

# Learning to Recognize Reliable Users and Content in Social Media with Coupled Mutual Reinforcement

Jiang Bian  
College of Computing  
Georgia Institute of Technology  
jbian3@mail.gatech.edu

Yandong Liu  
Math & Computer Science  
Emory University  
yliu49@emory.edu

Ding Zhou  
Facebook Inc.  
dzhou@facebook.com

Eugene Agichtein  
Math & Computer Science  
Emory University  
eugene@mathcs.emory.edu

Hongyuan Zha  
College of Computing  
Georgia Institute of  
Technology  
zha@cc.gatech.edu

## ABSTRACT

Community Question Answering (CQA) has emerged as a popular forum for users to pose questions for other users to answer. Over the last few years, CQA portals such as Naver and Yahoo! Answers have exploded in popularity, and now provide a viable alternative to general purpose Web search. At the same time, the answers to past questions submitted in CQA sites comprise a valuable knowledge repository which could be a gold mine for information retrieval and automatic question answering. Unfortunately, the quality of the submitted questions and answers varies widely - increasingly so that a large fraction of the content is not usable for answering queries. Previous approaches for retrieving relevant and high quality content have been proposed, but they require large amounts of manually labeled data - which limits the applicability of the supervised approaches to new sites and domains. In this paper we address this problem by developing a semi-supervised coupled mutual reinforcement framework for simultaneously calculating content quality and user reputation, that requires relatively few labeled examples to initialize the training process. Results of a large scale evaluation demonstrate that our methods are more effective than previous approaches for finding high-quality answers, questions, and users. More importantly, our quality estimation significantly improves the accuracy of search over CQA archives over the state-of-the-art methods.

## Categories and Subject Descriptors

H.3.3 [Information Systems]: Information Search and Retrieval; H.3.5 [Information Systems]: On-line Information Services; G.2.2 [Discrete Mathematics]: Graph Theory

## General Terms

Algorithms, Measurement, Experimentation

## Keywords

Community Question Answering, Authority and Expertise in Online Communities

Copyright is held by the International World Wide Web Conference Committee (IW3C2). Distribution of these papers is limited to classroom use, and personal use by others.

WWW 2009, April 20–24, 2009, Madrid, Spain.

ACM 978-1-60558-487-4/09/04.

## 1. INTRODUCTION

Many information needs are better served by explicit, self-contained *answers* than by lists of results or documents. Recently, Community Question Answering (CQA) portals emerged that allow users to ask natural language questions that are answered by other users of the system. For millions of users, and tens of millions of questions posted on popular CQA sites such as Yahoo! Answers<sup>1</sup>, and Naver<sup>2</sup>, this method of information seeking has proven to be more successful than general-purpose Web search. For example, Yahoo! Answers already has tens of millions of users, and stores hundreds of millions of answers to previously asked questions, and serves millions of visits each day. These databases of past questions and respective answers are proving to be a valuable resource for specific information needs not well served by general-purpose Web search engines. Unfortunately, the quality, accuracy, and comprehensiveness of the content in the CQA archives varies drastically, and a large portion of the content is not useful for answering user queries. Not surprisingly, the reputation and expertise of the contributors can provide crucial indicators into the quality and the reliability of the content. The reputation of the contributor could also be a valuable factor for ranking search results from CQA repositories, as well as for improving the system interface and incentive mechanisms.

In a CQA environment, schematically shown in Figure 1, there are three sets of connected entities: users, answers and questions. In addition to the intuitive connection between questions and answers, users are also connected with two other sets of entities by both expressing specific information needs via posting questions, and by responding to existing question via posting their answers to questions.

Unfortunately, existing methods for estimating content quality in CQA either require large amounts of supervision (e.g., [2]) or focus on the network properties of the CQA without considering the actual content of the information exchanged (e.g., [17]). We observe that user reputation and the quality of the content they produce are often tied together in a mutually reinforcing relationship. Building on this observation, we propose a general framework for simultaneously calculating the answer and question quality and

<sup>1</sup><http://answers.yahoo.com/>

<sup>2</sup><http://www.naver.com/>

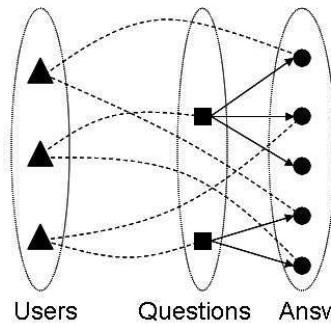


Figure 1: Network of interactions in CQA connecting users, questions and answers

user reputation based on their network relationships, coupled with the individual quality/reputation-related features. In our framework, the CQA interactions are viewed as composite bipartite graphs where each pair of entity types (e.g., users and the answers they generate) can form one bipartite graph. We develop a machine learning approach that starts with a set of known labels for users and answers, and exploits the mutual reinforcement between the connected entities in each bipartite graph to compute the respective quality and reputation scores simultaneously, iteratively refining the labels for the unlabeled entities. Our specific contributions include:

- A mutual reinforcement framework to calculate the quality and reputation scores of multiple sets of entities in a network relationship, simultaneously.
- A semi-supervised learning method to identify high-quality content and users in CQA that dramatically reduces the required amount of manually labeled data for training, while outperforming state-of-the-art supervised methods.
- An important practical application of our framework to enhance search over CQA archives, by incorporating the predicted content *quality* into the ranking.

Next we review related work, which sets the context for our new method that we introduce in Section 3.

## 2. RELATED WORK

Our work builds on the research in traditional automatic question answering (QA) [15], and on the more recent research area of QA over the Web [4]. However, question answering over Community QA archives is different from traditional QA. The most significant difference is that we are attempting to retrieve answers from a social media archive with a large amount of associated user information [2]. This kind of information (such as user reputation in a QA community) can be crucial to the answer quality and retrieval accuracy.

Due to the explosive rise in popularity of Yahoo! Answers and similar sites, CQA has become an active area of research. Jeon et al. [8] presented a machine translation model to find similar questions from a community QA service, but did not take quality of answers into consideration. Su et al. [14] analyzed the quality of answers in QA portals. Jeon et al. [9] built a model for answer quality based on features derived from the specific answer being analyzed. Agichtein et al. [2] presented a supervised approach to mining user interaction and content-based lexical

features to identify high-quality content in CQA. Recently, Bian et al. [3] developed a ranking system to retrieve relevant and high-quality answers, but they did not explicitly integrate content quality and user reputation information into the ranking process. While these models have shown to be quite effective for finding high quality [2] and relevant [3] content, they do not explicitly model user reputation, and require substantial amounts of manual supervision.

