Retrieving People: Identifying Potential Answerers in Community Question-Answering

Long T. Le
Department of Computer Science, Rutgers University, 110 Frelinghuysen Road, Piscataway, NJ 08854-8019.
E-mail: longtle@cs.rutgers.edu

Chirag Shah
School of Communication and Information, Rutgers University, 4 Huntington Street, New Brunswick, NJ 08901-1071.
E-mail: chirags@rutgers.edu

Community Question-Answering (CQA) sites have become popular venues where people can ask questions, seek information, or share knowledge with a user community. Although responses on CQA sites are obviously slower than information retrieved by a search engine, one of the most frustrating aspects of CQAs occurs when an asker's posted question does not receive a reasonable answer or remains unanswered. CQA sites could improve users' experience by identifying potential answerers and routing appropriate questions to them. In this paper, we predict the potential answerers based on question content and user profiles. Our approach builds user profiles based on past activity. When a new question is posted, the proposed method computes scores between the question and all user profiles to find the potential answerers. We conduct extensive experimental evaluations on two popular CQA sites - Yahoo! Answers and Stack Overflow - to show the effectiveness of our algorithm. The results show that our technique is able to predict a small group of 1000 users from which at least one user will answer the question with a probability higher than 50% in both CQA sites. Further analysis indicates that topic interest and activity level can improve the correctness of our approach.

Introduction

Seeking information is a crucial part of human life. In the past decade, the development of the Internet has changed the way humans look for information. The process of information seeking often starts by submitting simple queries to search engines. However, results from search engines either do not give satisfactory answers for complex questions, or do not provide a searcher with their required personal information. Question-Answer (Q&A) services offer an alternative method for seeking information on the Internet. Online Q&A services allow people to post a question and receive responses from multiple participants in the online Q&A community. These services facilitate question asking in natural language, as opposed to relying on keywords when using a search engine. Rather than examine a summary list of documents displayed on a search engine results page, Q&A participants read and respond to other users who deliver personalized answers tailored to the asker's information need. The questions posted on CQA sites provide individualized answers that often rely on the *Wisdom of the Crowd*, or the idea that everyone knows something (Surowiecki, 2005). The emergence of Community Question-Answering (CQA) sites allows users to ask more complex questions. Some sites such as Yahoo! Answers and Stack Overflow are successful examples of CQA that can attract millions of users with millions of questions. Because of the popularity of CQA, there is a substantial amount of research focusing on its various aspects.

CQA supports question-answering activities by allowing users to post questions, then receive answers from the community. When posting a question, it is natural for the asker to assume or expect that their question will be answered. Receiving an incorrect answer or receiving no response at all could create a frustrating experience for the askers, and may affect the community's overall health (G. Li, Zhu, Lu, Ding, & Gu, 2015; Shah, Oh, & Oh, 2009). It is, therefore, an important issue within the context of CQA because this aspect influences how we could attract or identify users who could respond to a given question. Several works in the past have looked at failed questions to understand what went wrong (e.g., Shah, Radford, Connaway, Choi, & Kitzie, 2012).

Received September 29, 2016; revised January 28, 2018; accepted February 28, 2018

© 2018 ASIS&T • Published online July 18, 2018 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/asi.24042

Furthermore, some previous works also considered the problem of sending the questions to potential users. Such studies used topical similarity among content to locate possible answerers. Our work showed that using different similarity metrics could improve the correctness significantly. Another approach collected personal data from users' own sites or social networks to improve the efficacy of recommending answerers. In practice, collecting personal information from external social media is not practical. For example, it is challenging to match users' platforms from various sites, and privacy is a significant problem (Srba, Grznar, & Bielikova, 2015). Some research also studied commercial Q&A, where the motivations and purposes are very different from public CQAs such as Yahoo! Answers and Stack Overflow. For example, the users in commercial CQA sites contribute content as part of their daily work routine, whereas public CQA community members provide answers based on their interests and available time.

In the work presented here, our objectives are to identify who could/should answer a given question and presumably prevent the question from failing to be acknowledged. For this work, we analyzed two popular CQA sites to identify answerers. These CQA sites—Yahoo! Answers and Stack Overflow—attract millions of active users, and finding potential answerers is not an easy task. Furthermore, the passing of each day brings a larger number of posted questions. Thus, a simple and efficient algorithm could quickly divert new questions to potential answerers.

Finding the potential answerers is an important and useful problem. The online user typically starts the process of finding new information by using the search engine, then using the CQA to type the question. Typing the question takes more time and effort than typing the query. The user will be frustrated if the answer receives no response. Thus, finding the possible answerers could reduce the waiting time. Unfortunately, finding the potential answerers is a challenging task because of the diversity of content and users. Furthermore, the huge number of users in CQA also creates the challenge of finding the potential answerers. Our work will demonstrate a solution to this problem.

The work presented here has following primary contributions.

- *Empirical*: Analyzing the characteristics of CQA sites: We present a large-scale analysis of two popular CQA sites—Yahoo! Answers and Stack Overflow—to observe their similarities and differences. Yahoo! Answers is a general purpose CQA site, whereas Stack Overflow is a focused CQA revolving around programming-related content.
- *Theoretical*: Proposing an efficient algorithm to find potential answerers: We propose a new method to find potential answerers in CQA. The first step of our method builds users' profiles based on their history. Then, our method computes the scores between new questions and user profiles. Higher scores indicate a higher chance that a user will answer the question. Our method can capture users' topic interests and activity levels, providing a more effective technique for retrieving not only potential answers, but also other possible special types of people in CQA communities. We additionally measure the importance of each similarity feature in our framework. Our results demonstrate that similarity in an information network is a less robust metric than content and topics of interest for finding potential answerers.

