An Unsupervised Approach for Low-Quality Answer Detection in Community Question-Answering

Haocheng Wu[1(✉)], Zuohui Tian[2], Wei Wu[3], and Enhong Chen[1]

[1] University of Science and Technology of China, Hefei, China
ustcwhc@outlook.com,cheneh@ustc.edu.cn
[2] Harbin Institute of Technology, Harbin, China
zuohuitian@gmail.com
[3] Microsoft Research, Beijing, China
weiwu@microsoft.com

Abstract. Community Question Answering (CQA) sites such as Yahoo! Answers provide rich knowledge for people to access. However, the quality of answers posted to CQA sites often varies a lot from precise and useful ones to irrelevant and useless ones. Hence, automatic detection of low-quality answers will help the site managers efficiently organize the accumulated knowledge and provide high-quality contents to users. In this paper, we propose a novel unsupervised approach to detect low-quality answers at a CQA site. The key ideas in our model are: (1) most answers are normal; (2) low-quality answers can be found by checking its "peer" answers under the same question; (3) different questions have different answer quality criteria. Based on these ideas, we devise an unsupervised learning algorithm to assign soft labels to answers as quality scores. Experiments show that our model significantly outperforms the other state-of-the-art models on answer quality prediction.

Key words: community question answering, answer quality evaluation

1 Introduction

In the last decade, many community question answering (CQA) sites such as Yahoo! Answers and Baidu Knows have emerged and accumulated a large number of questions, answers, and users. The quality of answers may be high in the sense that the answers are precise and useful. However, it may be low in the sense that the answers are irrelevant to the topic and thus useless. It becomes an important problem how to detect low-quality answers in order to improve the experience of user when he browses a question and its answers (a QA thread).

One way to improve user experience in CQA is to provide the best answer for each QA thread. Many studies have been conducted along this line [7, 21, 23, 18]. However, we have found that many non-factoid questions do not have single best answers, especially those asking for reasons, instructions or opinions. Therefore, selection of one best answers may not satisfy the user needs in such cases. In the meantime, the existence of low-quality answers can seriously decrease user satisfaction. Therefore, there is a clear need to detect low-quality answers, which is the problem we want to address in this paper.

There are three main challenges in low-quality answer detection. (1) Manually labeled datasets are costly and hard to obtain, which is one of the bottlenecks of existing methods, since they are mainly based on supervised learning. (2) Existing methods usually focus on relevance measures between questions and answers. However, a low-quality answer usually uses the same key words with the question but talks on totally different and irrelevant topics. In this case, it is very hard to say that they are useless answers. (3) The answer quality criteria may vary on different questions. For example, for non-factoid questions long answers are quite common, while for factoid questions short answers are often sufficient. Existing supervised learning methods usually favor long answers because long answers tend to have good human judged labels in training datasets.

Table 1 shows an example[1] of a question with a low-quality answer. We find that most answers can answer the question and the high-quality ones (A3-A6) are quite similar. While, A1 is marked as the best answer in Yahoo! Answers, but it makes an impolite joke and has no similarities in terms of content to other answers. Thus, we can pick A1 as low-quality answer from the others by comparing the differences among them. From this example, we can infer that low-quality answers are usually outliers from all the other answers.

In this paper, we propose a new method for low-quality answer detection, on the basis of unsupervised learning. There are three assumptions in our method: (1) Most answers are normal answers and only a few answers are of low-quality. (2) Low-quality answers can be found by checking whether they are significantly different from its peer-answers, i.e. the other answers under the same QA thread. (3) Different questions should have different answer quality criteria.

Our method takes three solutions based on the three assumptions to tackle the three challenges. (1) Inspired by outlier detection algorithms, we propose a novel unsupervised optimization approach to detect low-quality answers by minimizing the data variance and maximizing the number of normal answers. (2) We incorporate a set of features which can capture the content differences between the answer and its peer-answers. (3) We apply the optimization model on each question individually, rather than on all questions at once.

We conduct experiments with three datasets. We make use of two benchmark datasets for answer quality prediction: an English dataset with 3,229 questions and 20,162 answers, and an Arabic dataset with 1,700 questions and 8,501 answers. We also label a third dataset sampled from Yahoo! Answer with 636 questions and 3,723 answers to test the performances on the popular CQA site. Experimental results on three datasets show that our method significantly outperforms other state-of-the-art methods.

Our contributions in this paper are of three-fold:(1) a proposal for an unsupervised optimization model for low-quality answer detection; (2) a proposal of using a set of features for capturing content differences among peer-answers; (3) empirical verification of the efficacy of the proposed method on two benchmark datasets and another dataset from a popular CQA site.

[1] https://answers.yahoo.com/question/index?qid=20090408172834AArbCtu

Table 1. An example of a question thread.

Question	
What 2 colors make green?	
Description	
I was painting a mission so I needed green paint for the grass but I run out.	
Answers	
A1*:	You know there is something called google right?
A2:	If you are mixing pigments it is blue and green. Things work differently on a computer and in some photo stuff. If you are mixing paint/pigments lookup RBG color wheel.
A3:	Yellow, blue, mix em together.
A4:	Blue and yellow.
A5:	Blue and yellow, but more yellow than blue.
A6:	Blue and yellow?

*: A1 is marked as the best answer.

2 Related Work

Although there have been many studies on CQA, to our best knowledge, no work has been done aiming at detecting low-quality answers based on the content differences among answers in a QA thread before. The related work can be broadly categorized into three threads.