At the same time, there has been a long-standing interest in modeling authority, reputation and expertise in social networks and communities. Link-based ranking algorithms have been shown to be successful in the context of evaluating quality of Web pages. Two of the most prominent link-analysis algorithms are PageRank [12] and HITS [11]. Variations of PageRank and HITS have already been applied in many contexts, especially for propagating reputation and finding experts in the mutual reinforcement process. Guha et al. [7] and Ziegler [20] study the problem of propagating trust and distrust among users in social media, while considering trust as a transitive property in network relationships. Expert finding is also an active area of research, where researchers also take advantage of mutual reinforcement principle. Zhang et al. [17] analyze data from an on-line forum, seeking to identify users with high expertise. They apply both ExpertiseRank and HITS to identify users with high expertise. Jurczyk and Agichtein [10] show an application of the HITS algorithm to a CQA portal, especially the user interactions graph, and show a positive correlation between authority calculated with the HITS algorithm and answer quality. Campbell et al. [5] compute the score of HITS over the user-user graph in a network of e-mail exchanges, showing that it is more correlated to quality than other metrics. Zhou et al. [19] propose a method for co-ranking authors and their publications using their networks. Dom et al. [6] also study the performance of several link-based algorithms to rank people by expertise on a network of e-mail exchanges.

Although link-based and probabilistic approaches have been shown to be successful in ranking entities on the graph, most of them focus on ranking only one type of entity, and few of them utilize other properties of the entities except with link structure. Building on the previous work, our framework is based on the model of network relationships in social media [13] and exploits the mutual reinforcement principle [16]. We propose a mutual reinforcement framework for ranking sets of entities, specifically applied to the CQA network that connects users, questions, and answers. Our approach take advantage of mutually reinforcing relationship to rank various sets of entities simultaneously, and in this approach, many other features are used besides link structure.

## 3. LEARNING CONTENT QUALITY AND USER REPUTATION IN CQA

Answer and question quality are crucial to information retrieval in community question answering. It has been noted in previous work that user reputation is expected to correlate with answer and question quality but the relationship between user reputation and content quality is not straightforward. “Authoritative” users may provide poor answers, and “poor” users may occasionally provide excellent answers [2, 1]. In this section, we will explore a learning-based approach to calculate answer and question quality as well as user rep-

utation in CQA, simultaneously. We will focus on the specific characteristics of Yahoo! Answers and discuss how to employ coupled mutual reinforcement principle to learn answer and question quality and user reputation. We will start with a more precise definition of the problem of calculating answer and question quality and user reputation, and then describe the mutual reinforcement principle between these three types of entities in CQA. Then we present a coupled mutual reinforcement framework to model answer and question quality and user reputation. Based on mutual reinforcement in CQA network, we apply a semi-supervised regression-based approach to the problem of learning answer and question quality and user reputation.

### 3.1 Problem Statement

In a CQA system, there are three distinct types of entities: users  $\mathcal{U}$ , answers  $\mathcal{A}$ , and questions  $\mathcal{Q}$ . Questions and answers are posted by a diverse community of users. And one question can solicit several answers from a number of different users. We can further categorize users into two subsets: askers  $\mathcal{U}_q$  and answerers  $\mathcal{U}_a$ . Note that there can be an overlap between askers and answerers - that is, a user may post both questions and answers. Before proceeding, we define question and answer quality and user reputation more precisely:

**DEFINITION 1. Question Quality:** a score between 0 and 1 indicating a question’s effectiveness at attracting high-quality answers.

**DEFINITION 2. Answer Quality:** a score between 0 and 1 indicating the responsiveness, accuracy, and comprehensiveness of the answer to a question.

In previous work, question and answer quality were defined in terms of content, form, and style, as manually labeled by paid editors [2]. In contrast, our definitions focus on question effectiveness, and the answer accuracy – both quantities that can be measured automatically and do not necessarily require human judgments.

**DEFINITION 3. Answer-reputation:** a score between 0 and 1, indicating the expected quality of the answers posted by a user.

**DEFINITION 4. Question-reputation:** a score between 0 and 1, indicating the expected quality of the questions posted by a user.

Clearly, the definitions above are somewhat “circular” in that the reputation of the user depends on the quality of the questions or answers they post—where quality, in turn, can be influenced by the user reputation. In fact, we will soon show how to exploit this relationship in our mutual reinforcement framework. We now state our problem more formally:

#### Problem: Predicting Content and Contributor Quality

Given a CQA archive, determine the quality of each question and answer and the answer-reputation and question-reputation of each user, simultaneously, with minimal manual labeling.

In the rest of this section we will first introduce the “coupled mutual reinforcement principle” for content quality and user reputation in community question answering. We will then present our novel semi-supervised, regression-based approach, based on the mutual reinforcement idea.

### 3.2 Coupled Mutual Reinforcement Principle

Recall that our goal is to identify high-quality questions and answers, and high-reputation users, simultaneously. We now state the *mutual reinforcement principle* that underlies our approach to solving this problem with the minimum of manual labeling:

*An answer is likely to be of high quality if the content is responsive and well-formed, the question has high quality, and the answerer is of high answer-reputation. At the same time, a user will have high answer-reputation if she posts high-quality answers, and high question-reputation if she tends to post high-quality questions. Finally, a question is likely to be of high quality if it is well stated, is posted by a user with high question reputation, and attracts high-quality answers.*

Before we can turn this idea into an algorithm, we need to represent our setting more precisely. Recall that CQA systems are centered around three entities and their relationships: Users ( $\mathcal{U}$ ), questions ( $\mathcal{Q}$ ), and answers ( $\mathcal{A}$ ). The relationships between these entities are illustrated in Figure 2. In particular, to represent the relationships between answers and their authors, we can use a bipartite graph with an edge between each user and the answers that they post. Note that it is convenient to partition these graphs according to the *question thread* – that is, to consider subgraphs that involve answers to a particular question. Similarly, we can represent the relationship between askers and the questions they post by a bipartite graph with an edge connecting an asker to their posted question(s). We consider the sets of bipartite graphs that share the same question to be *coupled*.

We denote the whole graph in Figure 2 as

$$G(\mathcal{U}, \mathcal{A}, \mathcal{Q}, M_{\mathcal{U}\mathcal{A}}, M_{\mathcal{U}\mathcal{Q}}, M_{\mathcal{Q}\mathcal{A}})$$

where  $M_{\mathcal{U}\mathcal{A}} = [m_{ij}]$  is the  $|\mathcal{U}|$ -by- $|\mathcal{A}|$  matrix containing all the pairwise edges, i.e.,  $m_{ij} = 1$  if there is an edge between user  $\mathbf{u}_i$  and answer  $\mathbf{a}_j$ . Similarly,  $M_{\mathcal{U}\mathcal{Q}}$  and  $M_{\mathcal{Q}\mathcal{A}}$  are the matrices containing pairwise edges representing the association between users and questions they post, and question and the posted answers, respectively. Note that users may appear in both the asker and the answerer sets; however, we purposefully remove this additional coupling by separating the “asker” from the “answerer” personas of each user and modeling the reputation of each persona separately.