Background and Related Work

CQA has attracted interest from the research community in recent years. In this section, we review background and related work on CQA. Because we are interested in identifying potential answerers from a community, we also present a brief review of works related to expert finding.

Online Q&A

Q&A services could be broadly classified as either online or face-to-face interactions, with traditional reference service in libraries being an example of the latter. Although online Q&A usually refers to user-generated answers, there are examples of systems that do automatic extractions of answers, such as Ask.com. There are two prominent types of human Q&A services: vertical and horizontal. The former is an online Q&A service that focuses on a specific topic. Examples of vertical Q&A, also referred to as online forums, include Stack Overflow (http://stackoverflow.com/) for programmers and PRIUSchat (http://priuschat.com/) for Toyota Prius owners. Four types of online Q&A fall under horizontal Q&A: community-based, collaborative, expert-based, and social. These sites typically cover a broad range of general topics rather than a single topic. To make this classification more precise, community-based, collaborative, and social Q&A can be placed under peer-based services, separate from expert-based Q&A.

Community Q&A (CQA)

How people share knowledge and communicate is an important research topic. In the last few decades, the Internet has changed the way people communicate and exchange ideas. CQA exemplifies this change in how people share their knowledge. In CQA, users post their questions to receive answers from anyone who can supply a correct answer or pertinent information.

Several works have looked at users' interest and motivation when participating in CQA (Preece, Nonnecke, & Andrews, 2004; G. Wang, Gill, Mohanlal, Zheng, & Zhao, 2013). Adamic, Zhang, Bakshy, and Ackerman (2008) studied the impact of CQA. The authors analyzed questions and clustered them based on their contents. The results showed that many users only participate in a narrow topic area, whereas some users can participate in a wide range of topics. The researchers also demonstrated that it was possible to predict the best answer by using basic features, such as an answer's length and the answerers' respective histories. In previous work (Le & Shah, 2016), we showed that a small

DOI: 10.1002/asi

fraction of users contribute heavily to the community. This work proposed an efficiency method to detect these top contributors in their early stages.

Question and Answer Quality in CQA

The quality of questions and answers in CQA is important because quality affects the users' experiences. The better a user's experience, the more actively they participate in a site. In CQA, a question may receive multiple answers and the community selects which is the best. Several works predicted question and answer quality in CQA (Gkotsis, Stepanyan, Pedrinaci, Domingue, & Liakata, 2014; Le, Shah, & Choi, 2016; B. Li, Jin, Lyu, King, & Mak, 2012; Ravi, Pang, Rastogi, & Kumar, 2014). These works used textual and nontextual features to evaluate quality. Some research also evaluated answer quality based on question quality, finding a high correlation between the two (Yao et al., 2014; Arora, Ganguly, & Jones, 2015). Different advance machine learning techniques such as random forest and deep neural network are applied to find the quality of contents in CQA and the users who fail to generate the good content (Chen et al., 2017; Le, Shah, & Choi, 2017). In our work, we concentrate on the problem of finding potential answerers. By sending the proper questions to the proper answers, we can reduce askers' wait time and possibly increase answer quality.

Ranking in CQA

CQA has become an important source of information because of its scope and its ability to attract widespread participation. Many questions in Yahoo! Answers appear in other search engines, such as Google or Bing. Searching content in CQA sites is a useful task and recent research has proposed different models to rank information in CQA (Bian, Liu, Agichtein, & Zha, 2008; Xue, Jeon, & Croft, 2008; X.-J. Wang, Tu, Feng, & Zhang, 2009; Zhou, Lan, Niu, & Lu, 2012). Bian et al. (2008) proposed a framework to rank posts in CQA based on both the structure and content of CQA archives. Wang et al. (X.-J. Wang et al., 2009) assumed that answers are connected to questions via different latent links, and used these characteristics to rank community answers. Zhang, Kong, Luo, Chang, and Yu (2014) used a network-based approach to develop a scalable ranking algorithm. Link-based ranking algorithms are also popular in CQA (Jurczyk & Agichtein, 2007; Nam, Ackerman, & Adamic, 2009; Yin, Han, & Yu, 2008). Because of the popularity of CQA in information retrieval, improving sites' search relevance is also an important topic (Carmel, Mejer, Pinter, & Szpektor, 2014; Wu et al., 2014). Dror, Koren, Maarek, and Szpektor (2011) represented the user and the question as a vector of multi-channel features, which included social signals and required hundreds of features. By surveying different ranking methods and evaluating question and answer relevance, we can better understand how ranking can assist with CQA question and answer relationships. Gollapalli, Mitra, and Giles (2013) rank the users with a graph-based method, but it does not clearly demonstrate how to rank the answers based on the users. The neural network is also deployed to rank the content in CQA (Iyyer, Boyd-Graber, Claudino, Socher, & Daumé, 2014; Qiu & Huang, 2015). Neural network shows the potential capability of analyzing complex content generated by users, and it performs better than feature-based methods (Andreas, Rohrbach, Darrell, & Klein, 2016).

Our work is also a type of ranking problem, but we rank the potential users. The more promising users will be on the top of the list, whereas those with less interest will be toward the bottom. To create the potential ranking, we compute the score between each user and the newly posted questions.

