Answer Quality Assessment in CQA Based on Similar Support Sets

Zongsheng Xie[1], Yuanping Nie[1], Songchang Jin[1], Shudong Li[1,2] and Aiping Li[1]

[1] School of Computer Science,National University of Defense Technology, Changsha 410073, China
[2] College of Mathematics and Information Science, Shandong Institute of Business and Technology,
Yantai, Shandong 264005, China
xiezongsheng,yuanpingnie@nudt.edu.cn
jsc04@126.com
lishudong@126.com
apli1974@gmail.com

Abstract. Community question answering portal (CQA) has become one of the most important sources for people to seek information from the Internet. With great quantity of online users ready to help, askers are willing to post questions in CQA and are likely to obtain desirable answers. However, the answer quality in CQA varies widely, from helpful answers to abusive spam. Answer quality assessment is therefore of great significance. Most of the existing approaches evaluate answer quality based on the relevance between questions and answers. Due to the lexical gap between questions and answers, these approaches are not quite satisfactory. In this paper, a novel approach is proposed to rank the candidate answers, which utilizes the support sets to reduce the impact of lexical gap between questions and answers. Firstly, similar questions are retrieved and support sets are produced with their high quality answers. Based on the assumption that high quality answers of similar questions would also have intrinsic similarity, the quality of candidate answers are then evaluated through their distance from the support sets in both aspects of content and structure. Unlike most of the existing approaches, previous knowledge from similar question-answer pairs are used to bridged the straight lexical and semantic gaps between questions and answers. Experiments are implemented on approximately 2.15 million real-world question-answer pairs from Yahoo! Answers to verify the effectiveness of our approach. The results on metrics of MAP@K and MRR show that the proposed approach can rank the candidate answers precisely.

1 Introduction

In the age of web 2.0, people can easily publish on the World Wide Web in all kinds of systems and modes, such as twitter, facebook, blogs, online discussion forums, wikis and question answering sites. In most of these platforms, the information is represented in natural language form rather than more structured formats, and most people involved are in equal positions to share experience and express opinions. Consequently, the amount of user-generated content (UGC) available on the Web is conspicuously increasing which constitutes an important source of information. However, it is the lack of editorial control that makes the information quality on the Web vary dramatically from professional to abusive.

Community question answering services, such as Yahoo! Answers (Y!A)[1], Baidu Knows[2] and Quora[3] , are typical UGC portals that have become very popular in recent years. In CQA services, people post problems they encountered at work and in daily life and seek for help. Fellow users who know the answers or have similar experiences would reply their opinions, sometimes under the incentive mechanism of the site. Usually, the content in the site is organized as questions and lists of answers associated with metadata such as user votes and askers awards to the best answers. Attracted a great number of users, these web sites have become hot platforms for people to seek help, share knowledge, and learn from each other. What is more, with the accumulation of question-answer pairs over time, this data in CQA archives becomes valuable repositories of information and knowledge.

Although CQA service has brought significant benefits for us to solve daily problems and seek information, there are still some drawbacks in most of the CQA systems. As a type of UGC portal, one of the most important problems is the quality of the answers. An example of question with answers of various quality is given below:

[1] http://answers.yahoo.com
[2] http://zhidao.baidu.com
[3] http://www.quora.com

- **question:** What are the best techniques to deal with stress?
- **answers 1:** When dealing with stressful situations, consider the four points below.
    1. Nothing and no one can make you feel anything. How you feel and the way you deal with a situation is a choice. Im reminded of a counselor who would often state no one can drive your car unless you give them the keys. You cannot control others actions, but you can be responsible for your reactions.
    2. Exchange attitude for gratitude. Our attitude has a profound effect on how we deal with situations. Negative attitudes affect our physical, spiritual, and mental well being.
    3. Relax, relax, relax. Amidst the hustle and bustle of everyday life, sometimes we forget to take care of ourselves. If we do not help ourselves, how can we effectively help others? Relaxation rejuvenates the body, mind, and spirit and leaves us better equipped to handle stressful situations when they come.
    4. Look at the big picture. Evaluate your stressful situation from a big picture point of view. Ask yourself how important is this? and will this matter in the long run? If the answer is no, its likely not worth your time and energy. OR you can get online consultation at helpingdoc.com
- **answers 2:** When i am so stressed out, i usually do exercise, go for walks and meditate and listen to music. Thats how i manage stress!
- **answers 3:** dont refuse help from friends

As can be seen from the above, the quality of the answers range from very high to low, or even abusive. As previously noted [1], the quality of answers in CQA portals is good on average, but the quality of specific answers varies significantly. According to the author's study on the answers of a set of questions in Yahoo! Answers, the fraction of correct answers to specific questions varied from 17% to 45%, while the fraction of questions with at least one good answer was much higher, varying from 65% to 90%. This study suggests that most of the questions are expected to obtain a good answer, but are also likely to get low quality ones. Methods to find high quality answers therefore can have a significant impact on the users satisfaction of the system. Besides, the efficiency of solving new problems is often not quite commendable in CQA. As shown in the study [2], more than 80% new questions cannot be solved within 48 hours. Recommending similar questions with satisfactory answers would be an advisable solution to remedy the situation. In this case, the answer quality assessment is also very meaningful. Though CQA sites have provided many mechanisms to find high quality answers, such as thumb up and thumb down voted by viewers and the best answer award voted by askers, this shortcoming still exists since such feedback requires some time to accumulate, and often remains sparse for obscure or unpopular topics [3].

