JAIST: Combining multiple features for Answer Selection in Community Question Answering

Quan Hung Tran[1], **Vu Duc Tran**[1], **Tu Thanh Vu**[2], **Minh Le Nguyen**[1], **Son Bao Pham**[2]
[1]Japan Advanced Institute of Science and Technology
[2]University of Engineering and Technology, Vietnam National University, Hanoi
[1]{quanth,vu.tran,nguyenml}@jaist.ac.jp
[2]{tuvt,sonpb}@vnu.edu.vn

Abstract

In this paper, we describe our system for SemEval-2015 Task 3: Answer Selection in Community Question Answering. In this task, the systems are required to identify the good or potentially good answers from the answer thread in Community Question Answering collections. Our system combines 16 features belong to 5 groups to predict answer quality. Our final model achieves the best result in subtask A for English, both in accuracy and F1-score.

1 Introduction

Nowadays, community question answering (cQA) websites like Yahoo! Answers play a crucial role in supporting people to seek desired information. Users can post their questions on these sites for finding help as well as personal advice. However, the quality of these answers varies greatly. Typically, only a few of the answers in an answer thread are useful to the users and it may take a lot of efforts to identify them manually. Thus, a system that automatically identifies answer quality is much needed.

The task of identifying answer quality has been studied by many researchers in the field of Question Answering. Many methods have been proposed: web redundancy information (Magnini et al., 2002), non-textual features (Jeon et al., 2006), textual entailment (Wang and Neumann, 2007), syntactic features (Grundström and Nugues, 2014). However, most of these works used independent dataset and evaluation metrics; thus it is difficult to compare the results of these methods. The SEMEVAL task 3 (Màrquez et al., 2015) addresses this problem by providing a common framework to compare different methods in multiple languages.

Our system incorporates a range of features: word-matching features, special component features, topic-modeling-based features, translation-based features and non-textual features to achieve the best performance in subtask A (Màrquez et al., 2015). In the remainder of the paper, we will describe our system with the focus on the features.

2 System Description

For extracting the features, we first preprocess the questions and the answers then build a number of models based on training data or other sources (Figure 1).

2.1 Preprocessing

All the questions and the answers are preprocessed through the following steps: Tokenization, POS-tagging, Syntactic parsing, Dependency parsing, Lemmatization, Stopword removal, Name-Entity recognition. These preprocessing steps are completed using The Stanford CoreNLP Natural Language Processing Toolkit (Manning et al., 2014). Because of the noisy nature of community data, the syntactic parsing, dependency parsing and Name-Entity recognition steps do not produce highly accurate results. Thus, we rely mainly on the bag-of-word representation of text. Removing stopwords or lemmatization can alter the meaning of the text, so in the system, we keep both the original version and the processed version of the text. The choice between using the two versions is made using experiments in

Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), pages 215–219,
Denver, Colorado, June 4-5, 2015.

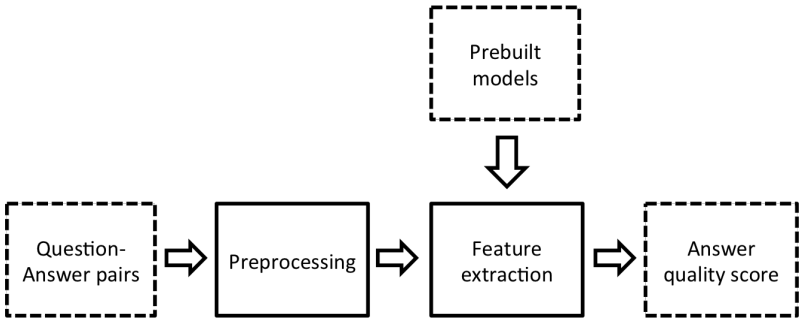

Figure 1: *System components*

development set.

2.2 Building models from data

In this section, we describe the resources we use, or build for extracting features, these resources are: Translation models, LDA models, Word vector representation models, Word Lists. The translation models are built to bridge the lexical chasm between the questions and the answers (Surdeanu et al., 2008). In previous works (Jeon et al., 2005; Zhou et al., 2011), monolingual translation models between questions have been successfully used in finding similar questions in Question Answering archive. We adapt the idea and build translation models between the questions and their answers using the training data and the Qatar Living forum data. We treat the question-answer pairs similar to dual language sentence pairs in machine translation. First, each question-answer pair is tokenized and all special characters are removed. In the process, if any answer has too few tokens (less than two tokens), it is removed from the training data. Then the translation probabilities are calculated by IBM Model 1 (Brown et al., 1993) and Hidden Markov Model. Each model is trained with 200 iterations. The calculated translation probabilities help us to calculate the probability that an answer is the translation of the question. The translation feature will be detailed in Section 2.3.