Expert Finding

Some previous works proposed simple methods to find the expert in a small community with several thousands of users (White & Richardson, 2012; White, Richardson, & Liu, 2011). Dror et al. (2011) considered question recommendation as a binary classifier by extracting numerous features from questions and data. Zhang, Tang, and Li (2007) used a social network to identify experts by proposing a propagation algorithm. Yarosh, Matthews, and Zhou (2012) studied the process of selecting an expert from a list of recommended users. Qu et al. (2009) used the topics' similarity to find potential answerers, which is not robust enough. In our work, we show that using others' signals can improve the ability to find potential experts. Srba et al. (2015) collected personal data from individuals' sites or social networks to improve the efficacy of recommending answerers, but this approach is not practical because of privacy issues and missing personal information in the CQA. In the recent work, Macina, Srba, Williams, and Bielikova (2017) also collected the information from a third party to find the answerers in an educational environment where the motivation is different from public CQA. Yan and Zhou (2015) used three-way tensor decomposition to recommend the answerer, but this method is not scalable for a large dataset with millions of questions and users. Another approach is collaborative answering (Pal, Wang, Zhou, Nichols, & Smith, 2013), where the question is sent to a group of similar users who would collaborate to create high-value content instead of finding top potential answerers. The approach proposed by Luo, Wang, Zhou, Pan, and Chen (2014) differs from those described above in several aspects. For example, they studied commercial QA as opposed to CQA. Their work considers user motivation and expertise, which is not available in public CQA. Furthermore, the motivation behind providing an answer in a working environment is also different from that in a public community.

It is clear from this brief literature review that much of the success and sustainability of CQA sites depend on platforms' respective content quality, community engagement, and user satisfaction. Important factors that affect user satisfaction include reduced wait time and increased probability that a question will be answered. Furthermore, most of the previous studies evaluated a single CQA site. Yahoo! Answers is used widely. Our work presented here is based on a large-scale study of two popular CQA sites: Yahoo! Answers and Stack Overflow. We hope to build upon previous work by analyzing methods and proposing strategies that help enhance satisfaction by recommending questions to the users best equipped and/or most likely to provide answers.

Identifying Potential Answerers

In this Section, we introduce the problem of finding answerers in CQA and propose a method to address this challenge.

Problem Definition

The problem is concerned with identifying potential answerers within a CQA community when a new question is posted. Given new questions, we want to find the top-K potential users who will be willing to give the corresponding answer.

This is an important problem because recommending possible answerers could reduce an asker's wait time or increase the likelihood that their question is answered. Finding the potential answerers is a difficult problem because of the diversity of the users and content in CQA. Next, we will describe our approach to solve the problem.

Our Approach

We propose a framework to predict suitable answerers based on a posting's history and features. The first step is building a user profile from a user's past activities. Then, given a new question *q*, we compute the score between *q* and all user profiles. A higher score indicates a better chance that a person will answer a certain question. Our method includes the following features:

- Similarity between question content and user profile
- Similarity between question topics and user expertise topics
- Similarity between asker and answerer in the information network
- User's activity level

Constructing a User Profile

We can build a user profile implicitly or explicitly. Online users might provide short descriptions about themselves, such as *"Software developer who spent some time in the C++ world but now lives in Eclipse developing java apps."* From self-declared profiles, we know that this user has expertise in C++, Java, and Eclipse. Unfortunately, explicitly constructed user profiles have two limitations. First, many users do not have self-declared profiles, or if they do, their description may not be complete. Second, many users do not consistently update their profile with current information. In Yang and Fang (P. Yang & Fang, 2013), the authors build the user profile based on user opinion, such as likes/dislikes in online reviews. However, implicitly inferring the user profile is a better method. We do this based on the list of questions a user has answered; their profile is the concatenation of all the questions they have addressed.

Computing Similarity Between Question and User Profile

Given all user profiles and a new question *q*, we measure the similarity between *q* and all user profiles. To measure the similarity, we treat each profile or question as a document. A corpus of documents is built from all user profiles and questions. The next step is computing the *tf-idf* of each document in this corpus (Manning, Raghavan, & Schutze, 2008). The *tf-idf* value of a word increases linearly with the number of times that word appears in a document but decreases by the frequency of the word in the corpus. After computing *tf-idf*, each document *d* is represented by a vector $\vec{\mathbf{v_d}}$. Similarity between a user and a question is measured by the *cosine similarity* between corresponding vectors. The cosine similarity between vectors $\vec{\mathbf{a}}$ and $\vec{\mathbf{b}}$ is described in Equation 2. Cosine similarity is used widely because of its simplicity and efficiency. The cosine similarity in the formula is between 0 and 1 because the *tf-idf* only returns nonnegative values. The value of 1 indicates an exact match whereas 0 indicates no relevance.

$$tf\_idf_{t,d} = tf_{t,d} \times idf_t = tf_{t,d} \times log\left(\frac{N}{df_t}\right) \tag{1}$$

$$\cos\left(\vec{\mathbf{a}}, \vec{\mathbf{b}}\right) = \frac{\vec{\mathbf{a}}\vec{\mathbf{b}}}{\|\vec{\mathbf{a}}\|\|\vec{\mathbf{b}}\|} = \frac{\sum_{i=1}^{n} \mathbf{a}_i \mathbf{b}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{a}_i)^2}\sqrt{\sum_{i=1}^{n} (\mathbf{b}_i)^2}} \tag{2}$$

Inferring Users' Topic Interests

Many users do not specify their topics of expertise. However, we can explicitly infer users' respective preferred topics. This idea is like building user profiles based off inferences. Our framework collects the topics of all questions answered by a user, *u*. User *u*'s strong topics are the concatenation of subjects of all questions they answered. By using the content that a user has answered, we can use topic modeling techniques such as latent Dirichlet allocation (LDA) to find each question's topic (Blei, Ng, & Jordan, 2003). LDA is a generative model. Each document in LDA is considered a mixture of different subjects. Figure 1 describes the generative process of LDA. The only observed variable is *W*; the rest are latent variables. The process of generating the topics for each document is as follows: (a) Choose α and β as the parameters of the Dirichlet prior on the per document topic distributions and per topic word distribution respectively; (b) Choose a topic from θ distribution; (c) Pick up a word *w* from multinomial distribution. Repeat this process for all documents.