It is common to analyze answer quality in CQA through superficial features, such as user voting, user or editor recommendations, and the metadata of users who provide the answers. There are several common problems with the approaches utilizing these popularity and social interaction measures to predict the answer quality. Firstly, this information is not always available in real world applications. For instance, there is no voting or recommendation available for new posted answers, and no metadata available for new users or visitors without login. Secondly, there is no necessarily causal relationship between user metadata and answers the he posts, since an expert user associated with lots of good answer voting and recommendations may not be good at answering all questions, while many good answers are provided by common users whose metadata may not be indicative of the quality of their answers. It is also a familiar way to measure answer quality by calculating textual features, such as the length of answers, overlapped words between questions and their answers, length ratio between questions and answers [4, 5]. However, the lexical gap between questions and answers is usually very large, which makes these approaches powerless.

In this paper we propose two hypotheses: the the lexical gap between similar questions and that between their answers are much smaller than the gap between questions and their answers; high quality answers of similar questions should also share some common intrinsic features. Based on these two hypotheses we propose a novel approach to evaluate the answer quality. Unlike most of the existing approaches, we do not calculate relevance between questions and answers directly, instead we use previous knowledge from similar question-answer pairs to bridge the lexical gap. Utilizing the support sets, the impact of the straight lexical gap between questions and answers is reduced effectively. The results of experiments on approximately 2.15 million real-world question-answer pairs from Yahoo! Answers suggest that our hypotheses are meaningful indeed, and our approach can rank the candidate answers precisely.

2 Related Work

Providing popular platforms for people to seek solutions to problems and share opinions, CQA portals have drawn a great number of users in the last decade. A great deal of attentions from researchers are also attracted in related fields, such as investigating information seeking behaviours [6], user motivations [7], expert recommendation [8, 9], question retrieval [3, 8] and answer quality assessment [4, 10, 11]. In this section, we will discuss the latter two, which are relevant to our study.

2.1 Question Retrieval

Content from community-built question-answer sites can be retrieved by searching for similar question already answered, and the task of question retrieval is to find relevant question-answer pairs for new questions posed by users in the QA archive [12]. The issue of question retrieval was first raised in the field of frequently asked questions (FAQs). Burke et al. [13] produced a FAQ finder which combined statistical similarities and semantic similarities between questions to rank FAQs. With the flourishing of CQA, more attention is paid to question searching in this field recently. The major challenge for question retrieval in CQA is the word mismatch between new question and the question-answer pairs in the archive, which is similar to most information retrieval tasks. To solve this problem, many different approaches have been proposed. Based on the assumption that the relationship between words can be modeled through word-to-word translation probabilities, many researchers adopted translation-based approaches [14] to solve the word mismatch problem. Jeon et al. [12] proposed a word-based translation model to fix the lexical gap problem. Xue et al.[15] combined the query likelihood language model with the classic IBM translation model 1. Cai et al. [16] assembled the semantic similarity based latent topics with the translation-based language model. Besides, some other methods improved the traditional language models by leveraging metadata in CQA. The language model by Cao et al. [17, 18] estimated new smoothing item with leaf category smoothing. Zhang et al. [8] proposed a topic-based approach to match questions on both term level and topic level.

Unlike the normal question retrieval, we are not aiming at finding relevant question-answer pairs for new questions to improve user experience, but finding questions similar in content and in structure which is a preparation for our model. We simply find similar questions through the overlap of words with different weights.

2.2 Answer Quality Evaluating

A range of approaches have emerged for evaluating answer quality in CQA, which can mainly be divided into two categories. The first kind of method is based on content analysis. These methods mainly assess the answer quality through the relevance between answers and questions. Toba et al. [4] classified questions into several types, and implemented a type-based quality classifier to predict answer quality of different types of questions with different groups of content features. Surdeanu et al. [19, 20] built a answer ranking engine for non-factoid questions combining several strategies into a single model, such as question-to-answer transformations, frequency and density of content. They also expanded them with large amounts of available Web data. The second kind of methods mainly utilized user information to estimate the quality of the answers they posted. There are generally two kinds of user information that are usually used: the user log information and the linked graphs of users. User log analysis approaches use past performance of users to measure the popular and expert degree of the user [21, 22], such as the number of best answers they posted, user voting and recommendations. These methods supposed that the reputed users are more likely to give high quality answers. Methods analyzing linked graphs of users, which consist of users as vertices and the interactions of ask-answer as edges, usually employ link-based ranking algorithms such as HITs [23] and PageRank [24]. Exploiting the interactions between users, these methods assign active users who have posted more high quality answers with higher probabilities of giving good answers, such as ExpertiseRank by Zhang et al. [25] and CQARank by Yang et al. [11].

