2019

Enhanced Answer Selection in CQA Using Multi-Dimensional Features Combination

Hongjie Fan
*the School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China.*

Zhiyi Ma
*the School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China.*

Hongqiang Li
*the School of Software and Microelectronics, Peking University, Beijing 100871, China.*

Dongsheng Wang
*the School of Software and Microelectronics, Peking University, Beijing 100871, China.*

Junfei Liu
*National Engineering Research Center for Software Engineering, Peking University, Beijing 100871, China.*

Follow this and additional works at: https://tsinghuauniversitypress.researchcommons.org/tsinghua-science-and-technology

Part of the Computer Sciences Commons, and the Electrical and Computer Engineering Commons

Recommended Citation


This Research Article is brought to you for free and open access by Tsinghua University Press: Journals Publishing. It has been accepted for inclusion in Tsinghua Science and Technology by an authorized editor of Tsinghua University Press: Journals Publishing.
Enhanced Answer Selection in CQA Using Multi-Dimensional Features Combination

Hongjie Fan, Zhiyi Ma*, Hongqiang Li, Dongsheng Wang, and Junfei Liu

Abstract: Community Question Answering (CQA) in web forums, as a classic forum for user communication, provides a large number of high-quality useful answers in comparison with traditional question answering. Development of methods to get good, honest answers according to user questions is a challenging task in natural language processing. Many answers are not associated with the actual problem or shift the subjects, and this usually occurs in relatively long answers. In this paper, we enhance answer selection in CQA using multi-dimensional feature combination and similarity order. We make full use of the information in answers to questions to determine the similarity between questions and answers, and use the text-based description of the answer to determine whether it is a reasonable one. Our work includes two subtasks: (a) classifying answers as good, bad, or potentially associated with a question, and (b) answering YES/NO based on a list of all answers to a question. The experimental results show that our approach is significantly more efficient than the baseline model, and its overall ranking is relatively high in comparison with that of other models.

Key words: community question answering; information retrieval; multi-dimensional features extraction; similarity computation

1 Introduction

Web forums for Community Question Answering (CQA), such as Stack Overflow, provide an interface for users to share knowledge, and communicate with each other[1]. CQA is a powerful mechanism that allows users to freely ask questions and look forward to obtaining good, honest answers. In this way, users can obtain specific answers to their questions, instead of searching through large volumes of information.

The increasing popularity of CQA websites has caused number of questions and new forum members to surge without restriction. Unfortunately, much effort must be exerted to assess all possible answers and select one that is the most accurate for a specific question. Thus, although the idea of receiving a direct response to a certain request for information is very appealing, CQA websites also cannot guarantee the quality of the information provided[2]. Many answers are often loosely related to the actual question, and some even shift the topic away from the subject. This issue is a common finding in relatively long answer, i.e., as the answer continues, users begin to discuss with each other instead of answering the initial question.

This issue presents a real problem, as a question can have hundreds of answers. Thus, searching for good answers among the many responses is necessary but time-consuming for participants. Some studies have
exploited techniques such as the use of syntactic tree structures to measure the semantic correspondence between a question and answer or translation-based language models for matching by transferring the answer to the corresponding question.

However, guaranteeing the quality of the provided information remains a major challenge. In this work, we adopt the following ideas to tackle the issue of quality in CQA:

1. Traditional Question Answering (QA) based on similarity sorting. We make full use of answers to questions and find the similarity between them. We then look at the forum closely and speculate that if the questions and answers are similar, then the answer must be more credible. In addition, if questions and answers are presented by the same user, the similarity among them could be higher than between questions and answers provided by different users.

2. Text-based descriptions of the answer to determine whether the answer is reasonable. We consider that if the answer is long, then it is likely to be a reasonable answer. In addition, if the answer is Yes, No, and several other words, then it is likely to be a reasonable answer. In this work, we take SemEval-2015 Task 3 (http://alt.qcri.org/semeval2015/) on answer selection in CQA to verify our hypothesis. Two subtasks are described in Section 2.2.

To handle these tasks, we enhance answer selection using the multi-dimensional feature combination method. Our main contributions are as follows:

1. We present a framework for enhancing answer selection in CQA using the multi-dimensional feature combination method.

2. We extract information for each question and comment from the data set. Twenty features are extracted based on content description, text similarity, and attribute description.

3. We build a model from these features using the machine learning approaches, Support Vector Machine (SVM), Gradient Boosting Decision Tree (GBDT), and random forest, to classify the dimensions obtained.

4. We conduct an experiment to show that our three approaches are significantly more efficient than the baseline model, and that its overall ranking is relatively high compared with those of other methods.

The rest of this paper is organized as follows: Section 2 introduces the background and preliminaries of CQA, task definition, and multi-dimensional feature extraction. Section 3 presents the methods including the selection of features, formulation of labels, and construction of these models. Experiments and the related discussion are conducted and provided in Sections 4 and 5, respectively. Section 6 analyzes the related work, and Section 7 draws conclusions and offers directions for future work.

2 Background

In this section, we briefly introduce the concepts of CQA, task description, and multi-dimensional feature extraction.

