and Technology Commission of Shanghai Municipality (14DZ2260800 and 15ZR1410700), Shanghai Collaborative Innovation Center of Trustworthy Software for Internet of Things (ZF1213).

References

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Comput. Linguist.*, 19(2):263–311, June.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Preslav Nakov, Lluís Màrquez, Walid Magdy, Alessandro Moschitti, Jim Glass, and Bilal Randeree. 2016. Semeval-2016 task 3: Community question answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval 2016)*, Berlin, Germany, August. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1532–1543.

Xuan-Hieu Phan and Cam-Tu Nguyen. 2007. Gibbslda++: Ac/c++ implementation of latent dirichlet allocation (lda).

Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2011. Learning to rank answers to non-factoid questions from web collections. *Computational Linguistics*, 37(2).

Andrew Trotman. 2005. Learning to rank. *Information Retrieval*, 8(3):359–381.

Xiaobing Xue, Jiwoon Jeon, and W. Bruce Croft. 2008. Retrieval models for question and answer archives. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 475–482, New York, NY, USA. ACM.

Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1744–1753, Sofia, Bulgaria, August. Association for Computational Linguistics.