ITNLP-AiKF at SemEval-2016 Task 3: a question answering system using community QA repository

Chang'e Jia[1], Xinkai Du[2], Chengjie Sun[1] and Lei Lin[1]
[1]Harbin Institute of Technology, Harbin Heilongjiang 150001, China
[2]Beijing Huilan Technology Co.,Ltd.
{cejia, cjsun, linl}@insun.hit.edu.cn
duxk@huilan.com

Abstract

Community Question Answering (CAQ) systems play an important role in people's lives due to the huge knowledge accumulated in them. In order to take full advantage of the huge knowledge, the target of semeval2016 task3 is to find the best answers to a new question in CQA. This work proposes to use rich semantic text similarity (STS) features to complete the task. We address the task as a ranking problem and Support Vector Regression (SVR) model is chosen to combine rich semantic similarity features and context features. Finally, we used genetic algorithm to do feature selection. Our method achieves an MAP (mean average precision) of 71.52%, 71.43% and 48.49% in subtask A, B and C respectively. It ranked 8th in subtask A and subtask B, and 7th in subtask C.

1 Introduction

The CQA system with interactive and open character, can better adapt to the diversity of needs of users. With the growth of the number of users, community question answering system has accumulated a lot of QA pair archives. It has presented new challenges to analyze user's requirement and recommends high-quality answers to users.

In response to this problem, Semeval2015 Task3 - "Answer Selection in Community Question Answering"[1] (Nakov et al., 2015) proposed a task to divide the answers into three levels in accordance with the relevance of the question. However, the classification system does not fully comply with the question requirement, as it does not implement the recommending function.

Semeval2016 Task3 - "Community Question Answering"[2] (Nakov et al., 2016) puts forward new requirements to automate the process of finding good answers to new questions in a community-created discussion forum based on the Semeval2015 Task3. The task is divided into three parts: subtask A - "Question-Comment Similarity", subtask B - "Question-Question Similarity" and subtask C - "Question-External Comment Similarity".

In our work, we focus on using features that employ STS knowledge, such as extracting text similarity features from word vectors, structured resource and topic models, to deal with the task. Word vectors has been used in (Liu, Sun, Lin, Zhao, & Wang, 2015) and (Nicosia et al., 2015) to compute STS, and (Jin, Sun, Lin, & Wang, 2014) has evaluated word-phrase semantic similarity with structured resource.

2 Feature

The main idea of our method is to find the similarity between most similar words in two sentences to estimate sentence similarity. Our features include the following categories: WordNet-based features, vector features, word matching features, topic features and answer features.

[1] http://alt.qcri.org/semeval2015/task3/

[2] http://alt.qcri.org/semeval2016/task3/

Proceedings of SemEval-2016, pages 829–834,
San Diego, California, June 16-17, 2016. ©2016 Association for Computational Linguistics

2.1 Vector Features

There are three approaches that we are applying to measure sentence similarity with word vector.

The first one uses the sum of all the words' vectors in sentence *s* as the representative of *s*, and calculate the distance of two sentences' vector.

$$vec(s) = \sum_{w_i \in s} vec(w_i) \tag{1}$$

$$sim(s_1, s_2) = c\_sim(vec(s_1), vec(s_2)) \tag{2}$$

Where *s* is a sentence, *vec(w)* is the vector of word *w*,and *c_sim(v,u)* is cosine similarity which will be mentioned below.

$$c\_sim(v,u) = \frac{\sum_i^n v_i \times u_i}{\sqrt{\sum_i^n (v_i)^2} \times \sqrt{\sum_i^n (u_i)^2}} \tag{3}$$

Where *v* and *u* are two *N-dimensional* vectors. v_i is the *i-th* element of *v*.

The second and the third are similar to each other. The procedure that computing sentence pair similarity includes the following three steps.