Despite the success of these approaches in many situations, there are some shortages in them. For example, the lexical gaps in some question-answer pairs may be very large where the content analysis approaches will be incapable to evaluate their relevance precisely, and the approaches utilizing user information tend to recommend expert users and may fail when facing new users or visitors without login whose user information is not available. In our approach, different from the above approaches, the impact of large lexical gap between questions and answers is reduced by utilizing previous knowledge from the similar question-answer pairs, and we also don't need to analyze user profiles.

The method most close to ours is [26]. However, they used the vector of textual and non-textural features to represent question-answer pairs, and mainly trained a bayesian logistic regression to predict answer quality, which are different from our method.

3 The Approach

3.1 Process Overview

Our approach is based on two hypotheses:

The lexical gap between similar questions and that between their answers are much smaller than that between questions and answers. Most of state-of-the-art methods evaluate the answer quality in CQA through the relevance between questions an answers. Unfortunately, through a large number of observations we found that, in most instances the lexical and semantic distances between questions and their answers were very considerable. On the other hand, given a new question, we could find some similar questions in the CQA archive having the gap between them much smaller, with their answers quite close to the candidate answers too. There are two samples of questions and corresponding answers extracted from Yahoo! Answers:

- **question 1:**
  I'm 13 how can I keep fit ?
- **answers 1:**
  - Any and all exercise helps, ideally a mix of resistance and cardiovascular exercise. Jogging is excellent.
  - Walking/jogging Weight lifting Kettle bell (one kettle bell can allow for many exercises). My opinion is any weight lifting exercise is good for a young person. It builds strength, balance, and bone strength. It also forms a very nice body for when you get to the point to show it off–believe me.
  - exercises like pull ups or sit ups work for U and push ups
  - Swimming, Walking and riding a bike.

- **question 2:**
  How can I keep fit at a young age?
- **answers 2:**
  - Getting back into running can be very good for you physically and emotionally.
  - Running in place- self explanatory.
  - just regular basic cardio exercises like going for a run or cycling can keep you in pretty good shape

As we can see in that two samples, few meaningful words are shared by the questions and their answers. It is obviously inadvisable to evaluate the quality of answers through the relevance between question and answer in this case. However, if we take question 2 as the similar question to question 1, we could find that the lexical gap between them is much smaller, while the answers of them also share lots of common features in semantics and structure. Based on this discovery, we suppose that it would be more effective to evaluate answer quality utilizing previous knowledge from similar questions and their answers than through the relevance between questions and answers directly.

Similar questions will have similar answers, and high quality answers of similar questions will also share some common traits. As demonstrated by previous studies [27, 28], there are some common intrinsic features that are shared by high quality answers. Intuitively, questions asking about similar content should have answers similar in content, and questions similar in structure should have answers similar in structure, too. For example, high quality answers of factid questions asking for same thing should also talk about same object, and good answers of why-type questions may share same syntax of adverbial clause of cause. As the questions above, answers of them are usually talking about going on a diet or doing some exercise, and high quality ones should tell how to do that in detail.

Base on these two hypotheses, we propose a novel model to rank the candidate answers. The sketch of our model is shown in Figure 1. The processes of our approach are mainly as follow:

In the offline stage, we build a support base which is made up of a large number of question-high quality answer pairs. The best answers are used as the high quality answers in our method, because they are voted by askers which means that they are satisfactory, therefore supposed to be of high quality. The support base is built on the search platform of Solr [4] , which is powerful in indexing