To match the users' topics of interest with question topics, we apply *tf-idf* again. We treat each topic inferred by LDA as a term. Each user or question is represented as a document in which the document's "terms" are its topics. Then, we compute the *tf-idf* of each document in this corpus. The last step is to compute the topic similarity between a user and a question by applying cosine similarity.

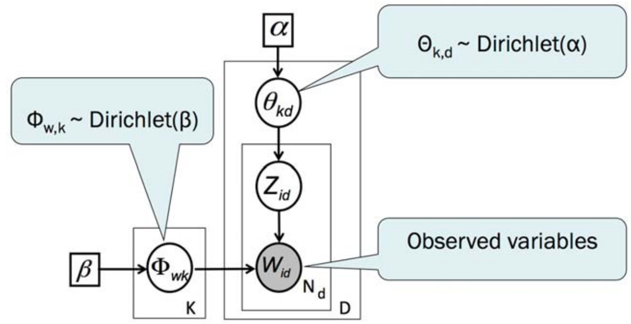

FIG. 1. Plate notation of finding topics of documents in LDA. α and β are the parameters of the Dirichlet prior on the per-document topic distributions and the per-topic word distribution respectively. θ is the distribution of topics in each document, ϕ is the word distribution for each topic, and W_{id} is the observed word. Blei et al. provided a detailed explanation of LDA (Blei et al., 2003). [Color figure can be viewed at wileyonlinelibrary.com]

Similarity in Information Network

In popular CQA sites such as Stack Overflow and Yahoo! Answers, user friendships are not as explicit as those found on social networking sites. We construct the graph $G=(V,E)$. The list of nodes, $V=\{u_1,u_2,\ldots,u_n\}$, is the set of users in the community. There is an edge $e\in E$ between u_i and u_j if user u_j answered a question posted by u_i. A popular method to measure the closeness between two nodes is using Random Walk with Restart (RWR). RWR provides a good relevance measure of two nodes in a graph. There are two main components of RWR starting at a *seed node i*: (a) a random walk to a neighbor is performed, and (b) at any step, there is a small probability c of jumping back to the seed node. Given a seed node i in a graph, the relevance between node i and j is computed as Equation 3, where $\vec{d}$ is starting vector. The relevance score is the j^{th} element of vector $\vec{r_i}$.

$$\vec{r_i}=(1-c)G\vec{r_i}+c\vec{d} \tag{3}$$

A traditional method to compute the RWR is the power iteration method, which repeats Equation 3 until convergence. Because of the large size of our graph, we applied fast RWR, as proposed in Tong, Faloutsos, and Pan (2006). The basic idea of fast RWR is partitioning the graph in smaller communities. These communities connect to each other through the bridge edges. Then, the RWR score can be combined based on the small communities and bridges. Ganu and Marian, (2014) examined the measure of the similarity between users in a small forum by using continuous posts in a thread.

Activity Level of User

Active users are more likely to answer new questions. The next Section (Datasets and Characterization of the Data) also shows that a small fraction of users contribute most of the content in CQA. Users in Yahoo! Answers are awarded points when answering questions, whereas Stack Overflow users also earn reputation value by answering questions. In general, users who give a greater number of answers can earn a higher reputation in Stack Overflow or higher scores in Yahoo! Answers. The majority of CQA sites have some metrics that measure their users' activity level. In cases where there is no such metric, we can use the number of posts as the user activity level.

Summary of Our Framework to Find Potential Answerers

First, we explain how to compute the score between a user and a question. Given a question q asked by asker a and an arbitrary user u in the community, the score between u and q is calculated, as in Equation 4. The score is the product between the similarity and the $log(activity\_level)$. We use the log of activity level because of the power law distribution of user activity level.

$$\begin{aligned}&score(u,q)=log(AL(u))\\&\times[\alpha_0+\alpha_1*sim\_C(u,q)+\alpha_2*sim\_T(u,q)+\alpha_3*sim\_U(u,akr)]\end{aligned} \tag{4}$$

where:

- $AL(u)$ is the activity level of user u
- $sim\_C(u,q)$ is the similarity between question content and user profile
- $sim\_T(u,q)$ is the similarity between question topics and user expertise topics
- $sim\_U(u,akr)$ is the similarity between user u and asker akr as in equation 3
- α_i is the parameter that controls the different weighting of these similarity values.

To find the parameter α_i, we used answerers' past answering history. During the observation period, we calculated the real score of each pair of users and questions. If user u is the n^{th} person who answers a question, the real score $y=1/n$. Otherwise, we assign score 0 if this person does not answer said question. In general, the sooner a user answers a question, the higher the score value they should receive. Thus, the parameter α_i can be inferred as:

$$\alpha_i=\arg\min_{\alpha}\sum_{i=1}^{m}\left[y_i-log(AL)\sum_{i=0}^{3}\alpha_i\times sim_i\right] \tag{5}$$

where y_i is the actual output of answering activity in the observation period, m is the number of trainings, and sim_i is the similarity value including similarity of content, topics, and information network. The problem in Equation 5 can be converted to a standard linear regression problem by dividing by $log(AL)$. Then, we can easily calculate the parameter α_i. In general, the Equation 5 is converted to:

$$\alpha^*=\arg\min_{\alpha}\sum_{i=1}^{m}\left[\frac{y_i}{log(AL)}-\sum_{i=0}^{3}\alpha_i\times sim_i\right] \tag{6}$$

In Equation 6, the value of y_i and $log(AL)$ are constant based on users' answering history. In particular, $y_i=1/n$ in the case of the n^{th} answer and $log(AL)$ is the logarithm number of users' answers in the observation period. Thus, the value α_i is the solution of the standard regression problem, and can be solved easily by using different approaches such as using closed form solution or gradient descent (Bishop, 2006).