We build two topic models, the first one is trained in the training data, the second one is trained in Wikipedia data[1] using Gensim toolkit (Řehůřek and Sojka, 2010) and Mallet toolkit (McCallum, 2002). These LDA models have 100 topics. The choice between which model will be used is based on experiments in the development set.

We experiment with two word vector representation models built using Word2Vec tool (Mikolov et al., 2013), the first one is pre-trained word2vec model provided by the authors, and the second one is trained from the Qatar Living forum data. Our Word2Vec model was built with word vector size of 300, window size of 3 (n-skip-gram, n=3) and minimum word frequency of 1. In Section 2.3, we detail how to extract feature using these models.

We also build several word lists from the training set to extract features:

- The words that usually appear on each type of answers (Good, Bad, Potential).
- The words pairs (one from the question, one from the good answers) that have high frequency in the training set. We aim to extract the information about word collocations through this list.

2.3 Features

Word-matching feature group: This feature group exploits the surface word-based similarity between the Question and the Answer to assign score:

- Cosine similarity:

$$cosine\_sim = \frac{\sum_{i=1}^{n} u_i \times v_i}{\sqrt{\sum_{i=1}^{n} (u_i)^2} \times \sqrt{\sum_{i=1}^{n} (v_i)^2}} \quad (1)$$

With *u* and *v* are binary bag of words vectors (with stopwords are removed), u_i is the i-th dimension of vector *u* and *n* is vector size. This

[1]The compressed version of all article from Wikipedia downloaded at http://dumps.wikimedia.org/enwiki/

feature returns the cosine similarity between question vector and answer vector.

- Dependency cosine similarity: We represent the questions and the answers as bag of word-dependency, with words are associated with their dependency label in the dependency tree. For example: a dependency arc in the dependency tree: prep(buy-4, for-7) will generate the following word-dependency: prep-by-for. We consider the sentence to be the collection of these word-dependencies. The cosine similarity score is calculated similar to bag-of-word cosine similarity.

- Word alignment: We also use the Meteor toolkit (Denkowski and Lavie, 2014) to align the words from the question and the answers, and use the alignment score returned as a feature in the feature space

- Noun match: This feature is similar to Cosine similarity feature, however; only nouns are retained in the bag-of-word.

Special-component feature group: This feature group identifies the special characteristics of the answers that show the answer quality:

- Special words feature: This feature identifies if an answer contains some of the special tokens (question marks, laugh symbols). Typically, the posts that contains this type of tokens are not a serious answer (laugh symbols), or a further question (question marks). The laugh symbols are identified using a regular expression.

- Typical words feature: This feature identifies if an answer contains some specific words that are typical for an answer quality class (good, bad, potential). The typical word lists are built using training data and described in the previous section. After the experiment step, however, only the typical word list for bad answers was found to be effective and was used in the final version of the system.

Non-textual feature group: This feature group exploits some non-textual information of the posts in the answer thread to assign answer quality:

- Question author feature: This feature identifies if an answer in the answer thread belongs to the author of the question. If a post belongs to the author of the question, it is very unlikely to be an answer.

- Question category: We also include the question category (27 categories) in the feature space because we found out that the quality distribution of different types of question are very different.

- The number of posts from the same user: We include the number of posts from the same user as a feature because we observe that if a user has a large number of posts, most of them will be non-informative, irrelevant to the original question.

Topic model based feature: We use the previously mentioned LDA models to transform questions and answers to topic vectors and calculate the cosine similarity between the topic vectors of the question and its answers. We use this feature because a question and its correct answer should be about similar topics. After experimenting on the development set, only the LDA model built from training data is effective and thus, it is used in the final system.

Word Vector representation based feature: We use the word vector representation to model the relevance between the question and the answer. All the questions and answers are tokenized and the words are transformed to vector using the pretrained word2vec model. Each word in the question will then be aligned to the word in the answer that has the highest vector cosine similarity. The returned value will be the sum of the scores of these alignments normalized by the question's length:

$$align(w_i) = \max_{0<j\leq m}(cosine(w_i, w'_j)) \quad (2)$$

$$word2vec\_sim = \frac{\sum_{i=1}^{n} align(w_i)}{n} \quad (3)$$

With $cosine(w_i, w'_j)$ is the cosine similarity of two vector representations of i-th word in the question with the j-th word in the answer. *n* and *m* are the length (in number of words) of the question and the answer respectively.