First, given two sentences s_1 and s_2, and for each word *v* in sentence s_1, we find the most similarity word *u* in sentence s_2, to word *v*. And we do the same to sentence s_2.

$$sc(v, u \in s_1) = \max_{u \in s_2}(sim(v,u)) \tag{4}$$

Second, we calculate the similarity of a sentence-sentence pair based on each sentence respectively:

$$sim(s) = \frac{\sum_{w \in s} sc(w)}{l} \tag{5}$$

$$idfsim(s) = \frac{\sum_{w \in s} \text{sc}(w) \times idf(w)}{\sum_{w \in s} idf(w)} \tag{6}$$

Where *l* is the number of the words with stopwords removed from sentence *s*, and *idf(w)* is the inverse document frequency (Sparck Jones, 1972) of word *w* in the Wikipedia data.

Third, the value is averaged over the two sentence:

$$sim\_ag(s_1, s_2) = \frac{sim(s_1) + sim(s_2)}{2} \tag{7}$$

$$idfsim\_ag(s_1, s_2) = \frac{idfsim(s_1) + idfsim(s_2)}{2} \tag{8}$$

We trained two word2vec[3] models using Gensim toolkit[4] (Řehůřek & Sojka, 2010). The first one is trained on the training data, and the second one on Wikipedia data[5]. Only these latter two ways are used in both models.

In addition, we also make use of existing word vectors mentioned by earlier researchers (Nicosia et al., 2015), Glove[6] (Pennington, Socher, & Manning, 2014) and COMPOSES[7] (Baroni, Dinu, & Kruszewski, 2014), which have been proved to be helpful in many NLP applications.

2.2 WordNet-based Features

WordNet (Fellbaum, 2005) is widely used in semantic similarity computing in the field of natural language processing. WordNet provides six ways to calculate the similarity of words depending on the meanings: path-similarity (Resnik, 1999), Leacock-Chodorow Similarity (Leacock & Chodorow, 1998), Wu-Palmer Similarity (Wu & Palmer, 1994), Resnik Similarity (Resnik, 1995), Jiang-Conrath Similarity (Jiang & Conrath, 1997) and Lin Similarity (Lin, 1998).

In our systems, the six methods are all used to measure word similarity. The WordNet-based features are computed using the same formulas as the last two methods of vector features.

2.3 Word Match Features

Longest Common Subsequence (Allison & Dix, 1986) can retain the words' position information when computing the sentence similarity:

$$sim(s_1, s_2) = \frac{\frac{lcs(s_1, s_2)}{l_1} + \frac{lcs(s_1, s_2)}{l_2}}{2} \tag{9}$$

Where *lcs*(s_1,s_2) is the length of the longest common subsequence, l_1 and l_2 are the numbers of the words in s_1 and s_2.

We also use the bag of word to search the hidden relationship between words and sentences,

[3] The size is 400.

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

[5] https://dumps.wikimedia.org

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

[7] http://clic.cimec.unitn.it/composes/semantic-vectors.html

and the cosine similarity is used to be the measure of vector similarity.

Besides, we use Stanford CoreNLP toolkit (Finkel, Grenager, & Manning, 2005) to get the two sentences' nouns and measure their similarity by bag of word.

2.4 Topic Features

All the features mentioned above are based on lexical similarity. In order to overcome the limitation of the lexical features, we build Latent Dirichlet Allocation (Blei, Ng, & Jordan, 2003) model and Latent Semantic Analysis (Hofmann, 2001) model using the Gensim toolkit (Řehůřek & Sojka, 2010), which are both trained on Wikipedia data.

The topic models[8] can get sentence vector directly, and we calculate the vector distance by cosine similarity.

2.5 Answer Features

Closely analyzing the train data, we noticed that many "Good" comments would like to suggest questioners to visit a web site or ask further questions by email, and many "Good" comments prefer to contain pictures or numbers to explain themselves more clearly. Moreover, "Good" comments' sentence length is much longer than "PotentiallyUserful" comments and "Bad" comments'.