In summary, our framework of finding the potential answers is based on different similarity metrics such as content similarity, topic similarity, and the similarity information network. But it is not clear how each similarity metric will contribute to the likelihood that the user will give the answer to a question. To solve that issue, we used the history of the answering activity to infer the parameter. This approach is standard in different data mining and machine learning problems by finding the optional parameter α_i. In our framework, we also consider the activity of users which caused the problem to be more complicated. To solve this issue, we convert the problem to standard linear regression by converting Equation 5 to Equation 6. Again, Equation 6 is very easy to solve because it is a standard regression problem.

Next, we explain in detail the algorithm that finds potential answerers. Algorithm 1 describes our *QRec* algorithm. The first step of this formula constructs the user profiles. Given a new question *q*, *QRec* calculates the scores between *q* and each user. The list of potential answerers is comprised of users who have top scores. We need to build a user's profile once (Line 2 in Algorithm 1), and apply it to multiple questions. In a real application, the user might change their interest. In such a case, we can update their profile, but this is only needed after a long period (i.e., after a few months). Steps 9 and 10 are expensive but can be computed off-line. Furthermore, we applied Fast RWR for this large information network. Another issue is finding the topics for new questions. In Step 6 of Algorithm 1, we consider each question and user profile as a document. When a new question appears, online LDA (Hoffman, Blei, & Bach, 2010) can be applied to quickly find the question's topic without training the whole corpus again. Thus, our ranking method is scalable and can be applied easily on large datasets.

Algorithm 1 *QRec* Algorithm

```
Input:
        • A set of $users_i$, $i = 1, \ldots, n$.
        • A question $q$
        • Size of possible answerers $k$
Output: The list of $k$ users most likely to answer $q$
 1: for $i=1:n$ do
 2:     Construct the user profile based on self-declared profile and
        answering history.
 3: end for//Compute the tf-idf of raw content.
 4: Construct the RAW content corpus, each document is a user profile
    or a question.
 5: Compute the tf-idf of each document in RAW content corpus.//Compute the tf-idf of hidden topics.
 6: Infer the topics of interest expressed by each user profile and the
    hidden topics of the question q by applying LDA.
 7: Construct the TOPICS corpus, where each term represents a hidden
    topic. Each document represents the topic interests of a user or hidden
    topics of a question.
 8: Compute the tf-idf of each document in the TOPICS corpus.
    //Construct the information network and compute user similarity.
 9: $G=(V,E)$. The list of nodes: $V=\{u_1, u_2, \ldots, u_n\}$, are the set of users
    in the community. $\exists\ e \in E$ between $u_i$ and $u_j$ if user $u_j$ answered a
    question posted by $u_i$
10: Compute $sim\_U(u_i, u_j$ as Equation 3)//Compute scores between each
    user and question
11: for $i=1:n$ do
12:     Compute scores between $user_i$ and $q$ (as in Equation 4)
13: end for
14: return topK: list of $k$ users with top scores
```

Datasets and Characterization of the Data

Data Description

We used data from two popular CQA sites: Yahoo! Answers and Stack Overflow. Yahoo! Answers is a general purpose CQA site, whereas Stack Overflow is a focused CQA that hosts programming-related questions.

Yahoo! Answers ("answers.yahoo.com") is a forerunner of CQA. It is a general-purpose Q&A site, which accepts any question if it does not violate the site's guidelines. The platform allows any of its users to post questions and answers. Each question in Yahoo! Answers is assigned to a category. A user in Yahoo! Answers might have many different types of interaction, such as sharing, discussion, advice, and polling (Adamic et al., 2008). To encourage user participation, the site awards points to users. The points determine the user's level; the levels range from 1 to 7.

Stack Overflow ("stackoverflow.com") is another CQA specifically focused on the field of programming; thus, it only accepts questions and answers related to programming. Similar to Yahoo! Answers, users in Stack Overflow can engage in a wide range of activities that include upvoting and downvoting posts or offering a bounty to a question to attract an answerer. Users in Stack Overflow earn reputations by providing high quality questions and answers. For example, a user's reputation increases if their question/answer is upvoted or their answer is accepted. In contrast, a user's reputation is diminished when their question/answer is downvoted or marked as spam. Each question is assigned tags based on its content. The tags can be considered the question topic. Because Stack Overflow is a focused site, the questions are normally difficult. The content in Stack Overflow is carefully managed by the site's administrator and community. For example, duplicate questions or nonuseful questions will be merged or removed to maintain the high quality of the site.

Table 1 lists the types of actions and how they affected a user's score. Stack Overflow uses the term "reputation," whereas Yahoo! Answers uses the term "point." In general, Stack Overflow has a stricter policy to maintain high quality posts, whereas Yahoo! Answers focuses on increasing the amount of time users spend within the site.

Table 2 describes some statistics of the dataset used in our experiment. We crawled the questions and answers in Yahoo! Answers whereas the data dump of Stack Overflow is available publicly[1] https://archive.org/details/stackexchange.

TABLE 1. Score system in Stack Overflow and Yahoo! Answers.

Stack overflow	Change in reputation	# Yahoo! Answers	Change in point
Answer is upvoted	+10	Join Yahoo! Answers	+100 (one time)
Question is upvoted	+5	Ask question	−5
Answer is accepted	+15	Choose best answer	+3
Answer is downvoted	−2	Answer a question	+2
Question is downvoted	−2	Log in Yahoo! Answers	+1
Answer win bounty	+bounty amount	Receive thumbs-up	+1
Offer bounty	-bounty amount	Receive a violation	−10
Answer marked as spam	−100		

TABLE 2. Description about data.