2.1 CQA

CQA websites provide an interface for users to share knowledge[3]. Unfortunately, given the increasing popularity of CQA forums, a user must browse all possible answers and understand them before arriving at the correct answer. Answer selection in CQA recognizes high-quality responses, which is greatly valuable for information retrieval, and aims to automatically categorize answers as good if they completely answer the question, bad if they are irrelevant to the question, and potential if they contain useful information about the question but do not completely answer it. Figure 1 shows an example question and its four possible answers. The dashed arrows depict the relationships of the answers in the sequence.

The title of “Can we resign from the job?” gives a brief summary of the question, and the body describes the question in detail. The question includes four parts. After checking all answers, good answers among a1, a2, a3, and a4 can be obtained. Answer a1 clearly is a good answer because it provides helpful information such as “contact”, and “require the employee to pay a certain amount of fee”. Besides, this answer mentions

![Fig. 1 An example of the answer body for a question.](image-url)
other details such as “how long have you been working with the company”. This answer is also long, which means it is likely to be a reasonable answer. Answer a2 is also a reply to the question but it does not contain any useful information; thus, it is regarded as a bad answer. In answer a4, some sentences are auxiliary and do not provide meaningful information for answer selection in CQA, it is better than answer a2 but not good enough as an answer result for the question. As such, a4 can be treated as a potential answer. Some answers, such as a3 and a4, would reply the former answer.

When a large number of CQA websites are available, a CQA website must be able to provide high-quality content to distinguish itself from other websites. Thus, although the idea of receiving a direct response to a certain need sounds very appealing, some risk is present because the quality of the provided information cannot be guaranteed.

In Ref. [4], the authors show a correlation between question quality and answer quality; good answers are more likely to be given in response to good questions, while bad answers appear in response to bad questions. According to the definition in Ref. [5], higher quality questions can be expected to draw greater user attention, include more answer attempts, and obtain the best answer within a shorter period of time than poorer-quality questions.

2.2 Task definition

Based on this description of CQA, we take the SemEval-2015 Task 3 in CQA to test our hypotheses. This task includes two subtasks:

Task A asks types of prediction for all questions under given questions and answers. It gives a question (including a short title and extended description) and its answers and then divides each answer into one of the following:

(a) Absolutely related (good);
(b) Bad or irrelevant (bad, dialog, non-English, or other) or;
(c) Potentially useful (potential).

Task B asks the prediction about Yes, No, or Unsure based on a list of all answers. It gives a YES/NO type problem (including a short title and its extended description) and a few answers based on good answers to the questions asked in task A.

2.3 Multi-dimensional feature extraction

Features are extracted from the answers, and feature values are passed through a regression model. Thus, we need to assign a regression value to each answer quality class. For example, a regression value of 1.0 can be assigned to good answers, a value of 0.5 can be assigned to potential answers, and a value of 0.0 can be assigned to bad ones.

Multi-dimensional feature extraction includes two parts: features based on contents and features based on attributes. We provide details for each type of feature extraction in Section 3.2. Five similarity calculation models, including Latent Semantic Index (LSI)[6], Latent Dirichlet Allocation (LDA)[7], Sentence LDA (SenLDA)[8], Word2Vector[9], and BagofWords[10], are proposed to analyze the final output answer quality scores.

3 Framework of Enhanced Answer Selection in CQA

For processing the enhanced answer selection in CQA, we combine multi-dimensional features to build several models. In this section, we present the process of model-building, which includes three components as shown in Fig. 2: data preprocessing, types of feature extraction, and classification of all features using SVM, GBDT, and random forest.

In the first step, data preprocessing is conducted to input data into the storage system. The parsing and partitioning steps are normally executed once from the given document. In the second step, we extract features based on contents and attributes. In the third step, we build models using the extraction features and assign an answer quality score.

3.1 Data preprocessing

Data preprocessing is an important phase of model-building as it extracts various information from websites and presents it in a database.

As shown in Fig. 3, an XML parser receives the input corpus in XML format. The XML file contains all of the questions, along with their respective answers.

An XML interpreter extracts the questions and associated answers. Here we describe the main preprocessing steps on a collection of CQA-QL corpus of SemEval-2015 Task 3 on answer selection.

During data preprocessing, tokenization, stop word removal, and stemming are common tasks applied to process the content of the documents. First, we tokenize
3.1 Data preprocessing

The content and attribute information is processed as training data, dev data, and test data.

3.2 Types of feature extraction

XML file cannot be split because they include opening and closing tags at the beginning and end, respectively, of the document. As such, we cannot begin processing at any point between those tags.

For example, we have an XML document representing Fig. 2, which includes the text "Can we resign from the job?? {QSubject} Can we resign from the job?? {QSubject}". A fatal
error is produced in XML grammar if the document is split into \( f_s_i \) representing "\( \text{QSubject} \) Can we resign from the job?" and \( f_s_{i+1} \) representing "\( \text{QSubject} \)”, because of the end-tag of an element must be intact in the form of '/tagname').