[4] http://lucene.apache.org/solr

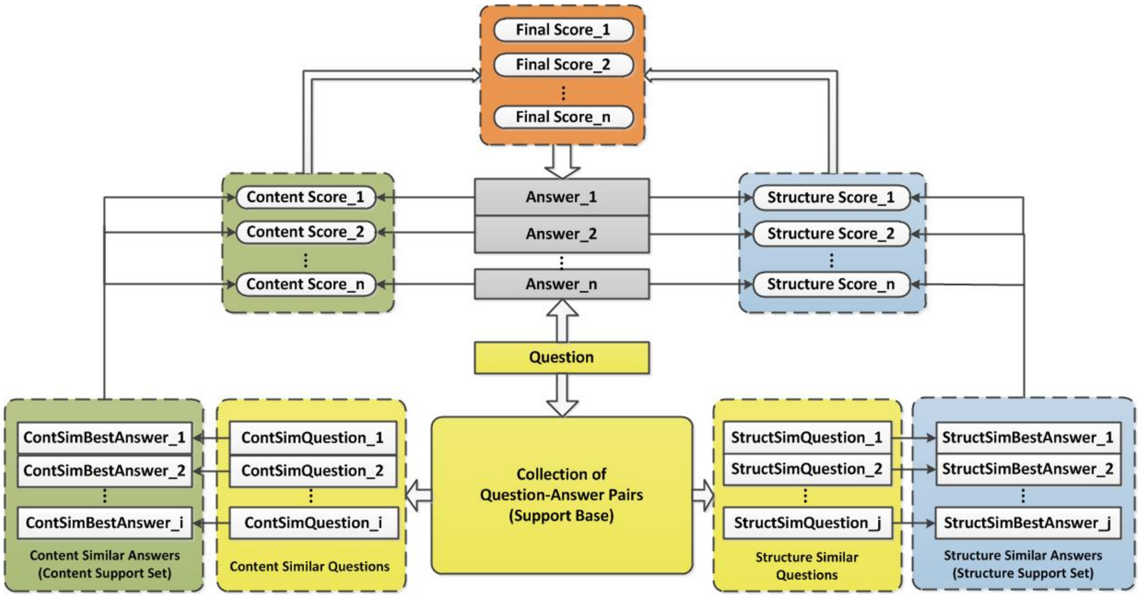

Fig. 1. Sketch of the our approach: a) Given a question as a query, get content similar questions and structure similar questions from the support base (yellow blocks in the figure); b) Get the corresponding best answers as the CSS and SSS, and count the content score and structure score of every candidate answers through the similarity with CSS and SSS (green blocks and blue blocks); c) Integrate the content score and structure score into the final score (orange block) ; d) Rank the candidate answers according to the final score.

and searching. The question-best answer pairs of the support base are stored and indexed, and can be queried through the search engine of Solr.

In the online stage, a new question is treated as a query, and two sets of similar questions and corresponding best answers (similar in content and similar in structure, which will be described in detail in section 3.2) will be returned from support base. These two parts of best answers will be used as our content support set (referred to as CSS) and structure support set (referred to as SSS) in our model. For every candidate answer of the question we are considering, we score it base on its content and structure similarity with the support sets. Then these two scores will be integrated into a final score. Finally, All the candidate answers will be ranked through their final scores, and the top-ranked answer is supposed to be the best one.

3.2 Obtaining Support Sets From Support Base

To obtain the CSS and SSS, we will search for two different kinds of similar questions from the support base: content similar questions and structure similar questions. Their corresponding best answers are then extracted to form CSS and SSS.

Obtaining content support set The key issue when searching for content similar questions from the support base is how to measure the relevance between the query and the documents. TF-IDF relevancy model is chosen in our approach, which combines boolean model (BM) of information retrieval with vector space model (VSM) of information retrieval. The formula of TF-IDF relevance scoring model is:

$$Score_{tf/idf}(q,d) = coord(q,d) * \sum_{t \in q} (tf(t \in d) * idf(t)^2 * boost(t)) \quad (1)$$

where q is the query; d is the the document to be scored; t is the terms appear in the query; $coord(q,d)$ is a score factor based on how many the query terms are found in the specified document. $tf(t \in d)$ is the frequency of terms; $idf(t)$ stands for inverse document frequency; $boost(t)$ is a search time boost of term t in the query q, which is specified by user in the query text. The formula of $idf(t)$ in our model is:

$$idf(t) = 1 + \log \frac{numDocs}{docFreq + 1} \quad (2)$$

where *docFreq* means the number of documents containing the term t, and *numDocs* is the number of documents to be scored.

The points of the TF-IDF relevance model is that:

- The more terms a document contains, the higher the score;
- The more times a term appears in a document, the higher the score;
- Rarer terms, which is supposed to be more discriminating, give higher contribution to the score;
- The higher weight a term is specified, the higher it contribute to the score.

Before querying, the sentence of the question will be parsed with the Stanford POS Tagger [29], which generates a syntactic parse tree of the sentence. Then the real words, i.e. nouns and adjectives in our model, are extracted and specified with a higher weight in the query text, which makes the content contribute higher in the query. Finally, we search the support base with the query text, and get a set of questions with similar content and a set of corresponding best answers, i.e. the CSS.

Obtaining structure support set Similar to the way the CSS is obtained , the SSS is extracted from the support base with the same score model and the same syntactic parse tool. The difference is that, after the syntactic parse tree is generated, the real words are removed from the query with the skeleton frame of the question left. An example of the skeleton frame extracted from the question is as below:

- **Question:** What is more important, love or money, why?
- **Tagged sentence:** What_WP is_VBZ more_RBR important_JJ ,_, love_NN or_CC money_NN ,_, why_WRB ?_.
- **Skeleton frame:** What is more , or, why?

With the query of skeleton frame, we will get a set of questions similar in structure, and their responding best answers, i.e. the SSS.