Answer Quality Prediction. In previous studies, there are no commonly agreed definitions of answer quality in CQA. Jeon, et al. [7] define that a good answer tends to be relevant, informative, objective, sincere and readable. Sakai, et al. [20] propose a new evaluation methods based on graded-relevance metrics. Recently, two workshops of answer quality prediction are hold for SemEval-2015&2016 Task 3 [15, 16]. The organizers publish large manually judged English and Arabic datasets where answers are labeled in three levels: good, potential useful, and bad. And bad answers are subdivided into four categories: Irrelevant, Dialogue, Non-English and Others. We conduct our experiments on the two datasets published in SemEval-2015 Task 3.

Despite the lack of agreement on the answer quality definitions, researchers have made great progress in answer quality prediction these years. In the beginning, many works focus on the methods based on non-textual features. Jeon, et al. [7] use the maximum entropy approach and kernel density estimation to predict answer quality scores based on statistics of question and answer. Shah, et al.[21] propose a supervised method based on additional user information, which is considered as one of the state-of-the-art methods who use only non-textual features. Recently, a workshop in SemEval-2015 Task 3 [15] starts to target on semantically oriented solutions using rich language representations. Tran, et al. [23] and Nicosia, et al. [18] win on English and Arabic datasets respectively by using supervised methods with various of lexical, syntactic and semantic similarity measures. We take the methods of Shah, et al. [21], Tran, et al. [23] and Nicosia, et al. [18] as three baselines in this paper.

Review Spam detection. Review spam detection are also related to our work. Crawford, et al. [4] categorize review spams into three groups in their survey: untruthful reviews, reviews only on brands and non-reviews. By representing a review using a set of features of reviews, reviewers and products, classification techniques are used to assign spam labels to reviews [8, 11].

However, our work on low-quality answer detection is clearly different from review spam detection on three aspects. (1) Target: our work only detects irrel-

evant and useless answers. While as suggested by Crawford, et al. [4], a good review spam detection system should be able to identify whether a review is fake or untruthful. (2) Feature: a bunch of features representing the relevance between question-answer pair can be used in our work. While review comments can only refer to product names, thus features may heavily rely on review content. (3) Candidates: a question only have seven answers on average (See Section 4), while a popular product may have thousands of review comments.

Anomaly Detection. Anomaly detection (also known as outlier detection), referring to the problem of finding patterns in data that do not conform to expected behavior, is also related to our work. Hodge, et al. [6] indicate in their survey that techniques in unsupervised mode do not require training data, and thus are most widely applicable on this problem. Unsupervised methods make the implicit assumption that normal instances are far more frequent than anomalies [3]. Based on this assumption, many optimization models are proposed based on minimizing a customized loss function of data variance, where variables are classification labels of 0(anomaly) or 1(normal), and parameters are features. And techniques like soft labels [12] and gradient descent method [25] are proved useful when solving the optimization problems.

Our work still makes a difference with classical anomaly detection methods on the scope of similarities between instances. In anomaly detection, instances usually share lower similarities. For example, in review spam detection, reviews are often on a wide range of topics. While in CQA, questions, especially factoid questions, usually have very specific information needs, which narrow the topics and contents of answers. Thus, similarities in answers tend to be higher than in reviews. We take advantage of this characteristic in our optimization model.

3 Our Method of Low-Quality Answer Detection

In this section, we first formally present the problem of low-quality answer detection, and then illustrate our three key assumptions, and then devise a unsupervised method, and finally propose a set of features.

3.1 Problem

Given a question q and a set of its n answers $\{a_1, a_2, \cdots, a_n\}$. Each answer a_i is represented by an m-dimensional feature vector $\mathbf{x}_i = \{x_{i1}, x_{i2}, \ldots, x_{im}\}^\top$. All answers $\{a_i\}$ construct a $m \times n$ feature matrix $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.

Our goal is to learn a label vector $\mathbf{y} = \{y_1, y_2, \ldots, y_n\}^\top$ with $y_i \in \{0, 1\}$, where $y_i = 0$ means the corresponding answer a_i is low-quality, and $y_i = 1$ means a_i is a normal one and should be kept.

3.2 Assumptions

The first assumption is based on the observations in Table 1. In fact, most unsupervised anomaly detection methods have the same implicit assumption [3]. And the labeled datasets in Table 3 verify that it is true. And it helps to construct the second factor of our loss function in Formula (1).

Assumption 1 *Most answers under a question are normal ones and only a few of them are low-quality answers.*

Secondly, a question usually has specific information needs, which makes answers tend to have similar content. Then if an answer is significantly different with its peer-answers, it is likely to be low-quality. The second assumption helps to construct the first factor of Formula (1), and also inspires us to design features to capture the content differences between an answer and its peer-answers.

Assumption 2 *Whether an answer is low-quality or not can be known by checking its peer-answers.*

Moreover, different questions should judge answers in different quality criteria. For example, non-factoid questions favor long answers, and general questions expect yes/no, and factoid questions accept short noun-phrase, etc. Based on this observation, we have the third assumption, and apply our unsupervised model on each question instance, rather than on overall questions like supervised models.

Assumption 3 *Questions should have different answer quality criterion.*

3.3 Method

We propose an unsupervised learning approach to detect low-quality answers by minimizing the data variance and maximizing the number of kept answers.