To solve this problem, the raw data can be divided into two parts: the content and the attribute of the XML information. The content information includes the content description, so as the text in specific XML information. The content information includes tags such as \( \text{QID}, \text{QSubject}, \text{QBody}, \text{CSubject}, \text{CBody} \). The attribute information includes tags such as \( \text{QID}, \text{QCATEGORY}, \text{QUSERID}, \text{QTYPE}, \text{QGOLD}, \text{YN}, \text{CID}, \text{CUSERID}, \text{CGOLD}, \text{CGOLD}.YN. \)

After preprocessing the raw data, we classify features into three types according to their characteristics: based on content description, based on text similarity, and based on attribute description.

The feature types, descriptions, and feature numbers are presented in Table 1.

### 3.2.1 Feature extraction based on content description

We have selected some key words to describe whether the answer is reasonable or not. Table 2 shows some of these features (13 features in this example) based on content description.

For example, if the answer to the text description appears "URL", "Email", which represents the key words of information, we think that the answer may contain valid information, which is a reasonable answer. If Yes, No, OK, and similar responses are provided, the answer could be the effective answer to the question. In addition, if the length of the answer is long, the information included in the answer information could be substantial and the answer is likely to be valid.

### 3.2.2 Feature extraction based on text similarity

Meaningful words can express the information of an entire sentence. Therefore, we determine the answer to a question through text similarity. We choose five popular similarity calculation models, namely LSI\[^6\], LDA\[^7\], SenLDA\[^8\], Word2Vector\[^9\], and BagofWords\[^10\], to analyze the final output answer quality score. These models are unsupervised.

Table 3 shows all similarity calculation models based on text similarity. These models can express the semantics of the whole text in various forms. Thus, they must be built with the training, devel, and test datasets as inputs for training. Finally, we obtain the similarity measure of two texts by calculating the cosine distance of their vectors.

1. **LSI-based feature extraction**

   LSI\[^6\] is an effective spectral document clustering method. This model explains text documents by mixing latent topics and analyzes the relationships between documents and terms. LSI decomposes the text vector onto the vector space of the topic size of the dimension through singular value decomposition to identify patterns in the relationships between the terms and concepts contained in a collection of texts.

   A key feature of LSI is its ability to retain latent structures in words; thus, it improves the clustering efficiency. Using Singular Value Decomposition (SVD), we project the text word from the three corpora to the dimensional vector space. Thereafter, we compute the \( tf-idf \) value of each word in the documents.

2. **LDA-based features extraction**

   Topic models are a set of models representing documents in a collection or corpus. They enable us to represent the properties of a large corpus containing numerous words with a small set of topics by

<table>
<thead>
<tr>
<th>Feature type</th>
<th>Description</th>
<th>Number of features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type1</td>
<td>Based on content description</td>
<td>13</td>
</tr>
<tr>
<td>Type2</td>
<td>Based on text similarity</td>
<td>5</td>
</tr>
<tr>
<td>Type3</td>
<td>Based on attribute description</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Dimension</th>
<th>Value type</th>
</tr>
</thead>
<tbody>
<tr>
<td>hasURL</td>
<td>Has(Not) URL</td>
<td>1</td>
<td>Bool</td>
</tr>
<tr>
<td>hasYes</td>
<td>Number of Yes in Contents</td>
<td>1</td>
<td>Int</td>
</tr>
<tr>
<td>hasOk</td>
<td>Number of Ok/Okay</td>
<td>1</td>
<td>Int</td>
</tr>
<tr>
<td>startWithYes</td>
<td>startWithYes or Not</td>
<td>1</td>
<td>Bool</td>
</tr>
<tr>
<td>wordNums</td>
<td>Number of Words</td>
<td>1</td>
<td>Int</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Dimension</th>
<th>Value type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SimLDA</td>
<td>LDA_Sim(Q, C)</td>
<td>1</td>
<td>Float</td>
</tr>
<tr>
<td>SimLDA</td>
<td>LDA_Sim(Q, C)</td>
<td>1</td>
<td>Float</td>
</tr>
<tr>
<td>SimSenLDA</td>
<td>SenLDA_Sim(Q, C)</td>
<td>1</td>
<td>Float</td>
</tr>
<tr>
<td>SimW2Vec</td>
<td>W2Vec_Sim(Q, C)</td>
<td>1</td>
<td>Float</td>
</tr>
<tr>
<td>SimBOWS</td>
<td>Bows_Sim(Q, C)</td>
<td>1</td>
<td>Float</td>
</tr>
</tbody>
</table>

### Table 1. Types of feature extraction.

<table>
<thead>
<tr>
<th>Feature type</th>
<th>Description</th>
<th>Number of features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type1</td>
<td>Based on content description</td>
<td>13</td>
</tr>
<tr>
<td>Type2</td>
<td>Based on text similarity</td>
<td>5</td>
</tr>
<tr>
<td>Type3</td>
<td>Based on attribute description</td>
<td>2</td>
</tr>
</tbody>
</table>