In addition, respondents themselves have a great influence on the quality of the answers. It may lead to a "Bad" comment if the respondent is also the questioner, and if the respondent is not the questioner but asks a question, it may also lead to a "Bad" comment. If a respondent was accustomed to submit high-quality comments, he/she has a high likelihood of offering a "Good" suggestion in the current question. So, we have voted the accuracy and error rates of comments for all users.

The answer features are only applied in subtask A.

3 Method and Result

3.1 Feature generation

Each question has brief description and detailed description. Take the following question as an example:

OrgQSubject: *What is the purpose of heaven?*

OrgQBody: *What is the point? What is in it for the ones that get there? Let's leave the purpose of hell for another thread. I invite you to ponder. You can quote scripture or Sura's etc if you want but you must expand upon them with your own thoughts.*

As we can see, people can get a broad understanding on the question by reading the brief description, and experiments show that the features of brief description lead to a better result.

Features	F1 score
Sub	0.4898
Body	0.4476
Sub+Body	0.5316

Table 1: Experiments for subtask B. A classification model is trained on training dataset and tested on development dataset.

We assume that if the features come off well on a classification model, they would do a good job on ranking model.

So, we extracted eigenvalue from both brief description and detailed description for subtask B. The subtask A is trained models with all the characteristics mentioned above. We multiply the subtask A's results by the subtask B's as subtask C's. And eventually we got 38 features for subtask A and 56 features for subtask B. Table 2 lists all the features.

3.2 Feature Selection

Considering that there may be a feature subset performing better than other subset of all features, we designed a genetic algorithm (Renna, 2000) to find the best one. The genetic algorithm (GA) can be described as follows:

Encoding: Assuming that there are *n* features, n-bit binary will be needed to encode a chromosome then. The process of feature selection is as the Figure 1 shows.

[8] The size of both models are 100.

	Category	Feature
Subtask A and Subtask B	Vector features	W2V_wiki, W2V_wiki_idf, W2V_qatar, W2V_qatar_idf, Glove_sp, Glove_w2w, Glove_w2w_idf, COMPOSES_sp, COMPOSES_w2w, COMPOSES_w2w_idf
	WordNet-based features	PATH_sim, LCH_sim, WUP_sim, RES_sim, JCN_sim, LIN_sim, PATH_sim_idf, LCH_sim_idf, WUP_sim_idf, RES_sim_idf, JCN_sim_idf, LIN_sim_idf
	Topic features	LDA_sim, LSA_sim
	Word match features	LCS_sim, BagOfW_sim, NOUN_sim, NOUN_sim_idf
Subtask A	Answer features	IS_QUsr, IS_Thank, IS_Ask, IS_Other_Ask, IS_Email, IS_URL, U_BestRate, U_GoodRate, U_BadRate, Sen_Lenght, IS_NUM, IS_IMG,

Table 2: The features we extracted

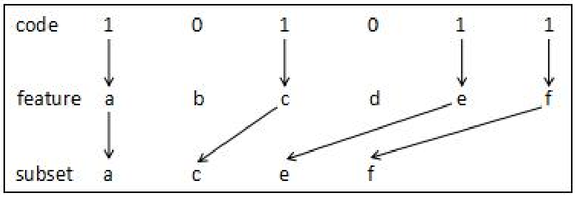

Figure 1: Feature subset selection. If the *i-th* feature is added into the subset, the value of the *i-th* binary is 1, and if not is 0.

Individual creation: Relying on the hypothesis that a feature can make a feature subset work better if it is added to the current feature subset, we increase the probability[9] that each feature is selected.
Fitness: We employ SVR as the evaluation function of feature selection.
Selection: The reproduction operator just selects the top individuals of fitness as a part of the next generation, instead of adopting a probability selection algorithm to select superior individuals.
Crossover: Here we use the single-point crossover.
Mutation: Get a probability, and if the value is less than the preset threshold, an individual will be selected and a binary will be changed randomly. In order to retain the best feature subset, all operations mentioned above are among the superiors, and the aberration rate is set to a larger value[10] to escape the local minimum.