Let $\overline{\mathbf{X} \cdot \mathbf{y}}$ denotes average weighted vector $\frac{1}{n}\sum_{i=1}^{n} y_i \cdot \mathbf{x}_i$. According to Assumption 3, we consider the optimization problem for each question instance:

$$\underset{\mathbf{y}=\{y_i\}}{\operatorname{argmin}} \frac{1}{mn}\sum_{i=1}^{n} ||y_i \cdot \mathbf{x}_i - \overline{\mathbf{X} \cdot \mathbf{y}}||^2 - \frac{\alpha}{n}\sum_{i=1}^{n} y_i \quad (1)$$
$$\text{s.t. } y_i \in \{0, 1\}, 1 \leqslant i \leqslant n$$

Where $\frac{1}{mn}\sum_{i=1}^{n} ||y_i \cdot \mathbf{x}_i - \overline{\mathbf{X} \cdot \mathbf{y}}||^2$ comes from Assumption 2, representing the data variance averaged on feature count. It will lead answers to have same labels, and those significantly different with other answers to have label 0. And $-\frac{\alpha}{n}\sum_{i=1}^{n} y_i$ comes from Assumption 1, helping to maximize number of answers with label 1, and α denotes the trade-off for the number of kept answers.

Formula (1) is novel in anomaly detection methods [12]. As described in Section 2 and Assumption 2, questions have specific information needs thus narrow topics, then answers tend to be more similar than instances in classical anomaly detection, such as reviews. We use variance with average weighted vector: $||y_i \cdot \mathbf{x}_i - \overline{\mathbf{X} \cdot \mathbf{y}}||^2$, instead of using variance with prediction: $||f(\mathbf{x}_i) - y_i||^2$, or variance with average vector: $||\mathbf{x}_i - \overline{\mathbf{x}}||^2$. In this way, the negative influence of low-quality answers is removed by labeling them 0, and the similarities between high-quality answers are highlighted since their feature values are similar.

This is an 0-1 programming problem, and thus it is NP-hard. To solve it, we adopt the soft label technique [12] and soften the label constraints to interval [0, 1]. By this means, it becomes a probabilistic constraint solving problem, and the learned distribution can represent the probabilities to be high-quality answers. By denoting as $\mathcal{L}(\{y_i\})$, we have a new optimization problem:

$$\underset{\{y_i\}}{\operatorname{argmin}} \mathcal{L}(\{y_i\}) = \underset{\{y_i\}}{\operatorname{argmin}} \frac{1}{mn} \sum_{i=1}^{n} \sum_{j=1}^{m} (y_i \cdot x_{ij} - \frac{1}{n} \sum_{k=1}^{n} y_k \cdot x_{kj})^2 - \frac{\alpha}{n} \sum_{i=1}^{n} y_i \quad (2)$$

$$\text{s.t. } 0 \leqslant y_i \leqslant 1, 1 \leqslant i \leqslant n$$

We employ a coordinate descent method to solve Problem (2) by taking y_i as a variable and fix other labels in each iteration. We calculate the partial derivative of $\mathcal{L}(\{y_i\})$ with respect to y_i, and set the result to be 0, we have

$$\frac{\partial \mathcal{L}}{\partial y_i} = \frac{2(n-1)}{mn^2} ||\mathbf{x}_i||^2 \cdot y_i - \frac{2}{mn^2} \sum_{k=1, k \neq i}^{n} \mathbf{x}_i^\top \cdot \mathbf{x}_k \cdot y_k - \frac{\alpha}{n} = 0$$

$$\text{s.t. } 0 \leqslant y_i \leqslant 1$$

By defining the solution as $\hat{y_i}$, we have:

$$\hat{y_i} = \frac{2 \sum_{k=1, k \neq i}^{n} \mathbf{x}_i^\top \cdot \mathbf{x}_k \cdot y_k + \alpha mn}{2(n-1)||\mathbf{x}_i||^2}$$

Then the optimal solution of Problem (2) is:

$$y_i^* = \begin{cases} 0, & \text{if } \hat{y_i} < 0, \\ \hat{y_i}, & \text{if } 0 \leqslant \hat{y_i} \leqslant 1, \\ 1, & \text{if } \hat{y_i} > 1. \end{cases}$$

The procedure ends when the Euclidean distance of two label vectors in consecutive iterations is less than ϵ or iteration number exceeds N^2. The final $\{y_1^*, y_2^*, \ldots, y_n^*\}$ represents the possibilities of being normal answers. A threshold $\mu \in [0, 1]$ is used for classification, if $y_i^* < \mu$, then a_i is a low-quality answer.

3.4 Features

Given a question q, and its description d and n answers $\{a_1, a_2, \cdots, a_n\}$, we have 173 features for each answer a_i. We choose these features because they represent nearly all aspects of a question-answer pair.

Table 2 clusters them into five groups. Group 1,2 and 3 are widely used and proved to be effective [23, 18, 21]. And Group 4 is a simple expansion of Group 3. We propose the last group, which seems to be new for answer quality prediction, as far as we know, although the idea is quite simple. All features are normalized to interval of $[0, 1]$ in our unsupervised models.

Question Features: features with prefix "Q" (denoted as $\{f_1(q)\}$, and its feature index set is denoted as F_Q) are obtained from statistics of question's and asker's information. Although all a_i share the same feature values (normalized 0 or 1) in this group, $\{f_1(q)\}$ can still be proved to be useful: in Formula (2), if t answers are classified as 1, the contribution of $\{f_1(q)\}$ in the first factor is:

$$\frac{1}{mn} \sum_{i=1}^{n} \sum_{j \in F_Q} (y_i \cdot x_{ij} - \frac{1}{n} \sum_{k=1}^{n} y_k \cdot x_{kj})^2 = \frac{t(n-t)}{mn^2} \sum_{j \in F_Q} x_{1j}^2 \quad (3)$$