### Table 2. Features based on content description.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Dimension</th>
<th>Value type</th>
</tr>
</thead>
<tbody>
<tr>
<td>hasURL</td>
<td>Has(Not) URL</td>
<td>1</td>
<td>Bool</td>
</tr>
<tr>
<td>hasYes</td>
<td>Number of Yes in Contents</td>
<td>1</td>
<td>Int</td>
</tr>
<tr>
<td>hasOk</td>
<td>Number of Ok/Okay</td>
<td>1</td>
<td>Int</td>
</tr>
<tr>
<td>startWithYes</td>
<td>startWithYes or Not</td>
<td>1</td>
<td>Bool</td>
</tr>
<tr>
<td>wordNums</td>
<td>Number of Words</td>
<td>1</td>
<td>Int</td>
</tr>
</tbody>
</table>

### Table 3. Features based on text similarity.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Dimension</th>
<th>Value type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SimLDA</td>
<td>LDA_Sim(Q, C)</td>
<td>1</td>
<td>Float</td>
</tr>
<tr>
<td>SimLDA</td>
<td>LDA_Sim(Q, C)</td>
<td>1</td>
<td>Float</td>
</tr>
<tr>
<td>SimSenLDA</td>
<td>SenLDA_Sim(Q, C)</td>
<td>1</td>
<td>Float</td>
</tr>
<tr>
<td>SimW2Vec</td>
<td>W2Vec_Sim(Q, C)</td>
<td>1</td>
<td>Float</td>
</tr>
<tr>
<td>SimBOWS</td>
<td>Bows_Sim(Q, C)</td>
<td>1</td>
<td>Float</td>
</tr>
</tbody>
</table>
representing documents according to these topics. Blei et al.\[^7\] proposed LDA as a generative probability model of a corpus that can be used to estimate multinomial observations using unsupervised learning.

A probabilistic generative model considers data as observations originating from a generative probabilistic process that includes hidden variables. The hidden variable is typically inferred via posterior inference. In posterior inference, one tries to identify the posterior distribution of the hidden variables that are conditioned on the observations.

LDA can set the number of topics to be determined, and the topic distribution of each text is obtained by training. Table 4 shows notations for LDA where the words \( w \) of a document share the same topic \( z \) and SenLDA (details to be presented later).

Topics in LDA are drawn from a conjugate Dirichlet prior that remains the same for all documents. For each document \( d \) in a corpus \( D \), LDA assumes the following generative process (Fig. 4):

(a) Sample document length \( N_d \) from a Poisson distribution \( N_d \sim \text{Poisson}(\xi) \).

(b) Pick a topic distribution \( \tilde{\theta}_d \) from a Dirichlet distribution \( \tilde{\theta}_d \sim \text{Dirichlet}(\alpha) \).

(c) For the \( n \)-th word in \( N_d \) words:
   (i) Choose a topic \( z_{d,n} \) from the multinomial distribution \( z_{d,n} \sim \text{Multinomial}(\tilde{\theta}_d) \).
   (ii) Choose a word \( w_{d,n} \) from the multinomial distribution \( \text{Multinomial}(\tilde{\varphi}_{z_{d,n}}) \) with the topic-word distribution \( \tilde{\varphi}_{z_{d,n}} \) sampled from a Dirichlet distribution \( \tilde{\varphi}_{z_{d,n}} \sim \text{Dirichlet}(\beta) \).

LDA estimation is achieved using the approximate estimation method called Gibbs sampling\[^13\]. We sample the topic assignment of each word \( w \) following the multinomial distribution:

\[
p(w_k = k | \tilde{z}_w, \tilde{w}, \alpha, \beta) = \frac{n_{k,\tilde{w}} + \beta}{\sum_{v=1}^{K} (n_{k,v} + \beta) - 1} \frac{n_{\tilde{w},k} + \alpha}{\sum_{j=1}^{N_d} (n_{d,j} + \alpha) - 1}
\]

where \( \tilde{z}_w \) denotes the topic assignments of all words except the current assignment, \( n_{k,\tilde{w}} \) is the number of times topic \( k \) is assigned to the word \( w \) except in the current assignment, \( \sum_{v=1}^{K} n_{k,v} - 1 \) is the total number of times that topic \( k \) is assigned to words in vocabulary \( n_{\tilde{w},k} \) except the current assignment, \( \tilde{w} \) is the number of words in document \( d \) assigned to topic \( k \) except the current assignment, and \( \sum_{j=1}^{N_d} n_{d,j} - 1 \) is the total number of words in document \( d \), not counting the current word.

At the last iteration of sampling, the topic assignment of each word is saved to enrich the corpus. Traditionally, topic models assume that each word occurrence within a document is independent.

(3) SenLDA-based feature extraction

SenLDA extends the LDA by adding an extra plate denoting the coherent text segments of a document. The goal of SenLDA is to overcome the limitation of LDA by incorporating the structure of the text in the generative and inference processes.

As shown in Fig. 5, SenLDA assumes the following
being sampled. The number of words in document $d$ from document $k$ number of times that topic $n$ number of times topic $k$ is assigned to topic $k$.