Site	Period	# of Users	# of Questions	# of Answers
Yahoo! Answers	Jan ’08 to Dec ’09	1.07 M	2.24 M	11.71 M
Stack Oveflow	Jan ’14 to Sep ’14	3.4 M	1.3 M	3.68 M

Characterization of the Data

In this section, we describe some of the CQA characteristics that help us gain a better understanding and rationale to justify our method. For example, when calculating the score between a user profile and a question, we use the *log* instead of the real value of activity level. This decision is based on the distribution of user activity levels.

User Activity Level. Figure 2 plots the distribution of the number of answers given by each user. We see that the distribution follows the power law distribution with heavy tail. Many users in CQA only answer a few questions, whereas a small number of users are very active. In both CQA sites, a small fraction of users contribute a majority of the content. Because the distribution follows the power law, our formula to calculate the score uses the *log* of activity level.

The Question Length. Figure 3 plots the distribution of the number of words per question in Stack Overflow. Questions in Stack Overflow are normally longer than questions in Yahoo! Answers. Stack Overflow is a focused CQA site and the questions are carefully managed. Unnecessary or meaningless questions are deleted to maintain the site’s high quality. But the length of questions in CQA sites is normally short. For example, half of the questions in Yahoo! Answers and Stack Overflow are less than 47 and 160 words respectively.

Topic Distribution. Questions in Stack Overflow and Yahoo! Answers are grouped into topics. There are 31,250 topics in Stack Overflow and 945 topics in Yahoo! Answers. Figure 4 plots the distribution of the number of questions per topic. We see that many topics contain only a few questions. Most of these questions belong to a few popular topics. Table 3 lists some of the most popular topics on these sites.

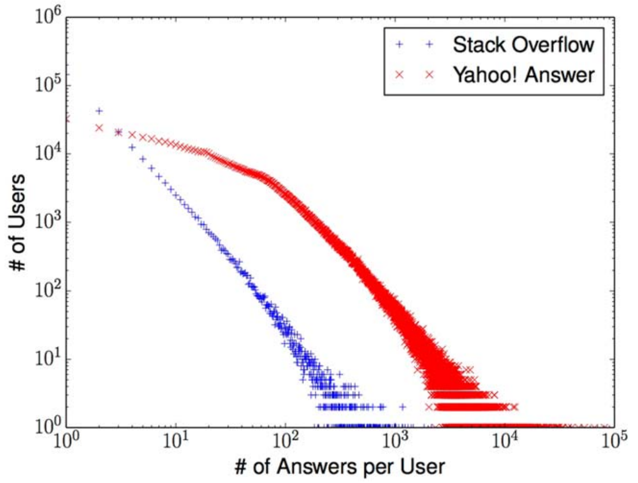

FIG. 2. Distribution of number of answers given by each user. There is a small percentage of users who are very active, whereas many users only give a few answers. A small fraction of users contribute the majority of a CQA site’s contents. [Color figure can be viewed at wileyonlinelibrary.com]

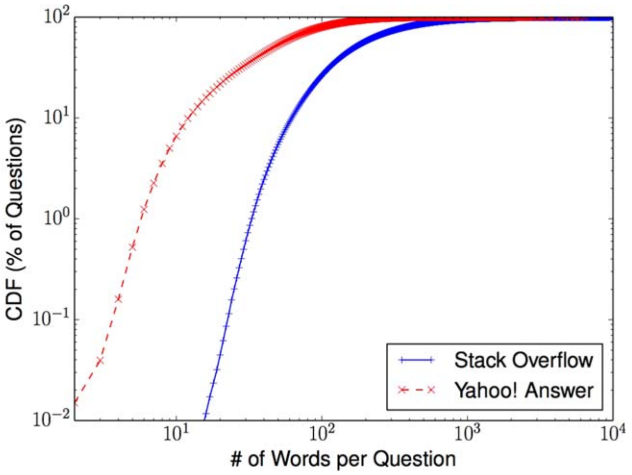

FIG. 3. Distribution of number of words per question in Stack Overflow. Many questions in CQA are short. Questions in Stack Overflow are normally longer than questions in Yahoo! Answers. [Color figure can be viewed at wileyonlinelibrary.com]

Questions in Yahoo! Answers belong to one topic only, whereas questions in Stack Overflow can belong to multiple topics. The number of topics ranges from 1 to 5. Figure 5 plots the number of topics per question. The majority of questions in Stack Overflow belong to multiple topics. Only 11% questions belong to one topic only.

Next, we discuss the results of experiments on these two popular CQA sites.

Experiments

In this section, we describe our experiments with the Yahoo! Answers and Stack Overflow datasets using the QRec algorithm presented earlier. For comparison, we will

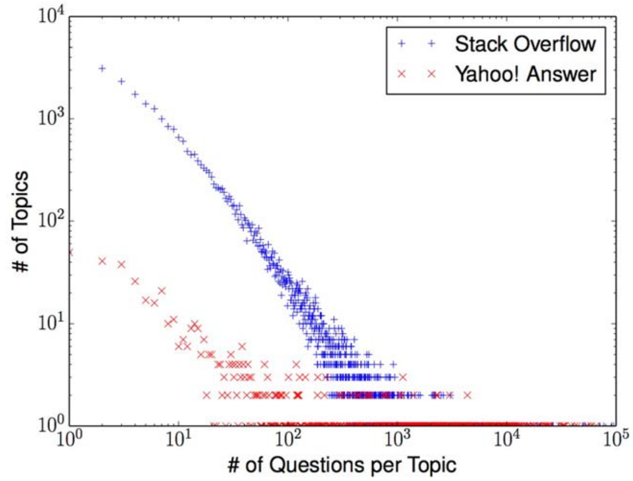

FIG. 4. Distribution of number of questions per topic. Many topics contain only a few questions. There is a small number of topics that contain most of questions asked. [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 3. Most popular topics in two CQA sites.