[2] We set $\epsilon = 0.00001$ and $N = 200$ in our experiments

Table 2. Features of low-quality answer detection

Feature	#*	Description	[23]	[18]	[21]
Q_len	2	Word # of q and d.			◯
Q_category	1	The q's category. This feature is discretized to c boolean dimensions, where c equals to # of categories.	◯		
Q_ans#	1	# of answers for the q.			◯
Q_u_post#	2	# of total questions and answers of the asker.	◯		◯
Q_u_other	5	Other features of the asker: points, level, # of best answers, # of resolved questions and star count.			◯
A_len	1	Length of the a_i.			◯
A_rank	1	Reciprocal rank of the a_i in the answer list.			◯
A_symbol	2	Two booleans to identify whether the a_i contains some special strings (question marks, laugh symbols), and words which are only frequent in bad answers.	◯		
A_u_same	1	A boolean to indicate whether answerer is the asker.	◯		
A_u_post#	2	# of total questions and answers of the answerer.	◯		◯
A_u_other	5	Other feature of the answerer: points, level, # of best answers, # of resolved questions and star count.			◯
QA_word	1	Cosine similarity of bag-of-word vectors of qa pair	◯		
QA_ngram	20	5 similarity measures for n-grams($n \in \{1,2,3,4\}$) of qa pair: greedy string tiling[24], long common subsequence, Jaccard index, word containment[13] and cosine similarity.		◯	
QA_pos	1	Cosine similarity of bag-of-POS[22] tags vectors of qa pair		◯	
QA_noun	1	Cosine similarity of bag-of-noun vectors of qa pair. Noun are words containing "NN" in POS tags.	◯		
QA_tfidf	2	Sum of tf-idf[17] scores in answer collection of intersect subset of unigrams/bigrams between q and a_i.		◯	
QA_dep	1	Cosine similarity of of bag-of-word-dependency vectors of qa pair. We parse sentences to dependency trees[10] and regard dependency arcs (like "pre:buy-for") as words.	◯		
QA_meteor	1	Alignment score from Meteor Toolkit[5] between q and a.	◯		
QA_lda	1	Cosine similarity of LDA[2] topic vectors of qa pair.	◯		
QA_w2v	1	Alignment score between q word vectors and to a_i word vectors from pre-trained word2vec model[14]. (See details in [23])	◯		
QA_trans	1	Translation probability[1] from q to a by utilizing pre-trained translation model. (See details in [23])	◯		
DA_repeat	30	For each method in $\{f_3(a,q)\}$, get $f_3(d,a_i)$ as feature.			
QDA_repeat	30	For each method in $\{f_3(a,q)\}$, get $f_3(q+d,a_i)$ as feature.			
AP_repeat	60	For each method in $\{f_3(a,q)\}$, get $\max_{p\neq i}(f_3(a_i,a_p))$ and $\frac{1}{n-1}\sum_{p\neq i}(f_3(a_i,a_p))$ as features.			

*: The second column represents the actual feature count.

When $\sum_{j\in F_Q} x_{1j}^2 \neq 0$, Formula (3) have minimum value when $t=0$ or $t=n$. And influenced by $-\frac{\alpha}{n}\sum_{i=1}^{n} y_i$, t will approach n.

Answer Features: features (denoted as $\{f_2(a)\}$) with prefix "A" are obtained from the statistics of answer a_i and the answerer's information.

Question-to-Answer Features: features (denoted as $\{f_3(q,a)\}$) with prefix "QA" are obtained from contents of question-answer pair. The relevance of q and a_i are measured by various similarities on lexical, syntactic and semantic levels. Many of the state-of-the-art methods focus on this part.

Description-to-Answer Features: features (denoted as $\{f_4(d,q,a)\}$) in DA_repeat and QDA_repeat are obtained from contents of question, description and answer. DA_repeat take the same feature calculation methods in $\{f_3(q,a)\}$ to measure the relevance between d and a_i. QDA_repeat does the same way by concatenating question and description.

Table 3. Labeling guideline and label distribution

Label	Description	Qatar	Fatwa	Yahoo
Good	The answer directly responds to the question with relevant and useful content.	49.3%	20.2%	66.3%
Potential	The answer is potentially useful to the question.	10.0%	22.5%	11.3%
Bad	The answer is bad or irrelevant that the asker does not expect to receive.	40.7%	57.3%*	22.4%
- Irrelevant	The answer is totally irrelevant to the question.	17.9%	-	12.5%
- Dialogue	The answer does not directly respond to the question but hold an irrelevant chat, such as expressing gratitude or asking questions.	22.3%	-	8.0%
- Non-English	Irrelevant non-English answer.	0.5%	-	1.5%
- Other	Other irrelevant answer, such as advertisements.	0.0%	-	0.4%

*: Bad answers in Fatwa are manually added as noise data, so the proportion is large.

Answer-to-Peer Features: features (denoted as $\{f_5(a, a_p)\}$) in AP_repeat are obtained from answer a_i and its peer-answers $\{a_p | 1 \leqslant p \leqslant n, p \neq i\}$. They consist of two basic values: maximum and average of the similarities between a_i and peer-answers $\{a_p\}$ by repeating the feature methods in $\{f_3(q, a)\}$. According to Assumption 2, we propose these features to capture the differences between bad answers and peer-answers.

The time complexity of our method consists of two parts: feature calculation and coordinate descent method. In feature calculation, the time complexity is $O((l_Q + l_d) l_a m n^2)$, where l_Q, l_d, l_a the maximum length of questions, descriptions and answers, respectively, and m, n are the feature count and answer count, respectively. In coordinate descent method, it is $O(N m n^2)$, where N is the maximum iteration count. How to improve the efficiency of the process is still an interesting topic for future research that we will not address at this time. One possibility is to calculate the features off-line and store them in database.