We use Word Vector representation-based features to model the relevance between the question and its answer. All of the questions and answers are tokenized, and the words are transformed into vector using the pretrained Word2Vec model. Each word in the question will then be aligned with 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:

$$\text{align}(w_i) = \max_{0<j\leq m} \cos(w_i, w'_j)$$

$$\text{sim}_{\text{word2vec}} = \sum_{i=1}^n \text{align}(w_i)/n$$

where $\cos(w_i, w'_j)$ is the cosine similarity of two vector representations of the $i$-th word in the question with the $j$-th word in the answer, and $n$ and $m$ are the lengths (in number of words) of the question and answer, respectively. Here, we use cosine distance since, compared with other distances, such as Euclidean distance, the cosine distance pays more attention to the difference of two vectors.

(5) BagofWords-based feature extraction

The BagofWords (BOW) model is a recently proposed framework for learning continuous word representations based on the distributional hypothesis. The BagofWords representation disregards the linguistic structures between words. Each dimension represents the frequency of a word, and the cosine distance is used to measure the similarity of two texts. We learn parameter values to maximize the log likelihood of each token given its context:

$$L(\theta) = \frac{1}{N} \sum_{i=1}^N \log \sum_{w_{i-k}^{i+k}} p(w_i | w_{i-k}^{i+k})$$

where $N$ is the size of the corpus and $w_{i-k}^{i+k}$ is the set of words in the window of size $k$ centered at $w_i$ (w_j excluded). The BOW model formulates the probability $p(w_i | w_{i-k}^{i+k})$ using a softmax function as follows:

$$p(w_j | w_{i-k}^{i+k}) = \frac{\exp(v_{w_j} \cdot \sum_{w \in W} v_{w_{i+j}})}{\sum_{w \in W} \exp(v_{w_j} \cdot \sum_{w \in W} v_{w_{i+j}})}$$

where $v_w$ and $v'_w$ represent the input and output vectors of the word $w$, respectively. To train the model efficiently, hierarchical softmax and negative sampling techniques are used. Some morphology-based methods have been proposed, for example, Ref. [14].

BagofWords features include $n$-grams, parts of speech, and features that account for the presence and absence of subjective words. The approach is quite simple to implement and attractive because this model reduces the feature space of a potentially large number of dimensions and can help classifiers boost their
3.2.3 Feature extraction based on attribute description

In terms of question and answer attribute information, if the problem user and the answer user are the same person, then the question is related to the answer. Different types of questions may be difficult to answer because of the different contents involved. Based on this observation, we extract the features shown in Table 5.

3.3 Model building

Classification is a general process related to categorization, the process through which objects are differentiated. In this paper, the types of features we have obtained must be classified. The common classification models include logistic regression, SVM, naive Bayes, GBDT, and random forest.

In the present case, we use SVM, GBDT, and random forest. The SVM model can implicitly map inputs into high-dimensional feature spaces and perform non-linear classification using what is called the kernel trick. The GBDT model can distinguish a variety of features and combine these features. The random forest model is capable of handling high-dimensional features without having to select features during fast training.

3.3.1 SVM model building

SVM[17] is a popular methodology for binary classification and a number of modified formulations have been derived from it. Consider a set of training vectors \( \{x_i \in \mathbb{R}^n, i = 1, \ldots, m\} \) and its corresponding set of labels \( \{y_i \in \{-1, 1\}, i = 1, \ldots, n\} \), this model can predict the class labels of unseen data.

The soft-margin SVM training problem can be expressed as follows:

\[
\min_{w,b,\xi} \frac{1}{2} \|w\|^2 + c \sum_{i=1}^{n} \xi_i
\]

subject to

\[
\begin{align*}
& y_i(w^T x_i + b) + \xi_i, \quad i = 1, \ldots, n, \\
& \xi_i \geq 0, \quad i = 1, \ldots, n
\end{align*}
\]

where \( \xi_i \) is a slack variable associated with a penalty term in the objective with a magnitude controlled by \( c \), a problem specific parameter. Vector \( w \) is the vector normal to the separating hyperplane \( (w^T x_i + b) = 0 \), and \( b \) is its position relative to the origin. Formula (7) maximizes the margin \( 2/\|w\| \) between the two separating hyperplanes \( (w^T x_i + b = 1) \) and \( (w^T x_i + b = -1) \). The use of slack variable \( \xi_i \) penalizes data points that fall on the wrong side of these hyperplanes.

We use the NLTK[11] to carry out various natural language processes on the raw documents. The documents are tokenized into raw words, and these words are converted based on morphology and part-of-speech tagging. They are also filtered by stop words provided by the NLTK. Thereafter, we compute the \( t f-idf \) value of each word in the documents and construct the vector \( d_i = (t_{i1}, \ldots, t_{im}) \), where \( t_{ik} \) is the \( t f-idf \) value of word \( k \) in document \( i \) of collection \( D \), \( m \) is the number of all words used in the collection.