Now we can state the mutual reinforcement principle introduced earlier more precisely, as the set of four simultaneous equations governing the answer-reputation  $y_u^a$  and question-reputation  $y_u^q$  of a user  $u$ , and the corresponding answer quality  $y_a$  and question quality  $y_q$  respectively:

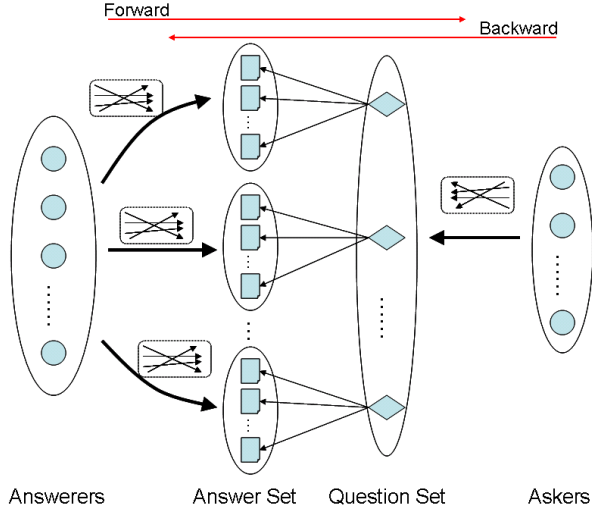
$$y_u^a \propto \sum_{u \sim a} m_{ua} y_a \quad (1)$$

$$y_a \propto \alpha \sum_{a \sim u} m_{ua} y_u^a + (1 - \alpha) y_{q(\sim a)} \quad (2)$$

$$y_u^q \propto \sum_{u \sim q} m_{uq} y_q \quad (3)$$

$$y_q \propto \gamma \sum_{q \sim a} m_{aq} y_a + (1 - \gamma) y_{u(\sim q)}^q \quad (4)$$

where  $u \sim a$  or  $u \sim q$  represents an edge between a user and her answers, or user and her questions;  $y_{q(\sim a)}$  denotes the



**Figure 2:**  $|\mathcal{Q}|$  coupled bipartite graphs connecting with user-question bipartite graph.

quality of answer  $a$ 's question;  $y_{u(\sim q)}^q$  denotes the question-reputation of the user who ask question  $q$ ; the symbol  $\propto$  stands for "proportional to"; and  $\alpha$  and  $\gamma$  are proportionality constants.

To simplify the notation, we collect answer-reputation and question-reputation scores of all users into vectors  $\mathbf{y}_u^a$  and  $\mathbf{y}_u^q$  respectively, and collect all answer and question quality scores into vector  $\mathbf{y}_a$  and  $\mathbf{y}_q$ , resulting in the simplified form of the same principle:

$$\mathbf{y}_u^a = M'_{UA}\mathbf{y}_a \quad (5)$$

$$\mathbf{y}_a = \alpha M'_{UA}^T \mathbf{y}_u^a + (1 - \alpha) M'_{QA}^T \mathbf{y}_q \quad (6)$$

$$\mathbf{y}_u^q = M'_{UQ}\mathbf{y}_q \quad (7)$$

$$\mathbf{y}_q = \gamma M'_{UQ}^T \mathbf{y}_u^q + (1 - \gamma) M'_{QA} \mathbf{y}_a \quad (8)$$

where  $M^T$  stands for the matrix transpose of  $M$ ;  $M'_{UA}$  and  $M'_{UQ}$  is derived from  $M_{UA}$  and  $M_{UQ}$  as for each  $m'_{ij} \in M'_{UA}$ ,  $m'_{ij} = \frac{m_{ij}}{\sum_{j=1}^{|\mathcal{A}|} m_{ij}}$  ( $m_{ij} \in M_{UA}$ ) and for each  $m'_{ij} \in M'_{UQ}$ ,  $m'_{ij} = \frac{m_{ij}}{\sum_{j=1}^{|\mathcal{Q}|} m_{ij}}$  ( $m_{ij} \in M_{UQ}$ ).

We can now turn the mutual reinforcement principle into a semi-supervised algorithm to estimates content quality and user reputation, as we describe next.

### 3.3 CQA-MR: Coupled Semi-Supervised Mutual Reinforcement

Due to the tight correlation and connection between those three sets of entities in CQA (questions, answers and users), we apply a mutually reinforcing approach to learn the question-reputation and answer-reputation of users as well as the quality of questions and answers, simultaneously. In the following, we first describe the features for learning question and answer quality and user reputation (Section 3.3.1). Then, we present a logistic regression approach for learning question and answer quality and user reputation (Section 3.3.2). However, we are given very few labels on answers and questions quality and users reputation in CQA. Thus, we apply the discussed coupled mutual reinforcement relationship for semi-supervised learning on answers and questions quality and users reputation, and such relationship

is also incorporated into the log-likelihood function (Section 3.3.2). Finally, we summarize a **CQA-MR** algorithm which can both fit the model and learn on answer and question quality and users reputation (Section 3.3.3).

#### 3.3.1 Features

In a CQA system, there are several complementary feature sets for answers, questions and users, respectively. Table 1 shows a list of features for answers, questions and users, which form the feature space of answers,  $X(\mathcal{A})$ , questions  $X(\mathcal{Q})$  and users,  $X(\mathcal{U})$ . We denote one answer as  $\mathbf{x}_a$  in answer feature space  $X(\mathcal{A})$ , one question as  $\mathbf{x}_q$  in  $X(\mathcal{Q})$  and one user as  $\mathbf{x}_u$  in  $X(\mathcal{U})$ .

#### 3.3.2 Learning Answer and Question Quality and User Reputation Using Coupled Mutual Reinforcement

Given an answer  $a$ , a question  $q$  and a user  $u$  described by feature vectors  $\mathbf{x}_a$ ,  $\mathbf{x}_q$  and  $\mathbf{x}_u$ , let the probability of them being a good answer, good question, good asker or good answerer be  $P(\mathbf{x}_a)$ ,  $P(\mathbf{x}_q)$ ,  $P_{\text{qst}}(\mathbf{x}_u)$  and  $P_{\text{ans}}(\mathbf{x}_u)$ , respectively. In the following, we will describe a generic approach to learning all these probabilities following the same way. We use  $P$  to denote any of  $P(\mathbf{x}_a)$ ,  $P(\mathbf{x}_q)$ ,  $P_{\text{qst}}(\mathbf{x}_u)$  or  $P_{\text{ans}}(\mathbf{x}_u)$  and use  $\mathbf{x}$  to represent the corresponding feature vector.

Using logistic regression, we model the log-odds of  $P(\mathbf{x})$  by the following linear models:

$$\log \frac{P(\mathbf{x})}{1 - P(\mathbf{x})} = \beta^T \mathbf{x} \quad (9)$$

where  $\beta$  are coefficients of the linear models. When given sufficient labeled instances, one can compute those coefficients by maximizing the corresponding log-likelihoods, say  $LL(\mathcal{X})$  for Equation 9:

$$LL(\mathcal{X}) = \sum_{\mathbf{x} \in \mathcal{X}} y \beta^T \mathbf{x} - \log(1 + e^{\beta^T \mathbf{x}}) \quad (10)$$

where  $y \in \{0, 1\}$  are the label of instance vector  $\mathbf{x}$ ;  $\mathcal{X}$  denotes the any of  $\mathcal{U}$ ,  $\mathcal{Q}$  or  $\mathcal{A}$ , which corresponds to the type of instance  $\mathbf{x}$ .