4 Experiment

4.1 Experimental Setup

Datasets Preparation. We conduct experiments on three datasets: two public datasets from a workshop for SemEval-2015 Task 3[3] and our labeled dataset from Yahoo! Answers, and their statistics are given in Table 4.

Qatar Corpus: The workshop provides an large English dataset from Qatar Living website. Each question contains a description, several answers and aliases of askers and answerers. Each answer is labeled with one of six labels: "Good", "Potential", "Irrelevant", "Dialogue", "Non-English", "Other". The last four are regarded as "Bad" answers. See Table 3 for labeling guidelines and distributions. The task provides a split: Train, Dev and Test, for training model, tuning parameters and testing performances, respectively.

Fatwa Corpus: The workshop also provides an Arabic dataset from The Fatwa website. Each question has five answers, some of them are carefully answered by knowledgeable scholars in Islamic studies, while some are answers to other questions. Each answer is labeled with one of three labels: "Good", "Potential" and "Irrelevant". The task also provides a split: Train, Dev and Test.

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

Table 4. Overview of three CQA datasets

	Qatar			Fatwa			Yahoo	
	Train	Dev	Test	Train	Dev	Test	Train	Test
# of question	2,600	300	329	1,300	200	200	419	217
# of description	2,599	300	329	1,300	200	200	367	175
# of answer	16,541	1,645	1,976	6,500	1,000	1,001	2,407	1,316
# of answer per question	6.3	5.5	6.0	5.0	5.0	5.0	5.7	6.1
# of good answer	8,069	875	997	1,300	200	215	1,582	888
# of potential answer	1,659	187	167	1,469	222	222	278	144
# of bad answer	6,813	583	812	3,731*	578*	564*	547	284
- # of irrelevant answer	2,981	269	362	-	-	-	305	160
- # of dialogue answer	3,755	312	435	-	-	-	196	101
- # of non-English answer	74	2	15	-	-	-	36	18
- # of other answer	3	0	0	-	-	-	10	5

*: Bad answers in Fatwa are manually added as noise data, so the number is large.

Yahoo Corpus: In order to evaluate our methods on popular CQA site, we also label a dataset from Yahoo! Answer. Specifically, we crawl 6.4M questions associated with descriptions, answers and users' information[4] from Yahoo! Answer using a public API[5]. We then sampled 636 questions for labeling. Two expert labelers are invited to give each answer one of six labels with guideline in Table 3. If the judges disagree on an answer, we invited a third expert to make the final decision. We provide a split by ratio 2:1: Train and Test, for tuning parameters by cross validation, and testing performances, respectively.

Preprocessing. We illustrate the preprocessing method on each corpus. (1) We prepare an initial collection $\{q, d, \{a_i\}\}$ consisting of questions with descriptions and answers, where words are stemmed and stopwords are removed. (2) We prepare a concatenated collection $\{qd, \{a_i\}\}$ by concatenating q and d. (3) We prepare a document collection of qd and a_i to train a LDA model and a Word2vec model. (4) We prepare a mapping collection where qd is source and a_i is target to train a translation model.

The tools we used are: Stemmers(English/Arabic) in Lucene system[6] for word stemming, stopword lists(English/Arabic) from the Ranks website[7] for stopword removal, GibbsLDA++[8] for training LDA models, Word2vec tool[9] for training word2vec models, GIZA++[10] for training translation models, Stanford Tagger[11] and Parser [12](English/Arabic) for part-of-speech tagging and dependency parsing, and Meteor System[13](English/Arabic) for translation alignment score evaluation.

Evaluation Measure. We use four widely used measures to evaluate the performances: accuracy, precision, recall and F1-score.

[4] Features of Q_u_other and A_u_other in Table 2 are only traceable in Yahoo dataset.
[5] http://developer.yahoo.com/answers
[6] http://lucene.apache.org/
[7] http://www.ranks.nl/stopwords
[8] http://gibbslda.sourceforge.net/
[9] https://code.google.com/archive/p/word2vec/
[10] http://www.statmt.org/moses/giza/GIZA++.html
[11] http://nlp.stanford.edu/software/tagger.shtml
[12] http://nlp.stanford.edu/software/lex-parser.shtml
[13] http://www.cs.cmu.edu/ alavie/METEOR/

Table 5. Low-quality answer detection results on Qatar, Fatwa and Yahoo datasets

Corpus	Measures	S^S[21]	S^U	N^S[18]	N^U	T^S[23]	T^U	A^S	A^U	O^S	O^U
Qatar	Precision	50.0	49.6	54.3	55.0	64.6	69.8*	64.7	70.1*	67.0#	**73.6***
	Recall	76.5	72.7	71.2	67.8	79.2	71.7	80.3	72.3	**80.5**	75.3
	F1-score	60.5	57.7	61.6	60.7	71.2	70.7	71.6	71.2	73.1#	**74.4**
	Accuracy	58.9	56.2	63.5	64.0	73.6	75.6*	73.9	75.9*	75.7#	**78.8***
Fatwa	Precision	56.7	56.8	80.9	79.4	84.3	84.7	84.5	84.8	87.3#	**89.4***
	Recall	84.9	**91.8***	87.8	90.4*	88.3	90.4*	89.3	89.9	90.1	90.5
	F1-score	68.0	70.2*	84.2	84.6	86.3	87.5	86.8	87.3	88.7#	**90.0**
	Accuracy	54.9	56.0*	81.4	81.4	84.1	85.4*	84.7	85.2	87.0#	**88.6***
Yahoo	Precision	51.4	54.0*	47.6	49.3	59.8	65.7*	63.5	69.1*	66.4#	**73.6***
	Recall	64.1	56.2	66.2	52.0	76.2	67.4	78.8	73.2	**79.2**	75.5
	F1-score	57.1	55.1	55.4	50.6	67.0	66.5	70.3	71.1	72.2#	**74.5***
	Accuracy	79.2	80.2	77.0	78.1	83.8	85.4	85.6	87.1	86.9	**88.9***

Bold: the highest performance in terms of the measure.