3.3 Scoring The Candidate Answers

In this part, we will score the candidate answers leveraging the similarity between them and the support sets. Corresponding to the two parts of support sets, the candidate answers will also be scored in two different angles.

Scoring in content Firstly, the candidate answers will be scored in the aspect of content based on the similarity between them and the CSS. We tried several different scoring methods: Cosine similarity, DRF similarity, TF-IDF model, and BM25 model, and the last one which performed the best in our experiment was finally chosen. The main idea of the BM25 model is to analyze the similarity of every term in the query, and count their weighted sum:

$$Score_{bm}(q,d) = \sum_{i}^{n} W_i * R(t_i, d) \tag{3}$$

where q is the query and t_i is a term in the query; W_i is the weight of t_i. The IDF of the term is used as weight in our model; d is the document; $R(t_i, d)$ is the similarity between the term t_i and the document d:

$$R(t_i, d) = \frac{f_i * (k+1)}{f_i + k * (1 - b + b * \frac{dl}{avgdl})} \tag{4}$$

where k and b are the regulative factors, which are generally specified according to experience and is set as $k = 2$ and $b = 0.75$ in our model; f_i is the frequency of t_i in d; dl is the length of d, and $avgdl$ is the average length of all documents. Consequently, the formula of score in the BM25 model is:

$$Score_{bm}(q,d) = \sum_{i}^{n} IDF(t_i) * \frac{f_i * (k+1)}{f_i + k * (1 - b + b * \frac{dl}{avgdl})} \tag{5}$$

Notice that, to every candidate answer A_i and similar best answer d_i^j pair, there will be a score calculated to measure the similarity between them. Unlike the normal query systems, we do not rank the documents according to the score, but calculate the average score of every candidate answer, which is supposed to be the similarity of the candidate answer A_i and its whole CSS:

$$Score_{css}(A_i) = \frac{1}{\mid CSS_i \mid} \sum_{d_i^j \in CSS_i} Score_{bm}(A_i, d_i^j) \tag{6}$$

where CSS_i is the CSS corresponding the candidate answer A_i; $\mid CSS_i \mid$ is the size of the CSS_i.

Scoring in structure In the process of scoring in structure, we follow the approach used in many previous studies[4, 27, 19] that to quantify the properties of the answers by extracting and calculating representative features from the question-answer pairs. The features used in our method can be generally divided into two categories:

- **Numeric features:** It is supposed that the answers of questions similar in structure will share some common numeric features, such as the number of sentences, the number of nouns, verbs and adjectives and so on.
- **Ratio features:** As the answers of similar questions may have different length, the ratio features are taken into consideration, such as the ratio of the length of answer and question, and the ratio of nouns, verbs, adjectives in the answer.

All the features and their explanations are listed in Table 1.

Table 1. Features used to score in structure

Features	Explanation
aLength	Length of answer
aNumNoun	Number of nouns in answer
aNumVerb	Number of verbs in answer
aNumAdj	Number of adjectives in answer
aNumSent	Number of sentences in answer
aRatioNoun	Ratio of nouns in answer
aRatioVerb	Ratio of verbs in answer
aRatioAdj	Ratio of adjectives in answer
qaRatioSent	Ratio of sentences in question and answer
qaRatioLen	Ratio of question length and answer length

These features are extracted from the best answers in the SSS, and their average value is counted as the representative. On the other hand, the features from the candidate answers are also extracted, and their distance with the representative one are calculated. Notice that, a smaller distance means the feature of the candidate answer is more close with the representative. Then the candidate answers are ranked base on the distance where the smaller the distance, the closer it is to the top. A matrix of the ranks is generated as the result:

$$\begin{bmatrix} R11 & R12 & ... & R1m \\ R21 & R22 & ... & R2m \\ ... & ... & ... & ... \\ Rn1 & Rn2 & ... & Rnm \end{bmatrix} \tag{7}$$

where the element of Rif means the rank of candidate answer i on the feature f. In order to integrate all the features and get rid of the difference between dimensions, the sum of the inversion rank is used as the integrated score:

$$Score_{sss}(A_i) = \sum_{f=1}^{m} \frac{1}{Rif} \tag{8}$$

where m is the number of features we extracted.

Getting final score After getting both the scores in content and in structure, we combine them to get the final score with the same method mention in the above section:

$$Score_{final}(A_i) = \frac{1}{Rank(CSS)} + \frac{1}{Rank(SSS)} \tag{9}$$

where $Rank(CSS)$ is the rank based on the content score, and $Rank(SSS)$ based on the structure score.

Finally, the candidate answers are ranked depending on this final score, and it is supposed that the closer a answer is to the top, the higher its quality is.

4 Experiment

4.1 Dataset

In the WebScope Program of Yahoo! Research[5], there are several datasets available to researchers. The dataset of Yahoo! Answers Comprehensive Question and Answers is one of them, which is collected from the website of Yahoo! Answers, and covers all categories in Yahoo! Answers. Our dataset used in this experiment is extracted from the above corpus, containing full text of 2,258,383 questions and their answers, including the best answers voted by the askers to each question.