We can see that the above learning model depends exclusively on the corresponding feature space of the specific type of instances, i.e., answers, questions or users. Thus the quality of questions and answers are learned only based on answer-related or question-related features while the reputation of users is estimated based on user-related features.

After adapting the coupled mutual reinforcement principle between the answer and question quality and the user reputation, showed in Equation 5, 6, 7 and 8, we are able to measure the conditional log-likelihood of observing one label set given some others belonging to different kinds of but associated entities. We use  $\mathbf{y}$  to denote the current labels for  $\mathbf{x}$  and use  $\mathbf{y}'$  to denote new expected labels given the other kinds of old labels are known. We represent  $Y_d$  as the set of different types of entity associated with  $\mathbf{y}$ . For instance, based on Equation 6, the set  $Y_d$  of answer entity  $\mathbf{y}_a$  is  $\{\mathbf{y}_u^a, \mathbf{y}_q\}$ , and in Equation 7, the set  $Y_d$  of users for questions  $\mathbf{y}_u^q$  is  $\{\mathbf{y}_q\}$ .

We use KL-divergence to measure the conditional log-

**Table 1: Features Spaces:  $X(\mathcal{Q})$ ,  $X(\mathcal{A})$  and  $X(\mathcal{U})$** 

| Question Feature Space $X(\mathcal{Q})$ |   |
|---|---|
| Q: subject length                       | Number of words of question subject                               |
| Q: detail length                        | Number of words of question detail                                |
| Q: posting time                         | Date and time when the question was posted                        |
| Q: question stars                       | Number of stars received earned for this question                 |
| Q: number of answers                    | Number of answers received for this question                      |
| Answer Feature Space $X(\mathcal{A})$   |   |
| A: overlap                              | Words shared between question and answer                          |
| A: number of comments                   | Number of comments added by other participants                    |
| A: total thumbs up                      | Total number of thumb up votes for the answers                    |
| A: total thumbs down                    | Total number of negative votes for the answers                    |
| User Feature SPace $X(\mathcal{U})$     |   |
| U: total points                         | Total points earned over lifetime community                       |
| U: questions asked                      | Number of questions asked   |
| U: questions resolved                   | Number of questions resolved                                      |
| U: total answers                        | Number of posted answers  |
| U: best answer                          | Number of answers that were selected as "best answer"             |
| U: stars                                | Number of stars the user receive                                  |
| U: thumbs up ratio                      | The ratio of thumbs up votes the user posted before               |
| U: thumbs down ratio                    | The ratio of thumbs down votes the user posted before             |
| U: indegree                             | number of other users whose questions are answered by the user    |
| U: outdegree                            | number of other users who answer the questions posted by the user |
| U: hub score                            | the hub score of the user computed by HITS                        |
| U: authority score                      | the authority score of the user computed by HITS                  |

likelihood of  $\mathbf{y}$  given associated  $Y_d$ :

$$LL(\mathbf{y}|Y_d) = - \sum_{i=1}^{|\mathcal{X}|} \mathbf{y}(i) \log \frac{\mathbf{y}(i)}{\mathbf{y}'(i)} - (1 - \mathbf{y}(i)) \log \frac{1 - \mathbf{y}(i)}{1 - \mathbf{y}'(i)} \quad (11)$$

And now we can extend the objective function from the original log-likelihood in Eq 10 to the following:

$$L(\mathcal{X}) = LL(\mathcal{X}) + \sigma LL(\mathbf{y}|Y_d) \quad (12)$$

where  $\sigma$  is a prescribed weight parameter. This equation represents the combined log-likelihood for learning the probability of each type of entity. Note that the KL-divergence can be combined with the original log-likelihood naturally because both are log-likelihood measured on probability distributions so are of the same units.

### 3.3.3 Fitting Models–Algorithm

Next, we show how to fit the above models and how to solve the parameter estimation problem. The idea is to start with uniform distributions for  $P(\mathbf{x}_a)$ ,  $P(\mathbf{x}_q)$ ,  $P_{\text{qst}}(\mathbf{x}_u)$  and  $P_{\text{ans}}(\mathbf{x}_u)$ , and then iteratively update them to increase the likelihood based on their coupled mutually reinforcing relationship. In the following, we will first describe a generic approach to fitting any of the four logistic regression models for  $P(\mathbf{x}_a)$ ,  $P(\mathbf{x}_q)$ ,  $P_{\text{qst}}(\mathbf{x}_u)$  or  $P_{\text{ans}}(\mathbf{x}_u)$  in the mutual reinforcement framework. Then, we will describe an algorithm to learn answer and question quality and question-reputation and answer-reputation of user simultaneously.

We now describe how to fit the logistic regression model in more detail. As an example, consider fitting the model for the answer-reputation of users. From the current answer quality  $\mathbf{y}_a$ , we use Equation 5 to calculate  $\mathbf{y}_u^a$ . Note that we will keep the given labeled scores to the corresponding users. Then, given user-related features  $X(\mathcal{U})$  and  $\mathbf{y}_u^a$ , we use the Newton-Raphson update to compute the new  $\beta_u^a$  (denoted

as  $\beta_u^{\prime a}$ ) for the logistic regression model:

$$\beta_u^{\prime a} = \beta_u^a - \left( \frac{\partial^2 L(\mathcal{U})}{\partial \beta_u^a \partial \beta_u^{aT}} \right)^{-1} \frac{\partial L(\mathcal{U})}{\partial \beta_u^a} \quad (13)$$

Let  $\mathbf{X}_u$  denote the matrix of  $\mathbf{x}_u$  values,  $\mathbf{p}_u^a$  the vector of the fitted probabilities of users and  $\mathbf{W}_u^a$  a diagonal matrix with element  $(i, i)$  equal to  $p_u^a(\mathbf{x}_i)(1 - p_u^a(\mathbf{x}_i))$ , then the Newton-Raphson step is thus:

$$\beta_u^{\prime a} = \beta_u^a + (\mathbf{X}_u^T \mathbf{W}_u^a \mathbf{X}_u)^{-1} \mathbf{X}_u^T (\mathbf{y}_u^a - \mathbf{p}_u^a) \quad (14)$$

We rewrite this equation as:

$$\beta_u^{\prime a} = (\mathbf{X}_u^T \mathbf{W}_u^a \mathbf{X}_u)^{-1} \mathbf{X}_u^T \mathbf{W}_u^a \mathbf{z}_u^a \quad (15)$$

where  $\mathbf{z}_u^a = \mathbf{X}_u \beta_u^a + \mathbf{W}_u^{a-1} (\mathbf{y}_u^a - \mathbf{p}_u^a)$  is the residual. Using the new value of  $\beta_u^{\prime a}$ , we are able to calculate the new answer-reputation of users  $\mathbf{y}_u^{\prime a}$ .

Then we can apply the same approach to fit the logistic regression model for the answer and question quality and the question-reputation of users (denoted as  $\beta_a, \beta_q$  and  $\beta_u^a$ , respectively).