*,#: statistically significant improvement of our models (two-sided sign-test, $p < 0.05$).

Baselines and Our Methods. We consider four state-of-the-art baselines: Tran, et al. [23] and Nicosia, et al. [18] use feature-rich supervised models and win first place on the Qatar and the Fatwa datasets respectively, denoted as T^S and N^S. Method of Shah, et al. [21] is another state-of-the-art method among supervised methods using only non-textual features, denoted as S^S. We create another supervised baseline by utilizing all above features plus DA_repeat and QDA_repeat features, denoted as A^S. Features of baselines are listed in Table 2.

To evaluate the effectiveness of answer-to-peer features, we create a supervised model by utilizing all features in Table 2, denoted as O^S.

Last, and most importantly, to evaluate the effectiveness of our unsupervised method, five comparison models are considered. They are the unsupervised models based on the same features from T^S, N^S, S^S, A^S and O^S, denoted as T^U, N^U, S^U, A^U and O^U respectively.

Parameter Tuning. There are three parameters in supervised models and two parameters in unsupervised models need to be tuned.

We train SVM classification models for supervised methods by the tool of SVMLight [9]. There are three parameters $\{c, j, b\}$[14]. The ranges are: c in {0.001, 0.002, 0.005, 0.01, $\cdots$, 50}, j in {0.5, 1, 1.5, $\cdots$, 8}, and b in {1, 0}. For Qatar and Fatwa datasets, we train models on Train sets, and choose the combinations with best F1-score on Dev sets. And for Yahoo dataset, we conduct four-fold cross validation on Train set and choose the one with the best F1-score.

In unsupervised methods, there are two parameters: the tradeoff weight α and decision threshold μ. The ranges are: α in {0, 0.1, 0.2, $\cdots$, 1}, and μ in {0, 0.01, 0.02, $\cdots$, 1}. We choose the combinations with best F1-scores on Dev sets for Qatar and Fatwa and on Train set for Yahoo.

4.2 Main Results

Table 5 shows the results on Qatar(English), Fatwa(Arabic) and Yahoo(English). The experimental results show that: (1) Our unsupervised model O^U outper-

[14] c: trade-off between training error and margin. j: cost-factor of training errors difference between positive and negative examples. b: use biased hyperplane or not.

forms all baseline methods on three datasets for nearly all metrics. (2) And our supervised model O^S outperforms other supervised methods on three datasets for all metrics. Most of the improvements are statistically significant by two-sided sign-test ($p < 0.05$). The results indicate that our methods are effective for low-quality answer detection.

4.3 Analysis Based on Models

We investigate the main reasons of the improvements of our unsupervised method.

(1) Supervised models do not take advantage of the fact that only minority are low-quality answers. Therefore, more answers are tended to be classified as bad answers. That is why supervised models often have high recall but low precision on this problem. This is particularly serious when the question is short. Specifically, for short questions, feature values are usually small on relevance measures. Then answers will have low SVM score and tend to be classified as bad answers. For example, we observe that T^S misclassifies all six correct answers under question "Write 5/2 as a percent?", since all the answers do not have common words with the question. While our unsupervised method T^U only misclassifies one, which increases the precision.

(2) Answers in supervised models share the same quality criteria. For example, it can be inferred from the labeled datasets that longer answers tend to be "Good". Thus, the supervised models have a bias on short answers. This bias causes misclassification when short answers are also acceptable. For example, for question "What is your favorite poptart flavor?", six in eight are one-word-answer "strawberry". We observe that T^S misclassifies them to be bad. While our unsupervised method T^U applies on each question instance, and is able to capture the high similarities between these answers. T^U results in the safest solution by assigning them all "Good", which increases the precision again.

4.4 Analysis Based on Features

In Table 5, our unsupervised model O^U effectively improves supervised model O^S, while others with less features are not the same. For example, A^U is only comparable with A^S, and S^U is even worse than S^S on Qatar and Yahoo. Since feature utilization is the only difference between unsupervised models, an interesting question is: what features are effective or useless in unsupervised models?

(1) We study on the effective features for unsupervised models. Notice that by bringing in extra answer-to-peer features, both O^S and O^U outperform A^S and A^U, and O^U is even better then O^S, while A^U and A^S have similar performances. We investigate the main reason of effectiveness of answer-to-peer features.

Many baselines focus on the relevance between question and answer, but overlook the difference between answer and its peer-answers. In fact, normal answers in a question usually share words that bad answers do not have. For example, for question "Where can I buy carrot cake?", most normal answers contain the same shop names but do not contain "carrot cake". While one bad answer expresses his dislike on carrot cake, which is useless to the question but contains "carrot cake". A^S gives out totally opposite wrong classifications. On the contrary, answer-to-pear features in O^S provide more clues for identifying normal answers,

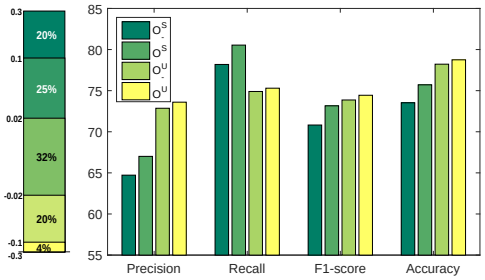

Fig. 1. Feature-label correlation coefficient distribution (left), and performances after removing no-correlation features (right).