Stack Oveflow	% question	# Yahoo! Answers	% question
Javascript	3.75	Video & Online Games	2.93
Java	3.40	Current Events	2.67
php	2.89	Polls & Surveys	2.48
C#	2.64	Singles & Dating	2.47
Android	2.46	Psychology	1.99

use a randomized method, active method, probabilistic model, and White's method as the baselines. This section is divided into three parts: experimental setup, results, and discussion of the results.

Experimental Setup

We worked with datasets collected from 9 months of Stack Overflow and 2 years of Yahoo! Answers to evaluate the proposed algorithm. For Stack Overflow, we used the postings from Jan-June 2014 to predict the July-Sept 2014 postings. For Yahoo! Answers, we used the postings in 2008 to predict the answer behavior in 2009.

Data Prepossessing. In the first step, we eliminated questions that were not answered. In the Stack Overflow dataset, 29.5% of questions were not answered. In Yahoo! Answers, 32.7% of questions were unanswered, and thus eliminated. All the content was then converted to lower case. In the final step, we eliminated all stop words from the question content.

Competing Methods. We compared our approach with the following methods:

- White's algorithm: this algorithm considers the raw content of a given question as well as the potential (matched) users' workload (White & Richardson, 2012; White et al., 2011)
- PLSA: a probabilistic question recommendation for CQA (Qu et al., 2009)
- Rand: Potential users are selected randomly

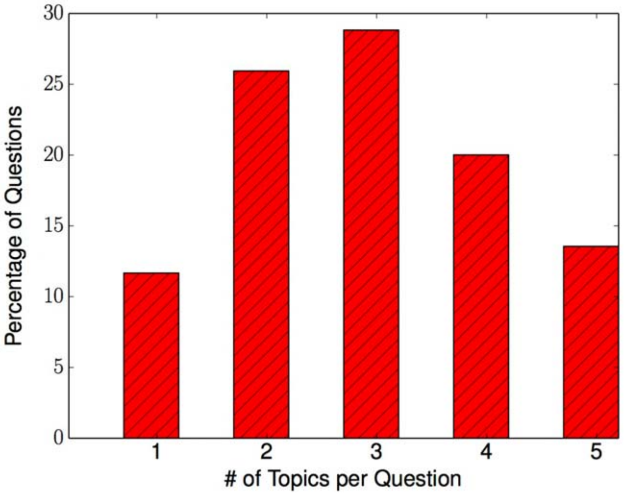

FIG. 5. Distribution of number of topics per question. Most of the questions in Stack Overflow belong to multiple topics. [Color figure can be viewed at wileyonlinelibrary.com]

- Active: Recommend questions to most active users

In White's method, authors computed the *tf-idf* score between a user profile and a question. White's method used raw content of question and raw content written by users without extracting any topic interest. The higher *tf-idf* score indicates that the user is more familiar with the question, which mean that the user will be more willing to give the answer. Furthermore, White's method also balanced the workload between users. Thus, White's method multiplied the *tf-idf* with a decay function. The decay function is defined as:

$$Decay=\begin{cases} 0 & \text{if } \Delta_t \leq \beta \\ 1-e^{\frac{-\Delta_t}{\alpha}} & \text{if } \Delta_t > \beta \end{cases}$$

where Δ_t is the duration since the last answer given by this user, β is the minimum duration between two questions, and α is set to 120/maximum number of questions answered per day. The decay function has a lower value if the user recently answered a question. The purpose of the decay function is to balance the load across users.

In PLSA method, the probability that user u will answer question $q=w_1, w_2, .., w_l$ is computed by $(\prod_{i=1}^{l} P(u, w_i))^{1/l}$.

We changed the size of potential answerers from 1 to 1000 users. The evaluation criteria is whether any users in the potential list answer the question. The larger size of potential answerers makes the correctness ratio higher. In cases where we select the whole community as potential answerers, we can make sure that at least one of them will answer the question. In practice, it is unrealistic to recommend a question to all users because of the community's large size.

Results

The results and evaluation for this problem are based on the correctness of a proposed method for finding the potential answerers. Specifically, the correctness of the method is

DOI: 10.1002/asi

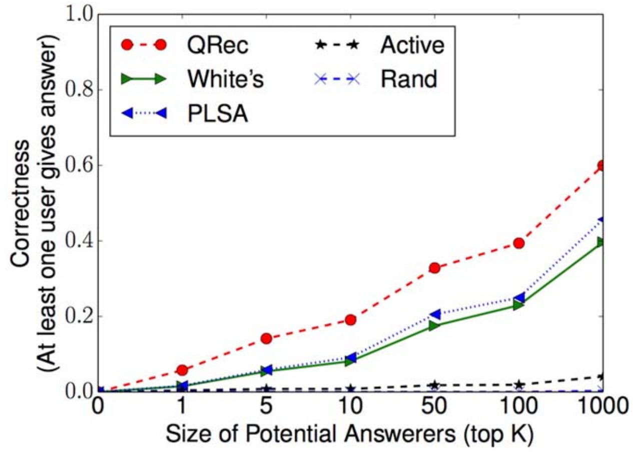

FIG. 6. Stack Overflow: Compare the correctness in selecting potential answerers. Higher is better. The *QRec* achieves the highest correctness. [Color figure can be viewed at wileyonlinelibrary.com]

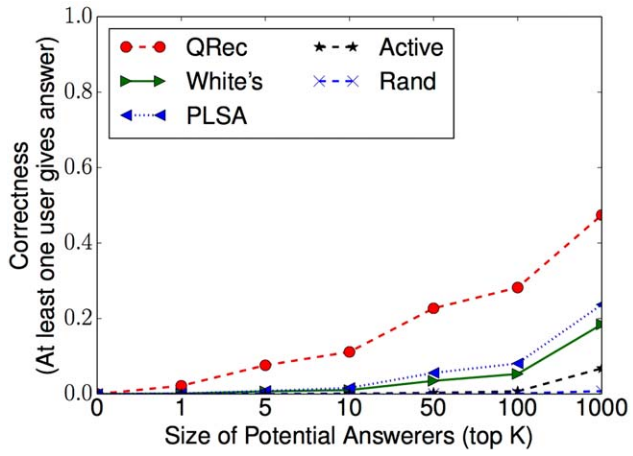

FIG. 7. Yahoo! Answers: Compare the correctness in selecting potential answerers. Higher is better. The *QRec* achieves the highest correctness. [Color figure can be viewed at wileyonlinelibrary.com]

determined by the likelihood that at least one person in a set of identified potential answerers would provide an answer to a question.