From our dataset, we randomly extracted 1,787,975 questions and their best answers to build the support base. We stored them into database, and created full text indexes of their questions, with the search platform of Solr. When creating the indexes and searching in the support base, we implemented the TF-IDF similarity model described in section 3.2 to rank the correlation of the questions. The other 497,408 questions left with their all candidate answers were used as the testing set. Statistics about our dataset are synthesized in Table 2

Table 2. Statistics about dataset

Dataset	Number of questions	Number of answers	Average number of answers
Support base	1,787,975	1,787,975(best answers)	-
Testing set	497,408	2,148,802	4.32

4.2 Evaluation Metrics

In our experiment, best answer tagged by the askers in Yahoo! Answers was used as the ground truth, i.e. if our approach found the best answer from all the answers, it was supposed to be correct, otherwise incorrect. In order to measure the performance of our approach, two metrics were used for the evaluation: Mean Average Precision at K (MAP@K) and Mean Reciprocal Rank (MRR). These metrics are commonly used to measure the accuracy of ranked retrieval results.

For a given query, the metrics of Average Precision at K is the mean fraction of relevant answers ranked in the top K results:

$$AP@K = \sum_{i=1}^{K} P(i)/min(m, K) \tag{10}$$

where P(i) means the precision at cut-off i in the item list, m is the number of relevant results returned by the rank system. The MAP@K is the mean value of the average precision:

$$MAP@K = \frac{1}{n}\sum_{i=1}^{n} AP_i@K \tag{11}$$

where n is the number of the questions in our testing set.

The Mean Reciprocal Rank (MRR) metric take the exact rank of a correct answer into account and the score is counted as the mean of the reciprocal rank:

$$MRR = \frac{1}{\mid Q \mid}\sum_{q \in Q}\frac{1}{r_q} \tag{12}$$

where Q is the set of the testing queries; r_q is the rank of correct answer for the query q.

As mentioned in the section 2.2, most of state-of-the-art approaches assess the answer quality though the relevance between questions and answers. We therefore compared our model with methods calculating these features. Two baselines were used as comparisons: the method based on Cosine Distance Metric (COS), and the method based on Linear Regression Prediction(LR). The method of COS measured the answer quality though the cosine distance between questions and answers. The approach of LR extracted textual features from questions and answers, and used the method of linear regression to predict the quality of answers. As is commonly used in many state-of-the-art approaches [4, 19], we extracted 15 textual features to represent a question-answer pair. In addition to the 10 features listed in Table 1, the other 5 are shown in Table 3. The linear regression model was trained and tested with our dataset in the proportion of approximately 70% : 30% of training : testing.

[5] http://webscope.sandbox.yahoo.com

Table 3. Portion of the textual features used in LR

Features	Explanation
qaTF	Term frequency of question answer pair
comStrLen	Length of common string
resemblance	Proportion of the set of overlapping n-grams, and the set of all n-grams for the question and its answer
containment	Proportion of n-grams from the answer that also appear in the question
cosDist	Cosine distance between a question and its answer

4.3 Performance

Overall Results The performance of our model and the baselines on the MAP@K metric when K is set as 1, 5 and 10 are illustrated in Figure 2, while Table 4 shows the performance on the metrics of MRR. As is illustrated in the figure and the table, the method of LR worked better than COS, and our method significantly outperforms the baselines on the metric of MAP@K in all cases when K is 1, 5 and 10, as well as on the metric of MRR. The performance of COS which is the worst in the result, tells that the lexical gap between questions and answers is usually very large, and it's not advisable to assess the answer quality through the word overlap between questions and answers directly. The better performance of LR suggests that high quality answers do share some common traits. The outperformance of our method on both MAP@K and MRR proves that our hypotheses are significative indeed, and our approach can rank the candidate answers more precisely.

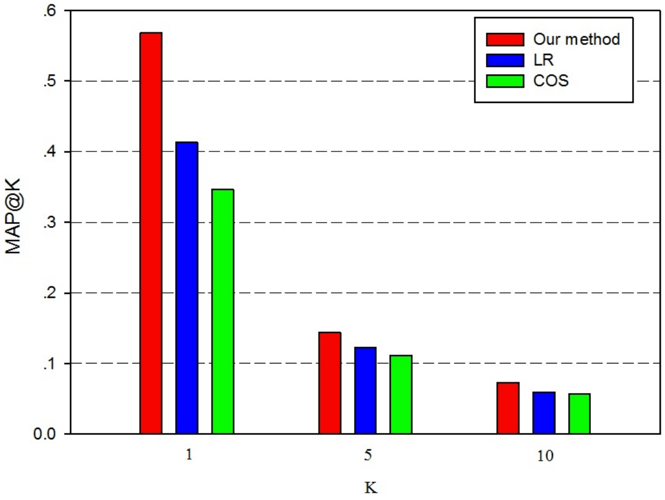

Fig. 2. MAP@K when K = 1, 5, 10

Table 4. MRR for baselines and our model

	COS	LR	Our Method
MRR	57.2%	66.4%	72.8%

Contribution of CSS and SSS In order to investigate the effectiveness of CSS and SSS, we conducted experiments assembling scores of CSS and SSS with different proportion. As mentioned in Section 3.3, we used reciprocal rank of CSS and SSS to count overall score:

$$Score_{final}(A_i) = \frac{\lambda}{Rank(CSS)} + \frac{1-\lambda}{Rank(SSS)} \tag{13}$$

where λ is the parameter to control proportion. Notice that when $\lambda = 0$, only the the score of SSS is valid, and when $\lambda = 1$ CSS only. When λ ranges from 0 to 1, the results are shown in Figure 3, which illustrates that we get the best result at both the metrics of MAP@1 and MRR when $\lambda = 0.5$.

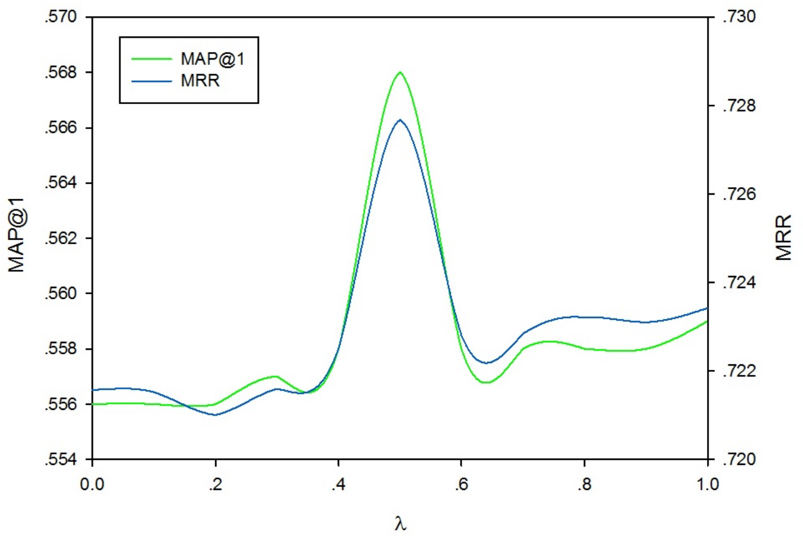

Fig. 3. MAP@1 and MRR with different λ

5 Conclusion

In this paper, we proposed a novel approach to evaluate the answer quality in CQA. We assumed that: the lexical gap between similar questions and that between their answers is much smaller than that between questions and answers; and high quality answers of similar questions will also share some common traits. We built a support base which was made up of a large number of question-high quality answer pairs in the offline stage. In the online stage we obtained content support set and structure support set for every candidate answer and measured their quality utilizing the similarity between them and the high quality answers in the support sets. Unlike most of state-of-the-art methods, we did not analyze the relevance between questions and answers directly, but used previous knowledge from the similar question-answer pairs to bridge the lexical gap. The experiment on dataset from real-world question-answer pairs from Yahoo! Answers showed that, our model ranked the candidate answers more precisely on both metrics of MAP@K and MRR. The comparison with the baselines suggested that taking previous knowledge into account is advisable when assessing the quality of noisy user-generated content in CQA.

This work can be extended in several directions. First, rather than treated equally, the similar answers in the support sets could be of different weights when counting the the candidate answer scores, according to the similarity between their corresponding questions and the original question. In addition, the value of λ, which regulates the score weights of two different support sets, could be dynamically adjusted based on the reliability of different support sets. Another interesting and meaningful problem is the methods to measure the similarity between questions and that between answers, both in content and structure. It could be beneficial to develop some more accurate method in this domain.

6 Acknowledgement

The work described in this paper was supported by National Natural Science Foundation of China (No.61202362), National Key Basic Research Program of China (NO. 2013CB329606) and Project funded by China Postdoctoral Science Foundation (NO.2013M542560).

References

1. Qi Su, Dmitry Pavlov, Jyh-Herng Chow, and Wendell C Baker. Internet-scale collection of human-reviewed data. In *Proceedings of the 16th international conference on World Wide Web*, pages 231–240. ACM, 2007.