Figure 2 is the flowchart of GA. Where *n* is quorum, *m* is selection scale, and *thresh* is fitness-threshold.

GA can lead to different results for each run, so we run the selection function several times and choose the best one. Table 3 and Table 4 show the results of subtask A and subtask B in 20 runs of GA respectively. All experiments below train models on training dataset, and test them on development dataset.

Operation	Statistics	Result
Undo selection	—	0.5263
Do selection	max	0.6182
	min	0.6121
	average	0.6163
	Standard deviation	0.0014

Table 3: 20 runs' results of GA for subtask A.

Operation	Statistics	Result
Undo selection	—	0.5889
Do selection	max	0.7801
	min	0.7320
	average	0.7627
	Standard deviation	0.0137

Table 4: 20 runs' results of GA for subtask B.

[9] The value is 0.75.

[10] The value we set is 0.3.

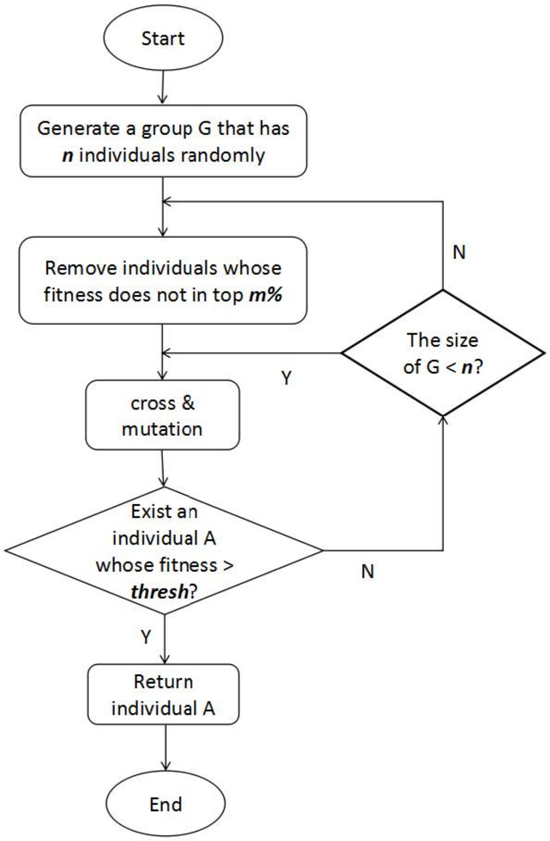
Start
Generate a group G that has *n* individuals randomly
Remove individuals whose fitness does not in top *m%*
The size of G < *n*?
N
Y
cross & mutation
Exist an individual A whose fitness > *thresh*?
N
Y
Return individual A
End

Figure 2: genetic algorithm

3.3 Training Model

We trained a Maximum Entropy Modeling using the maxent toolkit (Le, 2004) and a SVR model (Smola & Schölkopf, 2004) using scikit-learn toolkit (Pedregosa et al., 2011).

Table 5 and Table 6 show the results of different models.

Runs	**MAP**
random baselines	0.4556
IR baselines	0.5384
Maxent	0.5826
SVR	**0.6179**

Table 5: Experiments for Subtask A

Runs	**MAP**
random baselines	0.5595
IR baselines	0.7135
Maxent	0.7135
SVR	**0.7801**

Table 6: Experiments for Subtask B

3.4 Result

We just submit one time, and our system perform better in subtask A and subtask C than subtask B.

	IR	**SYS**
MAP	0.5953	**0.7152**
AvgRec	0.7260	0.8267
MRR	67.83	80.26

Table 7: System performance for subtask A.