Based on the proposed method of fitting logistic regression models, we present the following algorithm **CQA-MR** (Alg. 1) for simultaneously learning answer and question quality and user ask and answer reputation, where the **Forward** phase carries out the mutual reinforcement from the left to the right while the **Backward** phase from the right to the left as shown in Figure 2. Since we generate combined log-likelihood for learning the probability of each type of entity, the mutually reinforcing approach of our algorithm should cause the successive estimates of content quality and user reputation to converge. We will empirically demonstrate convergence in Section 5.

In this section, we have defined the problem of calculating content quality and user reputation in CQA. Then, We present coupled mutual reinforcement framework and a semi-supervised regression-based approach to solve the

**Algorithm 1: CQA-MR**


---

```

input : questions, answers and users and their
         connection from CQA-network.
output: answer quality  $\mathbf{y}_a$ ;
         answer-reputation of user  $\mathbf{y}_u^a$ ;
         question quality  $\mathbf{y}_q$ ;
         question-reputation of user  $\mathbf{y}_u^q$ 
Start with an initial guess, e.g. uniform values, for  $\mathbf{y}_a$ ,
 $\mathbf{y}_u^a$ ,  $\mathbf{y}_q$  and  $\mathbf{y}_u^q$ ;
begin
  while  $\mathbf{y}_a$ ,  $\mathbf{y}_u^a$ ,  $\mathbf{y}_q$ ,  $\mathbf{y}_u^q$  not converge do
    Forward fit the logistic regression models and
    calculate new values for  $\mathbf{y}_a$ ,  $\mathbf{y}_q$  and  $\mathbf{y}_u^q$  in
    sequence ;
    Backward fit the logistic regression models and
    calculate new values for  $\mathbf{y}_q$ ,  $\mathbf{y}_q$  and  $\mathbf{y}_u^q$  in
    sequence
  end

```

---

problem. In the following sections, we will setup and carry on a large scale evaluation on the framework and our new method.

## 4. EXPERIMENTAL SETUP

This section presents our evaluation setup. First, we describe our dataset including corpus of questions, answers and the corresponding users. Then, we describe our evaluation metrics and some methods for computing answer quality and user reputation used for comparison in the experimental results reported in Section 5. We also describe several ranking methods to illustrate the effects of user reputation and answer and question quality on general QA-retrieval.

### 4.1 Data Collection

Yahoo! Answers Web service supplies an API to allow Web users to crawl existing question answering archives and the corresponding user information from the website. In order to ensure that our experiments are representative and repeatable, we use the TREC QA benchmarks to crawl QA archives and related user information. This is done by submitting TREC QA queries into the CQA site and retrieving the returned questions, answers and related users. (All the data in this paper can be found at <http://ir.mathcs.emory.edu/shared/>)

#### Factoid questions from the TREC QA benchmarks

We use factoid questions from seven years of the TREC QA track evaluations (years 1999–2006)<sup>3</sup> for our experiments. It is worth noting that TREC questions from the years 1999 to 2003 are independent of each other: each question is self-contained and is submitted directly as a query. Starting from 2004, however, the questions are organized in groups with a ‘target’. For those questions, we submit their ‘target’ as well as the questions themselves. In total, approximately 3,000 factoid TREC questions were compiled as the initial set of queries. Since we need *some* candidate answers from Yahoo! Answers to estimate how well different ranking functions perform, we select the 1250 TREC factoid questions that have at least one similar question in the Yahoo! Answers archive.

<sup>3</sup><http://trec.nist.gov/data/qa.html>

### The CQA dataset

Our dataset was collected in order to mimic a user’s experience with a CQA site. We submit each TREC query to the Yahoo! Answers Web service<sup>4</sup> and retrieve up to 10 top-ranked related questions according to the Yahoo! Answers ranking. For each of these Yahoo! questions, we retrieve as many answers as there are available for each question thread. There are, in total, 107293 users, 27354 questions and 224617 answers. And after automatically judging 224617 (*query, question, answer*) tuples by TREC answer pattern, 19409 tuples are labeled as “relevant” while 182308 are labeled as “non-relevant”, and there is no answer pattern provided from TREC for the other 22900 tuples. Note that, although the proportion of factoid questions in Yahoo! Answers may not be large, we use them in order to have objective metric of correctness, and extrapolate performance to whole QA archives.

### Relevance Judgments

In our experiment, the data are labeled for evaluating QA general retrieval in the following two ways: by using the TREC factoid answer patterns, and, independently, manually in order to validate the pattern-based automatic labels. For automatic relevance labels we use the available regular expression answer patterns for the TREC factoid questions. We check every answer’s text body, and if the text matches one of the answer patterns, we consider the answer text to be relevant, and non-relevant otherwise. We manually validated the accuracy of our automatically-assigned relevance labels in our previous work [3], and we found 90% agreement of automatic labels with manual relevance judgments.

### Data Labeling

We use a set of labels for good users and good answers from Yahoo! Answers directly. For some question threads in Yahoo! Answers, there is one “*best answer*” which is selected by the asker. These “*best answers*” can be viewed as high-quality answers. In addition, Yahoo! Answers selects some users as “*top contributors*” based on those users’ answering history. These “*top contributors*” can also be viewed as users with high answer-reputation. In our data, there are 4000 “*top contributors*” and 18000 “*best answers*”.

In order to evaluate the effectiveness of our algorithm for calculating answer quality and answer-reputation of users, we utilize a portion of these labels for users and answers. The other labels are used for testing by comparing with corresponding results of **CQA-MR**. In our experiments, we will keep 3600 *top contributors*’ labels and 16000 *best answers*’ labels for training our model, and then use the rest 400 *top contributors* and 2000 *best answers* to test the performance of our algorithm for learning answer quality and user answer-reputation. More importantly, we will evaluate the improvements to search, as described next.

In order to evaluate the effectiveness of our algorithm for computing question quality, we manually label a portion of the data. We randomly chose 250 resolved questions from Yahoo! Answers website which have received at least 5 answers. Two annotators were given 150 questions with 50 in common, and asked to label the quality of those 250 questions independently. The instructions for this labeling task were to consider both question subject and detail when examining question quality, and to consider answers when there is difficulty to understand the question. Questions

<sup>4</sup><http://developer.yahoo.com/answers/>

were labeled as “good”, “fair” and “bad”, according to special guidelines share by annotators.

Table 2 reports the agreement between the two raters on the 50 common questions. Since sometimes it is very hard to distinguish between ‘good’ and ‘fair’ questions, we also combined ‘good’ with ‘fair’ to form a binary labeling. Both agreements are reported in Table 2. As we can see that we can get moderate agreement for both methods. As we can see, the binary labeling results in higher agreement; hence, we will use the binary “Good”/“Bad” labels to evaluate question quality.

**Table 2: Inter-annotator agreement and Kappa for question quality**

|                   | 3 categories | 2 categories |
|-------------------|--------------|--------------|
| Agreement         | 64%          | 80%          |
| Kappa coefficient | 0.41         | 0.46         |

## 4.2 Evaluation Metrics

We adapt the following information retrieval metrics to evaluate the performance of the our algorithm for learning answer quality as well as the performance of general QA retrieval.