Document similarity is computed by the cosine similarity of vectors using \( t f-idf \) as weights.

\[
sim_{\text{cos}} = \frac{\sum_{k=1}^{m} t_{i1}t_{jk}}{\sqrt{\sum_{k=1}^{m} t_{i1}^2} \sqrt{\sum_{k=1}^{m} t_{jk}^2}}, \quad t_{ik} = t_{fk} \cdot idf_{kD}
\]

3.3.2 GBDT model building

GBDT[18] is a gradient boosting algorithm that utilizes decision stumps or regression tress as weak classifiers. In GBDT, weak learners measure the error observed in each node, split the node using a test function \( \kappa: \mathbb{R}^n \rightarrow \mathbb{R} \) with a threshold \( \tau \), and then return the values \( \eta^l \) and \( \eta^r \). The optimal split of \( (\tau, \eta^l, \eta^r) \) to minimize the error after the split is given by

\[
\varepsilon(\tau) = \sum_{i: \kappa(x_i) < \tau} w_i^j (r_i^j - \eta^l)^2 + \sum_{i: \kappa(x_i) \geq \tau} w_i^j (r_i^j - \eta^r)^2
\]

where \( w_i^j \) and \( r_i^j \) respectively denote the weight and response of \( x_i \) in the \( j \)-th iteration. Formally, these terms are expressed as

\[
w_i^j = \exp(-y_i f_{j-1}(x_i))
\]

\[
r_i^j = g(x_i)/w_i^j = -y_i \exp(-y_i f_{j-1}(x_i))/w_i^j = -y_i
\]

We identify the optimal triplet \( (\tau^*, \eta^l^*, \eta^r^*) \) by minimizing the error over all possible \( \tau \)'s at each node, given \( \tau \) and \( (\eta^l^*, \eta^r^*) \) can be found simply by computing the weighted average of \( r_i^j \)'s over training examples that fall on the corresponding side of the split. \( \eta^l^* \) and \( \eta^r^* \) are given to the left and right children of the current node, respectively, and \( \eta \)'s stored in the leaf node are used as a scores of weak learners corresponding to the tree depending on its input \( x \).

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Dimension</th>
<th>Value type</th>
</tr>
</thead>
<tbody>
<tr>
<td>user-EqualQuser</td>
<td>Same(Not) User of EqualQuser</td>
<td>1</td>
<td>Bool</td>
</tr>
<tr>
<td>QCategory-Question corresponds to</td>
<td>6</td>
<td>List</td>
<td></td>
</tr>
<tr>
<td>Probability-all Comment’s CGOLD</td>
<td>6</td>
<td>List</td>
<td></td>
</tr>
</tbody>
</table>
Reference [19] provides more details on this algorithm.

3.3.3 Random forest model building

The random forest algorithm [20] is an ensemble classifier algorithm based on the decision tree model. It generates $k$ different training data subsets from an original dataset using a bootstrap sampling approach. Each sample of the testing dataset is predicted by all decision trees, and the final classification result is returned depending on the votes of these trees. Random forest models perform well in classification tasks and work efficiently on large datasets.

We train and save a random forest classification model based on the features we extracted. We apply the random forest model to the continuous input of the training set and collect false positives and negatives, which are examples of intervals from the training set that the classifier fails to classify correctly. The sets of false positive and negative instances are then added to the original training set, and the random forest is retrained using the new extended set of training examples.

4 Experimental Evaluation

We provide an experimental study of our method for evaluation. This demonstration uses the original CQA-QL corpus [2] shown in Table 6. The dataset consists of 3229 questions. The total number of comments is 20,162, 9941 of which were good (49.31%), 2013 of which were potential (9.98%), and 8208 of which were bad (40.71%) comments; each question had an average of 6.24 comments. In addition, each question had a title and description and answers.

Compared with those of the training data set, the $CGOLD$ value of the label distribution was basically similar but the $QGOLD_{YN}$ value of the label distribution was relatively large. Our experiments were performed on a cluster of four machines running on Ubuntu 14.04 LTS. Each node was equipped with an Intel Xeon E5-2609 2.5 GHz CPU and 32 GB of memory.

We used train, devel, and test datasets to train the LSI, LDA, SenLDA, Word2Vec, and BOW models to extract similarity features. The SVM, random forest, and GBDT models were also trained. We performed the experiments by setting the general number class to 6 and the $YES/NO$ number class equal to 3. In the SVM model, we set the parameter $C$ to 1 and gamma equal to 0.001 in the rbf kernel function. In random forest model, the max depth in the DecisionTreeClassifier function was set to 4. In the GBDT model, the parameter $n_{estimators}$ was set to 1000, learning_rate was set to 0.2, and max_depth was set to 2 in the function GradientBoostingClassifier.

Experiments were performed by Macro F1, Accuracy, and Ranking according to the test dataset. The performance of differentiating good and bad answers are among all answers.

We compared our method against the JAIST [21], HITSZ-ICRC [22], QCRI [23], ECNU [24], ICRC-HIT [25], VectorSlu [26], Shiraz [27], FBK-HLT [28], Voltron [29], and CICBUAPnlp [30].

5 Results and Discussion

Table 7 shows the results of Task A. In the evaluation, only the measurement of the Good, Potential, and Bad three labels are conducted.