	IR	**SYS**
MAP	0.7475	**0.7143**
AvgRec	0.8830	0.8731
MRR	83.79	81.28

Table 8: System performance for subtask B.

	IR	**SYS**
MAP	0.4036	**0.4849**
AvgRec	0.4597	0.5516
MRR	45.83	55.21

Table 9: System performance for subtask C

4 Conclusion and Future Work

We have tested the system by taking part in Semeval2016 Task 3 on English sub tasks, and our system works better on subtask A and subtask C than the IR system provided by organizers.

Aware our method's shortcomings that the features center on lexical similarity, we will pay attention to process long sentence similarity in further work.

Acknowledgments

We thank reviewers for their helpful comments on an earlier version of this work. This work is supported by National Natural Science Foundation of China (61572151 and 61300114).

References

Allison, L., & Dix, T. I. (1986). A bit-string longest-common-subsequence algorithm. *Information Processing Letters, 23*(5), 305-310.

Baroni, M., Dinu, G., & Kruszewski, G. (2014). *Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors.* Paper presented at the ACL (1).

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *the Journal of machine Learning research, 3*, 993-1022.

Fellbaum, C. (2005). WordNet and wordnets.

Finkel, J. R., Grenager, T., & Manning, C. (2005). *Incorporating non-local information into information extraction systems by gibbs sampling.* Paper presented at the Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics.

Hofmann, T. (2001). Unsupervised learning by probabilistic latent semantic analysis. *Machine learning, 42*(1-2), 177-196.

Jiang, J. J., & Conrath, D. W. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. *arXiv preprint cmp-lg/9709008.*

Jin, X., Sun, C., Lin, L., & Wang, X. (2014). Exploiting Multiple Resources for Word-Phrase Semantic Similarity Evaluation *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data* (pp. 46-57): Springer.

Le, Z. (2004). Maximum entropy modeling toolkit for Python and C++. *Natural Language Processing Lab, Northeastern University, China.*

Leacock, C., & Chodorow, M. (1998). Combining local context and WordNet similarity for word sense identification. *WordNet: An electronic lexical database, 49*(2), 265-283.

Lin, D. (1998). *An information-theoretic definition of similarity.* Paper presented at the ICML.

Liu, Y., Sun, C., Lin, L., Zhao, Y., & Wang, X. (2015). Computing Semantic Text Similarity Using Rich Features.

Nakov, P., Marquez, L., Magdy, W., Moschitti, A., Glass, J., & Randeree, B. (2015). *SemEval-2015 Task 3: Answer Selection in Community Question Answering.*

Nakov, P., Marquez, L., Magdy, W., Moschitti, A., Mubarak, H., Freihat, A. A., Glass, J., & Randeree, B. (2016). *SemEval-2016 Task 3: Community question answering.*

Nicosia, M., Filice, S., Barrón-Cedeno, A., Saleh, I., Mubarak, H., Gao, W., . . . Darwish, K. (2015). *QCRI: Answer selection for community question answeringexperiments for Arabic and English.* Paper presented at the Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Dubourg, V. (2011). Scikit-learn: Machine learning in Python. *the Journal of machine Learning research, 12*, 2825-2830.

Pennington, J., Socher, R., & Manning, C. D. (2014). *Glove: Global Vectors for Word Representation.* Paper presented at the EMNLP.

Řehůřek, R., & Sojka, P. (2010). Software framework for topic modelling with large corpora.

Renna, J. (2000). *Genetic algorithm viewer: Demonstration of a genetic algorithm.* Ph. D. May.

Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007.*

Resnik, P. (1999). Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *J. Artif. Intell. Res.(JAIR), 11*, 95-130.

Smola, A. J., & Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and computing, 14*(3), 199-222.

Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation, 28*(1), 11-21.

Wu, Z., & Palmer, M. (1994). *Verbs semantics and lexical selection.* Paper presented at the Proceedings of the 32nd annual meeting on Association for Computational Linguistics.