- **Mean Reciprocal Rank(MRR)**: The MRR of each individual query is the reciprocal of the rank at which the first relevant answer was returned, or 0 if none of the top  $N$  results contained a relevant answer. The score for a sequence of queries is the mean of the individual query’s reciprocal ranks. Thus, MRR is calculated as

$$MRR = \frac{1}{|Qr|} \sum_{q \in Qr} \frac{1}{r_q}$$

where  $Qr$  is a set of test queries,  $r_q$  is the rank of the first relevant document for  $q$ .

- **Precision at K**: for a given query,  $P(K)$  reports the fraction of answers ranked in the top  $K$  results that are labeled as relevant. In our setting, we require a relevant answer to be labeled “matched” for TREC pattern. For this metric, the position of relevant answers within the top  $K$  is irrelevant, while it measures overall user potential satisfaction with the top  $K$  results.
- **Mean Average of Precision(MAP)**: Average precision for each query is defined as the mean of the precision at  $K$  values calculated after each relevant answer was retrieved. The final MAP value is defined as the mean of average precisions of all queries in the test set. This metric is the most commonly used single-value summary of a run over a set of queries. Thus, MAP is calculated as

$$MAP = \frac{1}{|Qr|} \sum_{q \in Qr} \frac{\sum_{r=1}^N (P(r) \times rel(r))}{|R_q|}$$

where  $Qr$  is a set of test queries,  $R_q$  is the set of relevant document for  $q$ ,  $r$  is the rank,  $N$  is the number retrieved,  $rel()$  is a binary function on the relevance of a given rank, and  $P()$  is precision at a given cut-off rank.

## 4.3 Methods Compared

We now describe the methods used to compute user reputation, which we use for our main task of improving CQA retrieval. Specifically, we compare the following methods:

- **Baseline**: users are ranked by “indegree” (number of answers posted), an effective baseline estimate of user authority in CQA according to reference [17].
- **HITS**: we calculate the user reputation based on HITS algorithm. Users are ranked based on their authority scores.
- **CQA-Supervised**: we classify users into those with “high” and “low” reputation using a supervised classifier, namely SVM (SMO implementation), trained over the features in Table 1. Then user are ranked based on their reputation scores.
- **CQA-MR**: predict user reputation based on our mutual-reinforcement algorithm (Section 3.3).

Unfortunately, a direct experimental comparison with reference [2], which is most closely related to our work, is impossible as neither the dataset or the truth labels used for the experiments in [2] are available. However, **CQA-Supervised** is a similar approach and uses similar features to those described in [2], thereby providing a realistic state-of-the-art content quality classifier comparable to reference [2].

Our main task is to improve CQA retrieval by incorporating content quality and user reputation. We compare the following ranking methods:

- **Baseline**: In this method, the answers are ranked by the score computed as the difference of thumbs-up votes and thumbs-down votes received for each answer. This ranking closely approximates the ranking obtained when a user clicks “order by votes” option on the Yahoo! Answers site. The detail of this method and how to compute MRR and MAP under this setting is discussed in [3].
- **GBrank**: In this method, we apply the ranking method proposed in our previous work [3], which did not include answer and question quality and user reputation into ranking function. This method has been showed in [18] to have better performance than many state-of-the-art supervised ranking methods, such as RankSVM.
- **GBrank-HITS**: In this method, we optimize GBrank by adding user reputation calculated by HITS algorithm as extra features for learning the ranking function.
- **GBrank-Supervised**: In this method, we first apply a supervised method (SVM) to learn the answer and question quality and user reputation based on their individual feature set independently. Then, we optimize GBrank by adding obtained quality and reputation as extra features for learning the ranking function.
- **GBrank-MR**: In this method, we optimize GBrank by adding answer and question quality and user reputation calculated by **CQA-MR** as extra features for learning the ranking function.

Note that, **GBrank-MR** and **GBrank-Supervised**, we use the same set of labels in learning.

## 5. EXPERIMENTAL RESULTS

In this section, we will present several large-scale experiments. These experiments are used to demonstrate that (1) the **CQA-MR** algorithm exhibits good convergence behavior; (2) **CQA-MR** is effective for computing the question quality; (3) the performance of general QA retrieval can be improved by incorporating predicted quality features calculated by **CQA-MR**; (4) user reputation from **CQA-MR** tends to be better than those computed by other state-of-the-art methods; (5) the amount of supervision in **CQA-MR** affects the quality of predicted quality features.

### 5.1 Predicting Content Quality and User Reputation

#### 5.1.1 Mutual Reinforcement Convergence

We first perform training using Algorithm **CQA-MR** introduced above. We examine the convergence behavior of **CQA-MR** by calculating the log-likelihood function (Eq. 10) over the iterations. We find that the log-likelihood values increase and converge at around 40 iterations, when computing content quality and user reputation.

We also calculate the log-likelihood for different values of  $\sigma$ . We are able to find that the log-likelihood has much smaller values when  $\sigma$  is bigger, especially in the initial few iterations, which means that the conditional log-likelihood ( $LL(\mathbf{y}_a|\mathbf{y}_u^a, \mathbf{y}_q)$  and  $LL(\mathbf{y}_u^a|\mathbf{y}_a)$ ) is very small initially. Therefore, the difference in the labels between successive iterations is big, which implies labels' inconsistency, in the early stage of the algorithm. However, we can also find that when we take more than ten iterations, the log-likelihood is almost the same regardless of the  $\sigma$  values. Thus, at later stage of our algorithm, the log-likelihood values are more sensitive to the objective function of the logistic regression while the labels remain consistent across iterations, stabilizing at around 30 iterations.

#### 5.1.2 Predicting Answer-Reputation

We now compare the effectiveness of **CQA-MR** with other methods for predicting answer-reputation of users. For this task, we use the hold-out set of 3600 users, with 678 of them labeled as "top contributors". Figure 3 reports the fraction of "top contributor" users included in the top K users ranked by answer-reputation, for varying K. As we can see, **CQA-MR** exhibits significantly higher precision than **CQA-Supervised**, which, in turn outperforms **HITS** and the simple "in-degree" count baseline.

#### 5.1.3 Predicting Question Quality

We now compare the effectiveness of **CQA-MR** with other methods for predicting question quality. For this task, we use the set of 250 questions with manually labeled quality, which is described in section 4.1. We train **CQA-MR** as described above (that is, no additional question quality labels provided for training) and randomly select 100 labeled questions for evaluating the performance of predicting question quality. In order to compare with existing methods for predicting question quality, we also apply a supervised classifier, namely SVM (SMO implementation), trained over the features in Table 1. And the testing set is the same 100 labeled questions used above while the other 150 labeled questions are used for training SVM.