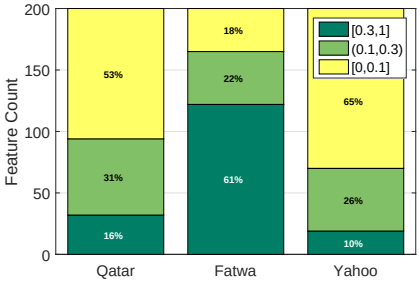

Fig. 2. Distributions of divergence of feature distributions between bad answer and normal answers.

then less normal answers are predicted to be bad, which increases the precision indeed. Moreover, O^U uses the strategy of operating on each question instance, then the effects of answer-to-peer features are larger in a single question than in the whole dataset, which makes O^U has better performances than O^S.

(2) Then, we investigate on the useless features for unsupervised model. A feature is useless if it is uncorrelated with human labels. Specifically, we assign 2, 1 and 0 to "Good", "Potential" and "Bad" answers, respectively. By this means, each feature has two vectors in a dataset, one is the vector of feature values, the other one is the assigned label vector, both dimensions are the size of dataset. We then calculate the Pearson correlation coefficient for the two vectors to represent the correlation between a feature and human labels.

Fig. 1[15] (left) counts the features according to correlation coefficients. All values turn out to be in range of [-0.3,0.3]. We divide them into five groups by thresholds {-0.1, -0.01,0.01,0.1}. We find that: 20% of features are highly positive correlated with quality, such as QA_w2v, QDA_trans, QDA_w2v, etc.; 25% are median positive, such as QA_tfidf, DA_lda, QDA_meteor, etc.; 4% are highly negative, such as A_u_same, A_len; 20% are median negative, which include most of answer-to-peer features. While 32% have near-zero coefficient, we treat them as uncorrelated with human labels, such as A_rank, A_symbol, etc.

To test the effectiveness of uncorrelated features, we conduct another experiment for O^S and O^U by removing uncorrelated features, denoted as O^S_- and O^U_-. Fig. 1 (right) shows their performances on Qatar dataset: O^S_- drops apparently from O^S in supervised scope, while O^U_- drops slightly from O^U in unsupervised scope. There are two reasons for these changes. Firstly, features are normalized in unsupervised methods but not normalized in supervised methods. Therefore, those features uncorrelated in unsupervised models may be correlated in supervised models. Secondly, features have global influences in supervised models, since all feature weights will change if any feature is removed. While uncorrelated features may have limited influences in unsupervised models since it is operating on each question instance. Take feature A_symbol as an example. In fact, only a few answers with special symbols and words inside are influenced by this feature.

[15] To save space we only report the results on Qatar dataset. The results in terms of Fatwa and Yahoo have similar trends.

4.5 Analysis Based on Data Sources

It is interesting to notice that, Fatwa dataset has more bad answers than other answers (see Table 3), which seems inconsistent with Assumption 1, but its performance scores are even higher than in Qatar and Yahoo (see Table 5). In fact, the Fatwa dataset is quite different. As we described in Section 4.1, bad answers in Fatwa are manually inserted as noise data. Thus, organizers can create as many bad ones as they want. That is why Fatwa has more bad answers. Moreover, all answers in Fatwa are carefully answered to the original questions. This means they are normal originally. Therefore, it is easier to distinguish a bad answer from other answers in Fatwa since they have totally different topics.

In order to confirm our guess, we investigate the divergences between bad and normal answers on each dataset. Specifically, (1) for each feature, we get the min and max among all answers. (2) Then we split the region $[min, max]$ into 100 slots, and count bad/normal answers on each slots, and then divide by total bad/normal answer count to get a discrete distribution. Thus, for each feature we have a distribution of bad answers and a distribution of normal answers. (3) We calculate a Jensen-Shannon divergence for the two distributions. (4) Finally, we count divergences in three slots [0, 0.1], (0.1, 0.3) and [0.3, 1].

Fig. 2 shows the results on three datasets. 61% of features in Fatwa have big divergences ([0.3, 1]), which is much larger than Qatar's 16% and Yahoo's 10%. It means that in Fatwa bad answers are more different from normal answers, making it easier to detect bad answers. Meanwhile, 65% of features in Yahoo have small values ([0, 0.1]). It means that in Yahoo bad answers are more ambiguous with normal answers, making it more difficult to detect bad answers. That explains why Yahoo has the lowest performance scores.

4.6 Analysis Based on Answer Labels

We study the effectiveness on different answer types. Models of A^S, O^S and O^U are used to see the gradual changes, where O^S uses extra answer-to-peer features to improve A^S, and O^U uses unsupervised model to improve O^S. We study on Qatar dataset since it is the largest one with full labels.

Figure 3 shows the number of correct and wrong predictions on each answer type[16]. (1) It is interesting to notice that O^S improves A^S mainly by reducing false predicted "Good" answers. It indicates that answer-to-peer features help to reduce false detections in supervised model, which also confirms the statements in Section 4.4. (2) It is also interesting that the "Potential" answers have exactly fifty-percent precision, well proving that they are potentially useful or useless. (3) O^U classifies less answers to be low-quality based on Assumption 1. Therefore, precision on "Good" answers is significantly improved, while the recall of "Irrelevant" answers also drops. (4) The performances on "Dialogue" answers is steady since some features are too strong, such as A_u_same, A_u_symbol, etc. For example, nearly all answers are labeled by "Dialogue" if their answerers are identical with the askers, or they contain special words like "thank".

[16] "Non-English" and "Other" answers are categorized into "Irrelevant" answers.