JAIST features are extracted from the answers (with their questions treated as the context), and the feature values are passed through a regression model. In our proposed method, we apply SVM, GBDT, and random forest classifiers to select high-quality result with a ranking function, similar to HITSZ-ICRC. However, the latter uses syntax and deep semantic features to improve its performance. The features determined by QCRI use a supervised machine learning approach and a manifold of features, including word $n$-grams,
Table 7: Result of Task A.

<table>
<thead>
<tr>
<th>Model</th>
<th>Macro F1 (%)</th>
<th>Accuracy (%)</th>
<th>Ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>22.36</td>
<td>50.46</td>
<td>–</td>
</tr>
<tr>
<td>JAIST</td>
<td>57.19</td>
<td>72.52</td>
<td>1</td>
</tr>
<tr>
<td>HITSZ-ICRC</td>
<td>56.41</td>
<td>68.67</td>
<td>2</td>
</tr>
<tr>
<td>QCRI</td>
<td>53.74</td>
<td>70.50</td>
<td>3</td>
</tr>
<tr>
<td>ECNU</td>
<td>53.47</td>
<td>70.55</td>
<td>4</td>
</tr>
<tr>
<td>ICRC-HIT</td>
<td>49.60</td>
<td>67.68</td>
<td>5</td>
</tr>
<tr>
<td>VectorSlu</td>
<td>49.10</td>
<td>66.45</td>
<td>6</td>
</tr>
<tr>
<td>Shiraz</td>
<td>47.34</td>
<td>56.83</td>
<td>7</td>
</tr>
<tr>
<td>FBK-HLT</td>
<td>47.32</td>
<td>69.13</td>
<td>8</td>
</tr>
<tr>
<td>GBDT</td>
<td><strong>46.9</strong></td>
<td><strong>68.12</strong></td>
<td>9</td>
</tr>
<tr>
<td>Random Forest</td>
<td>46.74</td>
<td>65.89</td>
<td>10</td>
</tr>
<tr>
<td>SVM</td>
<td>43.35</td>
<td>43.35</td>
<td>11</td>
</tr>
<tr>
<td>Voltron</td>
<td>46.07</td>
<td>62.35</td>
<td>12</td>
</tr>
<tr>
<td>CICBUAPnlp</td>
<td>40.40</td>
<td>53.74</td>
<td>13</td>
</tr>
</tbody>
</table>

Table 8: Result of Task B.

<table>
<thead>
<tr>
<th>Model</th>
<th>Macro F1 (%)</th>
<th>Accuracy (%)</th>
<th>Ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>25.0</td>
<td>60</td>
<td>–</td>
</tr>
<tr>
<td>VectorSlu</td>
<td>63.7</td>
<td>72.0</td>
<td>1</td>
</tr>
<tr>
<td>GBDT</td>
<td><strong>59.14</strong></td>
<td><strong>65.59</strong></td>
<td>2</td>
</tr>
<tr>
<td>ECNU</td>
<td>55.8</td>
<td>68.0</td>
<td>3</td>
</tr>
<tr>
<td>QCRI</td>
<td>53.6</td>
<td>64.0</td>
<td>4</td>
</tr>
<tr>
<td>HITSZ-ICRC</td>
<td>53.6</td>
<td>64.0</td>
<td>5</td>
</tr>
<tr>
<td>Random Forest</td>
<td><strong>52.86</strong></td>
<td><strong>65.52</strong></td>
<td>6</td>
</tr>
<tr>
<td>SVM</td>
<td><strong>47.34</strong></td>
<td><strong>58.63</strong></td>
<td>7</td>
</tr>
<tr>
<td>CICBUAPnlp</td>
<td>38.8</td>
<td>44.0</td>
<td>8</td>
</tr>
<tr>
<td>ICRC-HIT</td>
<td>30.9</td>
<td>52.0</td>
<td>9</td>
</tr>
<tr>
<td>FBK-HLT</td>
<td>27.8</td>
<td>40.0</td>
<td>10</td>
</tr>
</tbody>
</table>

Our methodology is quite straightforward in its combination of selected features and use of specific models to train the data. We then conduct differential selection from the features and models. In brief, all the evaluation results are better than the baseline by the indicators Macro F1 and Accuracy and achieve the good performance. Although, our results did not overcome the general average, especially such as JAIST[21], HITSZ-ICRC[22], and QCRI[23] (top three). Evaluation on the GBDT model is clearly better compared with that on the SVM and random forest models.

Table 8 summarizes the results of our submitted runs on Task B datasets officially released by the organizers; the top rank runs are also provided. Each question is given Yes, No, and Unsure labels. The reported results also include the results of the baseline model and the best result. All of the evaluation results demonstrate the good performance of the proposed method.

In this experiment, the results of the GBDT and random forest models surpass the general average (47.54%). The rank of the GBDT model is 2. These results indicate that the random forest and SVM models can show reasonable performance in the Yes and Unsure classes but are unable to obtain the No class. Moreover, most of the instances of the No class are misclassified to the Unsure class.

By analyzing features and model building, we believe the experiment can also be improved in the following areas:

1. This experiment only uses the traditional machine learning method and does not apply deep learning algorithms, such as Convolutional Neural Networks (CNNs), which may show good performance.
2. This experiment only randomly selects some of the super-parameters of the model, which are not fine-grained. Ultra-parameter selection options can be analyzed in future work.
3. Only 20 features were selected in this study; more features could be added in later studies.