The mean average precision (MAP) for **CQA-MR** is 0.940

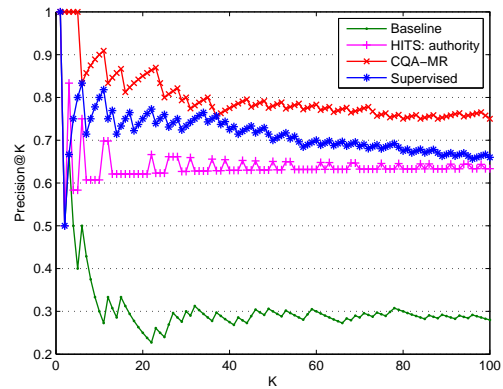


Figure 3: Precision at K for the status of *top contributors* in testing data

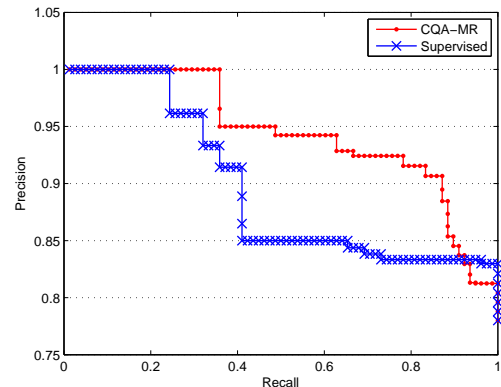


Figure 4: Precision-Recall curves for predicting question quality of **CQA-MR** and Supervised method.

while that value is 0.890 for the supervised method (SVM). Figure 4 shows the precision-recall curves for both methods, it is clear that **CQA-MR** gives good performance on predicting question quality and exhibits significantly higher precision than supervised method. In addition, we also try to add 150 labeled question as seeds in training **CQA-MR**. Interestingly, adding the question labels as additional seeds for training **CQA-MR** does not significantly improve performance.

### 5.2 Quality-aware CQA Retrieval

The answer and question quality and user reputation computed by **CQA-MR**, **CQA-Supervised** and **HITS** can be viewed as prior "static" features for QA retrieval since they are independent of queries. This complements "classical" information retrieval and QA retrieval, which primarily focused on query-dependent relevance features.

In this experiment, we seek to enhance the performance of general QA retrieval by incorporating predicted quality features (answer and question quality and user reputation). We use **GBrank** [18] as the ranking function and apply the same framework in our previous work [3]. For the training data we use 800 TREC queries and the associated community QA pairs, and we use another 450 queries and the associated community QA pairs for testing data. The set of features used to train the ranking function is described in detail in [3].

We train four ranking functions, **GBrank-MR**, **GBrank-**



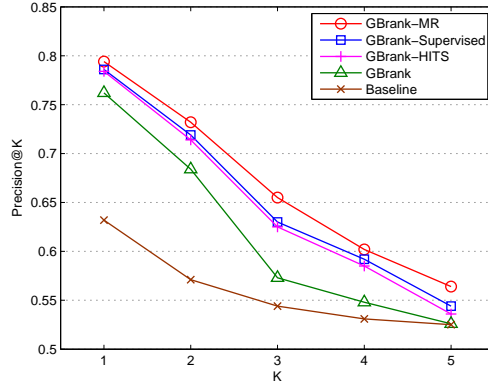


Figure 5: Precision at  $K$  for Baseline, GBrank, GBrank-HITS, GBrank-Supervised and GBrank-MR for various  $K$

Supervised, GBrank-HITS and GBrank, on training data (i.e., the 850 TREC queries) with predicted quality features added in the first three methods and training data without these features in the last one, respectively. Then, we test on the remainder hold-out testing data of 450 TREC queries and the associated community QA pairs.

Figure 5 illustrates the Precision at  $K$  of GBrank-MR, GBrank-Supervised and GBrank-HITS compared with GBrank and the baseline method. The figure shows that all of the four ranking functions out-perform the baseline method. Furthermore, after incorporating predicted quality features, i.e., answer and question quality and user reputation, GBrank-MR, GBrank-Supervised and GBrank-HITS give better precision than GBrank without these features. In particular, the Precision at 1 of GBrank-MR is about 79% compared to 76% Precision at 1 exhibited by GBrank.

Table 3: Accuracy of GBrank-MR, GBrank-Supervised, GBrank-HITS, GBrank, and Baseline (TREC 1999-2006 questions)

|                   | MRR          | Gain               | MAP          | Gain              |
|-------------------|--------------|--------------------|--------------|-------------------|
| Baseline          | 0.664        | -                  | 0.443        | -                 |
| GBrank            | 0.782        | -                  | 0.465        | -                 |
| GBrank-HITS       | 0.827        | +0.045(6%)         | 0.473        | +0.008(2%)        |
| GBrank-Supervised | 0.833        | +0.051(7%)         | 0.477        | +0.012(3%)        |
| GBrank-MR         | <b>0.865</b> | <b>+0.083(11%)</b> | <b>0.483</b> | <b>+0.018(4%)</b> |

Table 3 reports the MAP and MRR scores for GBrank-MR, GBrank-Supervised, GBrank-HITS, GBrank as well as the baseline method. Table 3 indicates that GBrank-MR, GBrank-Supervised and GBrank-HITS achieve much better performance than GBrank and the baseline method. In particular, for MRR scores, GBrank-MR achieves a gain of about 11% relative to GBrank; and GBrank-MR obtains double the gains of GBrank-HITS for both MRR and MAP scores. The above experiments illustrate the usefulness of the extracted static features in improving answer relevance.

### 5.3 Effects of the QA quality and User Reputation Features

We now explore the effects of the QA quality and user reputation, which is calculated by CQA-MR, on learning the ranking function. To this end, we perform a study on its influence on QA retrieval compared with existing graph-based

method and supervised learning method for computing user reputation, i.e., HITS and CQA-Supervised.

Figure 5 demonstrates the Precision at  $K$  of GBrank-MR compared with methods of GBrank-HITS and GBrank-Supervised. The GBrank-Supervised method replaces QA quality and user reputation calculated by a supervised learning method. Note that we use the same set of labels in learning for GBrank-MR and GBrank-Supervised. In order to compare the two methods GBrank-MR and GBrank-Supervised, we apply t-test based on their precision and the p-value of significance test is 0.02. The GBrank-HITS method replaces user reputation by those calculated by the HITS algorithm.

Figure 5 and Table 3 indicate that GBrank-MR achieves much better performance than GBrank-HITS, which implies that the user reputation calculated by CQA-MR gives more contribution than user’s authority scores computed by HITS. However, GBrank-HITS outperforms GBrank which does not contain QA quality and user reputation features. It shows that user’s authority scores from HITS are still useful to enhance the performance of QA retrieval. Our conjecture is that for user’s answer-reputation, it is much more important because CQA-MR not only utilizes network relationship but also individual reputation-related features while the authority scores in HITS only relies on the graph structure of CQA systems.

From Figure 5 and Table 3, we can also find that GBrank-MR performs significantly better than GBrank-Supervised ( $p < 0.03$ ). After analyzing information gain of features, we find that GBrank-MR assigns higher weights on QA quality and user reputation features. All of these imply that the QA quality and user reputation calculated by CQA-MR gives more contribution than those calculated by supervised method with limited amount of training data. GBrank-Supervised also outperforms GBrank which shows that QA quality and user reputation obtained by supervised method are still useful to enhance the performance of QA retrieval.