Translation based feature: We use the previously mentioned translation models to find the word to word alignments between the question and the answer. This feature is calculated similar to the Word Vector representation based feature. Each word in the question will be aligned with the word in the answer with the highest translation score. The feature value will be the sum of translation scores normalized by question' length.

2.4 System run configuration

The straightforward way to identify the quality classes for answers is using a classification model. However, the classification model has problem in identifying the Potential class. In our experiments, the classification model ignores the Potential class entirely. This problem may be caused by our feature design as the features actually aim to identify either good or bad answers.

To solve this problem, we use another approach. As we observe the data, most of the Potential answers can be considered "Not good enough" and "Not bad enough". An answer which is not quite good nor quite bad can be considered "Potential", thus using a regression model[2] to score the quality of the answer would probably be better. In our experiment with the development data, the regression model outperforms the classification model by 3.4 F-measure score.

Features are extracted from the answers (with their questions treated as the context), and then the feature values are passed through a regression model. However, the provided data only has quality classes but not regression value, thus we need to assign the regression value for each answer quality class: 1.0 for Good answers, 0.5 for Potential answers, and 0.0 for Bad answers.

Our system runs are different in the feature space. Our best run (JAIST-contrastive1) uses all the features described above. Our main run (JAIST-primary) excludes the topic-modeling based feature while the third run (JAIST-contrastive2) includes several other experimental features that did not have contribution when tested on the development set.

[2]We use SVM-regression model in WEKA toolkit (Hall et al., 2009)

Table 1: System performance

Runs	F1-score	Accuracy	Rank
primary	57.19 (%)	72.52 (%)	2
contrastive1	57.29 (%)	72.67 (%)	1
contrastive2	46.96 (%)	57.74 (%)	18

Table 2: Detail Class F1-score

Runs	F1-score
Good	78.96 (%)
Bad	78.24 (%)
Potential	14.36 (%)

3 Result and Discussion

We only take part in subtask A for English. Our system has the best accuracy and F1-score in subtask A (primary runs) shown in Table 1. Classifying the Potential class is quite difficult (Màrquez et al., 2015) and our system only achieve 14.36 % F1 score on this class. Although the use of regression model partly solves this problem, our feature space is not adequate for identifying this class reliably (Table 2)

4 Conclusion

In this paper, we present our approach for the subtask A - English of the SEMEVAL 2015 task 3 - Answer Selection in Community Question Answering. We propose an Answer quality scoring based approach for classifying answers in Community Question Answering. Our system achieves high results in the task, however, does not handle the Potential class well. A possible explanation is that we still rely heavily on the bag-of-word representation of text. In the future, other semantically rich representations of text would be employed to improve performance.

References

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):263–311, June 1993.

Jakob Grundström and Pierre Nugues. Using Syntactic Features in Answer Reranking. In *AAAI 2014 Workshop on Cognitive Computing for Augmented Human Intelligence*, pages 13–19, 2014.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA Data Mining Software: An Update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009.

Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. Finding Similar Questions in Large Question and Answer Archives. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, pages 84–90, New York, NY, USA, 2005.

Jiwoon Jeon, W. Bruce Croft, Joon Ho Lee, and Soyeon Park. A Framework to Predict the Quality of Answers with Non-textual Features. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 228–235, New York, NY, USA, 2006.

Michael Denkowski Alon Lavie. Meteor Universal: Language Specific Translation Evaluation for Any Target Language. *ACL 2014*, page 376, 2014.

Bernardo Magnini, Matteo Negri, Roberto Prevete, and Hristo Tanev. Is it the right answer?: exploiting web redundancy for Answer Validation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 425–432, 2002.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014.

Andrew Kachites McCallum. MALLET: A Machine Learning for Language Toolkit. http://mallet.cs.umass.edu, 2002.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *CoRR*, abs/1301.3781, 2013.

Lluís Màrquez, James Glass, Walid Magdy, Alessandro Moschitti, Preslav Nakov, and Bilal Randeree. SemEval-2015 Task 3: Answer Selection in Community Question Answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*.

Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010.

Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. Learning to rank answers on large online QA collections. In *In Proceedings of the 46th Annual Meeting for the Association for Computational Linguistics: Human Language Technologies (ACL-08: HLT*, pages 719–727, 2008.

Rui Wang and Günter Neumann. DFKILT at AVE 2007: Using Recognizing Textual Entailment for Answer Validation. In *In online proceedings of CLEF 2007 Working Notes, ISBN*, pages 2–912335, 2007.

Guangyou Zhou, Li Cai, Jun Zhao, and Kang Liu. Phrase-based Translation Model for Question Retrieval in Community Question Answer Archives. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 653–662, Stroudsburg, PA, USA, 2011.