Despite these limitations, the experimental results show that our approach is efficient, and its overall rank is fairly high compared with those of other methods.

6 Related Work

Several works have focused on the quality of answers on CQA websites[31–33], which is greatly valuable for information retrieval. Prior studies on answer selection generally treat the task as a classification problem, that relies on exploring various features to represent the QA pair. A number of researchers have attempted to explore and determine various features that define question quality[16, 34–36]. In Ref. [37], for example, the authors designed specific features and applied structure prediction models. The authors of Ref. [38] used various types of features including 8 similarity features, 44 heuristic features, and 16 thread-based features. Although these methods achieved good performance, they also rely heavily on feature engineering, which requires a large amount of manual work and domain expertise. In Ref. [39], the authors integrated textual with structural features to represent the candidate pairs and then applied SVM to classify the candidate pairs. In Ref. [40], the authors used a GBDT as the classifier, and built ensembles from different boosted decision
tree models to improve prediction accuracy. In Ref. [41], an approach based on conditional random fields was proposed; this approach could capture contextual features for semantic matching between question and answer. Similarly, we identified effective features in content dimension way. Furthermore, we applied SVM, GBDT, and random forest classifiers to select high-quality results with a ranking function.

Some studies have exploited syntactic tree structures to measure the semantic correspondence between question and answer, such as Refs. [42, 43]. In Refs. [44, 45], the authors proposed directly learning the distributed representation of QA pairs. The work in Ref. [46] demonstrated that 2D convolutional sentence models can represent the hierarchical structures of sentences and capture rich matching patterns between two language objects. In Ref. [47], the authors formulated answer selection as a semantic matching problem with a latent word-alignment structure and conducted a series of experimental studies on leveraging proposed lexical semantic models. One disadvantage of these approaches is that semantic correlations among answers and questions, which are very important for answer selection, are ignored.

The translation-based language model has also been used for QA matching by transferring the answer to the corresponding question [48, 49]. However, this method suffers from informal words or phrases in Q&A archives and shows limited applicability in new domains. In Ref. [49], a retrieval model that combines a language model for the question part with a query likelihood approach for the answer part was proposed.

Recently, deep learning-based approaches have recently been applied. In Ref. [44], for example, the authors proposed a deep belief net-based semantic relevance model to learn the distributed representation of QA pairs. The CNN-based sentence representation models have achieved successes in neural language processing tasks. In Ref. [50], the authors applied CNNs to learn the joint representation of QA pairs and then used the joint representation as inputs of the Long Short-Term Memory algorithm to learn the answer sequence of a question and label the matching quality of each answer. In Ref. [51], the authors trained a logistic regression classifier with user metadata to predict the quality of answers; this approach may show good performance, and we will consider employing it in the next phase of our research.

7 Conclusion

Answer selection in CQA is a challenging task in natural language processing. In this paper, we enhanced answer selection in CQA using multi-dimensional feature combination including two subtasks: (a) classifying answers as good, bad, or potentially relevant with respect to the question and (b) answering a YES/NO question based on the list of all answers. Two methods are proposed: using traditional QA based on the similarity sorting method and using the information of the answer to the question to find similarities between question and answer. We then use the answer text description information to judge whether the answer is reasonable. We first extract the attribute and content information from each question and comment from the data set. Then, we build models from these features and use SVM, GBDT, and random forest to classify them. The experimental results show that our approach is significantly more efficient than the baseline model, and its overall ranking is relatively high compared with those of other methods.

In the feature, we will use other models, such as neural networks, for experiments on model building and logistic regression to train the dataset. In addition, strong dependencies were observed among the different answers to the same problem.

Acknowledgment

This research was developed by the NLP601 group at School of Electronics Engineering and Computer Science, Peking University, within the National Natural Science Foundation of China (No. 61672046).

References

Hongjie Fan et al.: Enhanced Answer Selection in CQA Using Multi-Dimensional Features Combination


Hongjie Fan received the master degree from Peking University in 2010. He is working toward the PhD degree at Peking University. His research interests include data exchange technology, schema matching, etc.

Zhiyi Ma is an associate professor at Peking University. His research interests include software modeling technology and model driven development. He received the PhD degree from Northeastern University, China, in 1999.

Hongqiang Li received the bachelor degree from China University of Geosciences, Beijing, in 2015. He is working toward the master degree in software engineering at Peking University. His research interests include information retrieval, machine learning, etc.

Dongsheng Wang received the bachelor degree from Northeastern University, China, in 2016. He is working toward the master degree in software engineering at Peking University. His research interests include data exchange technology, information retrieval, etc.

Junfei Liu is a professor and PhD supervisor at School of Electronics Engineering and Computer Science, Peking University. His research interests include software engineering, information exchange technology, and smart city. He received the BS degree in 1985 from Hunan University, China, and completed the MS degree in National Defense University, China, in 1988. Then, he earned PhD degree from Peking University, China, in 1994.