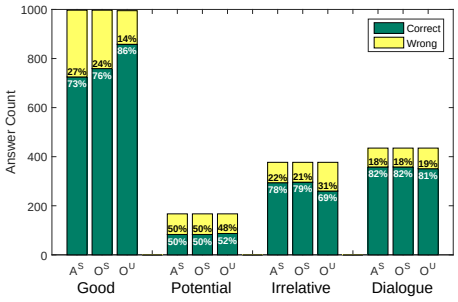

Fig. 3. Prediction results of A^S, O^S and O^U methods on each answer type.

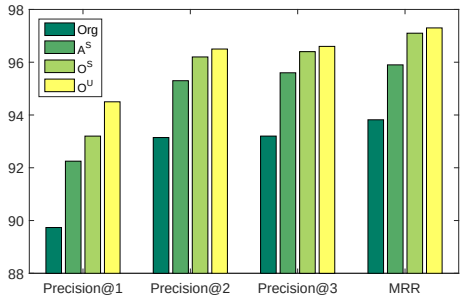

Fig. 4. Performances of re-ranking for improving user experience in browsing.

4.7 Analysis Based on User Experience

As we discussed in Section 1, the target of detecting low-quality answers is to improve the user experience when he browses a question page. A user usually browse answers from top-ranked to lower-ranked on question page. A top-ranked low-quality answer will reduce the user experience.

We conduct a re-ranking experiment to study whether user experience can be improved by our method. Specifically, (1) re-rank the answers according quality scores by descending order. (2) then evaluate user experience by precision at top positions (Precision@1,2,3) [19] and mean reciprocal rank (MRR) [19] methods. We report the results on Yahoo Test data since we can only get the original rank from Yahoo! Answer.

Fig. 4 shows the result of original rank (denoted as "Org") and the re-ranking results of three different models A^S, O^S and O^U. We can see that the user experience is improved by answer quality prediction methods from original rank results, which means some low-quality answers are correctly removed from the top positions by re-ranking methods. Moreover, the better method in classification generates better result in re-ranking. For example, O^U has better classification result than O^S in Table 5, and O^U also has better re-ranking result than O^S in Fig. 4. It is easy to understand that classification performance is highly correlated with re-ranking performance.

5 Conclusion

In this paper, we have investigated the problem of low-quality answer detection in community question and answering. We propose an unsupervised learning method based on three assumptions that most answers under a question are normal ones, and low-quality answers are different from other answers under the same question, and questions have different quality criteria. We propose a set of features to describe the difference from answers by taking advantage of the state-of-the-art methods. We empirically study the efficacy of the proposed unsupervised learning method as well as supervised methods on three datasets from the websites of Qatar Living, the Fatwa and Yahoo! Answer. The evaluation results show that our unsupervised method can significantly improve the supervised method on low-quality answer prediction.

References

1. Adam Berger et al. Bridging the lexical chasm: Statistical approaches to answer-finding. SIGIR '00, 2000.
2. David M. Blei et al. Latent dirichlet allocation. NIPS '01, 2001.
3. Varun Chandola et al. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), 2009.
4. Michael Crawford et al. Survey of review spam detection using machine learning techniques. *Journal of Big Data*, 2(1), 2015.
5. Michael J. Denkowski and Alon Lavie. Meteor universal: Language specific translation evaluation for any target language. EACL '14, 2014.
6. Victoria J. Hodge and Jim Austin. A survey of outlier detection methodologies. *Artif. Intell. Rev.*, 22(2), 2004.
7. Jiwoon Jeon et al. A framework to predict the quality of answers with non-textual features. SIGIR '06, 2006.
8. Nitin Jindal and Bing Liu. Review spam detection. WWW' 07, pages 1189–1190, 2007.
9. Thorsten Joachims. *Learning to Classify Text Using Support Vector Machines – Methods, Theory, and Algorithms.* Kluwer/Springer, 2002.
10. Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. ACL' 03.
11. Fangtao Li et al. Learning to identify review spam. IJCAI' 11.
12. Wei Liu et al. Unsupervised one-class learning for automatic outlier removal. CVPR '14, 2014.
13. Caroline Lyon et al. Detecting short passages of similar text in large document collections. EMNLP 01, page 118125, 2001.
14. Tomas Mikolov et al. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.
15. Preslav Nakov et al. Semeval-2015 task 3: Answer selection in community question answering. SemEval@NAACL-HLT 2015, 2015.
16. Preslav Nakov et al. Semeval-2016 task 3: Community question answering. SemEval@NAACL-HLT 2016, pages 525–545, 2016.
17. Ramesh Nallapati. Discriminative models for information retrieval. SIGIR' 04.
18. Massimo Nicosia et al. QCRI: answer selection for community question answering - experiments for arabic and english. SemEval@NAACL-HLT 2015, 2015.
19. Dragomir R. Radev et al. Evaluating web-based question answering systems. L-REC's 02.
20. Tetsuya Sakai et al. Using graded-relevance metrics for evaluating community qa answer selection. WSDM '11, 2011.
21. Chirag Shah and Jefferey Pomerantz. Evaluating and predicting answer quality in community qa. SIGIR '10, 2010.
22. Kristina Toutanova et al. Feature-rich part-of-speech tagging with a cyclic dependency network. In *HLT-NAACL*, 2003.
23. Quan Hung Tran et al. JAIST: combining multiple features for answer selection in community question answering. SemEval@NAACL-HLT 2015, 2015.
24. Michael J. Wise. YAP3: improved detection of similarities in computer program and other texts. SIGCSE' 96, pages 130–134, 1996.
25. Yan Xia et al. Learning discriminative reconstructions for unsupervised outlier removal. ICCV '15, 2015.