### 5.4 Effects of the Amount of Supervision

As mentioned before, we utilize a set of training labels for users and answers in the algorithms CQA-MR and CQA-Supervised to learn predicted quality features. In this experiment, we show the influence of the amount of training labels (i.e., degree of supervision) on the performance of CQA-MR and CQA-Supervised. The labeled set contains 3600 good users and 16000 good answers. We vary the size of training labeled set used in the learning process by randomly selecting a certain number of labels. For example, choosing 40% labels means to use 2400 good users and 10800 good answers in the learning process.

Figures 6 and 7 report the MRR and MAP scores for the hold-out validation data against varying amount of labeled training data for high quality question and answer retrieval. We can see that MRR and MAP scores increase when there are more labels in CQA-MR. CQA-MR can achieve same accuracy as CQA-Supervised with about half of the required training data. Therefore, CQA-MR can improve QA retrieval much more with less supervised learning compared to CQA-Supervised. We also find that GBrank-HITS have higher accuracy than GBrank-MR when the amount of supervision is less than 1200 examples, suggesting that HITS indeed identifies high quality content/users, but can be improved on by our method.

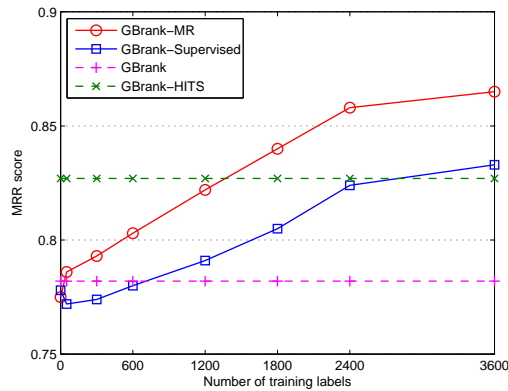


Figure 6: MRR of GBrank, GBrank-HITS, GBrank-MR, and GBrank-Supervised for varying fraction of training labels used in CQA-MR

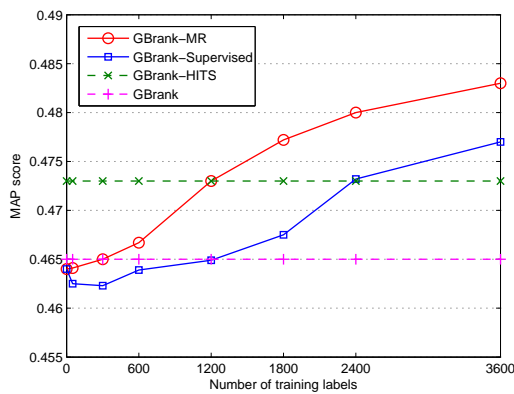


Figure 7: MAP of GBrank, GBrank-HITS, GBrank-MR, GBrank-Supervised for varying fraction of training labels used in CQA-MR

In summary, our experimental results show that **CQA-MR** is an effective method for identifying high quality content and highly-reputable users in CQA, particularly when training data is limited. More importantly, we have shown that the predicted quality and reputation features, when modeled explicitly, significantly improve the accuracy of CQA retrieval. Finally, our experiments show the effectiveness of our mutual reinforcement algorithm, as **GBrank-MR** significantly outperforms a state-of-the-art supervised method (**GBrank-Supervised**, implemented using SVM) operating on the same features.

## 6. CONCLUSIONS

We presented CQA-MR, a framework for semi-supervised quality and reputation estimation of content and users in Community Question Answering. We have demonstrated the effectiveness of CQA-MR in large-scale experiments of a CQA dataset comprising over 100,000 users, 27,000 questions, and more than 200,000 answers. Our experiments demonstrate significant improvements over the supervised method, both in accuracy and in reducing the required amount of training data. Interestingly, for the task of predicting question quality, CQA-MR significantly outperforms a supervised method while requiring *no* question quality training labels. Furthermore, we demonstrated a significant improvement that CQA-MR provides for the practical task of

searching the CQA archives.

In the future, we will explore the performance of our approach across different question domains, and take into account varying expertise (authority) of users for different domains. In addition, we also plan to focus on exploring probabilistic models of content quality and user authority, and on applying these techniques to other online communities.

## 7. REFERENCES

- [1] L. A. Adamic, J. Zhang, E. Bakshy, and M. S. Ackerman. Knowledge sharing and yahoo answers: everyone knows something. In *Proc. of WWW*, 2008.
- [2] E. Agichtein, C. Castillo, D. Donato, A. Gionis, and G. Mishne. Finding high-quality content in social media with an application to community-based question answering. In *Proceedings of WSDM*, 2008.
- [3] J. Bian, Y. Liu, E. Agichtein, and H. Zha. Finding the right facts in the crowd: Factoid question answering over social media. In *Proc. of WWW2008*, 2008.
- [4] E. Brill, S. Dumais, and M. Banko. An analysis of the askmsr question-answering system. In *Proceedings of EMNLP*, 2002.
- [5] C. Campbell, P. Maglio, A. Cozzi, and B. Dom. Expertise identification using email communications. In *Proc. of CIKM*, 2003.
- [6] B. Dom, I. Eiron, A. Cozzi, and Y. Zhang. Graph-based ranking algorithms for e-mail expertise analysis. In *Proc. of workshop on Data Mining and Knowledge Discovery*, 2003.
- [7] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proc. of 13th International World Wide Web Conference (WWW2004)*, 2004.
- [8] J. Jeon, W. Croft, and J. Lee. Finding similar questions in large question and answer archives. In *Proceedings of CIKM*, 2005.
- [9] J. Jeon, W. Croft, J. Lee, and S. Park. A framework to predict the quality of answers with non-textual features. In *Proceedings of SIGIR*, 2006.
- [10] P. Jurczyk and E. Agichtein. Discovering authorities in question-answer communities using link analysis. In *Proc. of CIKM*, 2007.
- [11] J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
- [12] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. In *Technical report, Stanford Digital Library Technologies Project*, 1998.
- [13] J. Scott. *Social Network Analysis: A Handbook*. SAGE Publications, January 2000.
- [14] Q. Su, D. Pavlov, J. Chow, and W. Baker. Internet-scale collection of human-reviewed data. In *Proc. of WWW2007*, 2007.
- [15] E. M. Voorhees. Overview of the TREC 2003 question answering track. In *Text REtrieval Conference*, 2003.
- [16] H. Zha. Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. In *Proc. of SIGIR*, 2002.
- [17] J. Zhang, M. Ackerman, and L. Adamic. Expertise networks in online communities: structure and algorithms. In *Proc. of (WWW)*, 2007.
- [18] Z. Zheng, H. Zha, K. Chen, and G. Sun. A regression framework for learning ranking functions using relative relevance judgments. In *Proc. of SIGIR*, 2007.
- [19] D. Zhou, S. Orshanskiy, H. Zha, and L. Giles. Co-ranking authors and documents in a heterogeneous network. In *Proc. of (ICDM)*, 2007.
- [20] C. Ziegler and G. Lausen. Propagation models for trust and distrust in social networks. *Information Systems Frontiers*, 7(4-5):337–358, 2005.