2. Baichuan Li and Irwin King. Routing questions to appropriate answerers in community question answering services. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1585–1588. ACM, 2010.
3. Guangyou Zhou, Kang Liu, and Jun Zhao. Joint relevance and answer quality learning for question routing in community qa. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1492–1496. ACM, 2012.
4. Hapnes Toba, Zhao-Yan Ming, Mirna Adriani, and Tat-Seng Chua. Discovering high quality answers in community question answering archives using a hierarchy of classifiers. *Information Sciences*, 261:101–115, 2014.
5. Jiang Bian, Yandong Liu, Eugene Agichtein, and Hongyuan Zha. Finding the right facts in the crowd: factoid question answering over social media. In *Proceedings of the 17th international conference on World Wide Web*, pages 467–476. ACM, 2008.
6. Soojung Kim, Jung Sun Oh, and Sanghee Oh. Best answer selection criteria in a social qa site from the user oriented relevance perspective. *Proceedings of the American Society for Information Science & Technology*, 44(1):1C15, 2007.
7. Chirag Shah, Jung Sun Oh, and Sanghee Oh. Exploring characteristics and effects of user participation in online social q&a sites. *First Monday*, 13(9), 2008.
8. Kai Zhang, Wei Wu, Haocheng Wu, Zhoujun Li, and Ming Zhou. Question retrieval with high quality answers in community question answering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 371–380. ACM, 2014.
9. Mohamed Bouguessa, Benoît Dumoulin, and Shengrui Wang. Identifying authoritative actors in question-answering forums: the case of yahoo! answers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 866–874. ACM, 2008.
10. Zhi-Min Zhou, Man Lan, Zheng-Yu Niu, and Yue Lu. Exploiting user profile information for answer ranking in cqa. In *Proceedings of the 21st international conference companion on World Wide Web*, pages 767–774. ACM, 2012.
11. Liu Yang, Minghui Qiu, Swapna Gottipati, Feida Zhu, Jing Jiang, Huiping Sun, and Zhong Chen. Cqarank: jointly model topics and expertise in community question answering. *Research Collection School of Information Systems*, 2013.
12. 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*, pages 84–90. ACM, 2005.
13. Robin D. Burke, Kristian J. Hammond, Vladimir Kulyukin, Steven L. Lytinen, Noriko Tomuro, and Scott Schoenberg. Articles question answering from frequently asked question files experiences with the faq finder system. *Finder System, AI Magazine*, 1997.
14. Adam Berger, Rich Caruana, David Cohn, Dayne Freitag, and Vibhu Mittal. Bridging the lexical chasm: statistical approaches to answer-finding. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 192–199. ACM, 2000.
15. Xiaobing Xue, Jiwoon Jeon, and W Bruce Croft. Retrieval models for question and answer archives. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 475–482. ACM, 2008.
16. Li Cai, Guangyou Zhou, Kang Liu, and Jun Zhao. Learning the latent topics for question retrieval in community qa. In *IJCNLP*, volume 11, pages 273–281, 2011.
17. Xin Cao, Gao Cong, Bin Cui, Christian S03ndergaard Jensen, and Ce Zhang. The use of categorization information in language models for question retrieval. *Association for Computing Machinery*, 2009.
18. Xin Cao, Gao Cong, Bin Cui, and Christian S. Jensen. A generalized framework of exploring category information for question retrieval in community question answer archives. *Www*, 2010.
19. Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. Learning to rank answers on large online qa collections. *In Proceedings of the 46th Annual Meeting for the Association for Computational Linguistics: Human Language Technologies (ACL-08: HLT*, pages 719–727, 2008.
20. Mihai Surdeanu, Massimiliano Ciaramita, Google Inc, and Hugo Zaragoza. Learning to rank answers to non-factoid questions from web collections. *Computational Linguistics*, 37(2):351–383, 2011.
21. Chirag Shah;Jefferey Pomerantz. Evaluating and predicting answer quality in community qa. *Proc.of Intl.conf.on Research & Development in Information Retrieval Sigir* , pages 411–418, 2010.
22. Jie Lou, Yulin Fang, Kai H Lim, and Jerry Zeyu Peng. Contributing high quantity and quality knowledge to online q&a communities. *Journal of the American Society for Information Science and Technology*, 64(2):356–371, 2013.
23. Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the Acm*, 46(5):604–632, 1999.
24. Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: bringing order to the web. *Stanford Infolab*, 1999.
25. Jun Zhang, Mark S Ackerman, and Lada Adamic. Expertise networks in online communities: structure and algorithms. In *Proceedings of the 16th international conference on World Wide Web*, pages 221–230. ACM, 2007.
26. Xin Jing Wang, Xudong Tu, Bei P. R. China Hu, and Lei Zhang. Ranking community answers by modeling question-answer relationships via analogical reasoning. *Sigir Proceedings of International Acm Sigir Conference on Research & Development in*, 2009.

27. Chirag Shah and Jefferey Pomerantz. Evaluating and predicting answer quality in community qa. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 411–418. ACM, 2010.
28. Daisuke Ishikawa, Noriko Kando, and Tetsuya Sakai. What makes a good answer in community question answering? an analysis of assessors criteria. In *Proceedings of the 4th International Workshop on Evaluating Information Access (EVIA), Tokyo, Japan.* Citeseer, 2011.
29. Kristina Toutanova and Christopher D Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics-Volume 13*, pages 63–70. Association for Computational Linguistics, 2000.