Correctness

Figures 6 and 7 plot the correctness of different systems. The *x*-axis is the size of *k* potential answerers. It is trivial that the higher the value of *k*, the higher the chance that at least one answerer will tackle the question. The *y*-axis is the probability that at least one user answers the question. A higher *y*-value is better. *QRec* which includes the content similarity, topic similarity and topics and the user similarity achieves the highest correctness. White's method performs poorly when using the similarity in the content. Because the questions in CQA are often short, computing the similarity between the question and the user profile is a weak signal which caused the poor accuracy of White's method. Our *QRec* achieves the best correctness in both datasets.

Even though there are more than one million users on both sites, the probability that *QRec* can find at least one user to answer the question is higher than 0.5 if the size of potential answerers is 1000. Obviously, if we enlarge the group size, we could increase this correctness. Although this may not seem like a big feat, one needs to consider the enormity of the communities considered here. For example, the results showed that randomly selected answerers will perform very poorly because of the large community size. We also see that including topics when computing the similarity can improve the correctness. This is because many questions in CQA are short. Thus, *tf-idf* of raw content does not reflect user expertise.

Mean Reciprocal Rank

We also measure the mean reciprocal rank (MRR) of the ranking. Let the set of questions be Q. For each question $q_i \in Q$, we find $rank_i$, which is the first correct answerer in the potential list. MRR is the average of the reciprocal ranks of results for all questions in Q as in Equation 7. In a traditional search engine, MRR is associated with the model where the user wants to see one result from the search page. In the case of finding potential answerers, we want to see when we can find the first user who will answer the question we recommend to the list of users. MRR provides us a measure of quality in finding the potential answerer, but it only cares about the single highest-ranked potential answerer. Table 4 compares the MRR scores of our approach, *QRec*, and competing methods. *QRec* achieves the highest MRR score in both data sets. Rand method has a very low MRR score because of many users in these CQA sites. *Active* method, which simply recommends questions to the most active users, also performs poorly because of the diversity of content in the community.

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{\text{rank}_i} \tag{7}$$

The Importance of Each Similarity Feature

Furthermore, we measure the importance of each similarity feature in *QRec*. To evaluate the importance, we drop each similarity metric from our framework and measure the resulting drop in efficacy. Let *Correctness* be the correctness when using all similarity metrics and *Correctness** be the correctness when we drop one similarity metric. The relative drop in correctness is measured as: $Drop = \frac{Correctness - Correctness^*}{Correctness}$. Table 5 lists how much correctness drops when removing the similarity features. The higher the drop, the more important the removed similarity feature. We see that the similarity in content and similarity in topics are more important in *QRec*.

TABLE 4. Compare the MRR of different algorithms. *QRec* achieves highest MRR score in both data sets.

Algorithm	Stack Overflow	Yahoo! Answer
Rand	0.000012	0.000009
Active	0.00019	0.00026
White's	0.033	0.029
PLSA	0.038	0.032
QRec	**0.053**	**0.039**

TABLE 5. The importance of each similarity feature in *QRec* (k =1000). For example, dropping the content similarity in Stack Overflow causes a 14.82% drop in correctness. The higher value indicates the higher importance of this similarity metric. Content and topic similarity are more important than information network.

Similarity metrics	Stack Overflow	Yahoo! Answers
Content	14.82%	19.86%
Topics	16.85%	18.87%
Information network	7.28%	4.31%

The similarity in information network not only has a lower effect but also has a higher computation cost. Computing the similarity in the content is equivalent to calculating the cosine similarity between two vectors (as in Equation 2). In contrast, calculating the similarity in the information network requires us to construct the whole graph and find the random walk with restart (RWR) score of each node. The RWR might be more efficient in a smaller community such as a forum where there are a few thousands of users (Ganu & Marian, 2014).

The Load on Each User

Because the question can be sent to the ordered list of users, some users might be overloaded, whereas others may not receive much. In this experiment, there is a set of questions and the list of potential answerers for each question q is $U=\{u_1, u_2, \ldots, u_k\}$, which is ranked in order. We assign each user's value based on their rank. If the user is the i^{th} in the list, their reciprocal rank (RR) is $\frac{1}{i}$. Then, given the list of Q questions, we can compute the total RR of each user to see the total workload that each user could receive.

Figures 8 and 9 examine the distribution of the users' workload. In the *Rand* method, all the users have small RR values. In the case of the *Active* method, only a small number of users will be potential answerers, but each user's RR is very high. In *QRec* and *White*'s methods, the distributions of RR follow the power law, but the distribution of *QRec* is more skewed. We use linear regression on the log-log scale to estimate the slope of distribution (Chakrabarti & Faloutsos, 2012). On Stack Overflow, the slopes of *QRec* and *White's* methods are −1.53 and −2.89 respectively. Similarly, the slopes of *QRec* and White's methods are respectively −2.22 and −3.01 on Yahoo! Answers. The workload distribution of *QRec* reflects the nature of users' contributions in CQA; for example, the number of answers per user follows the power law.

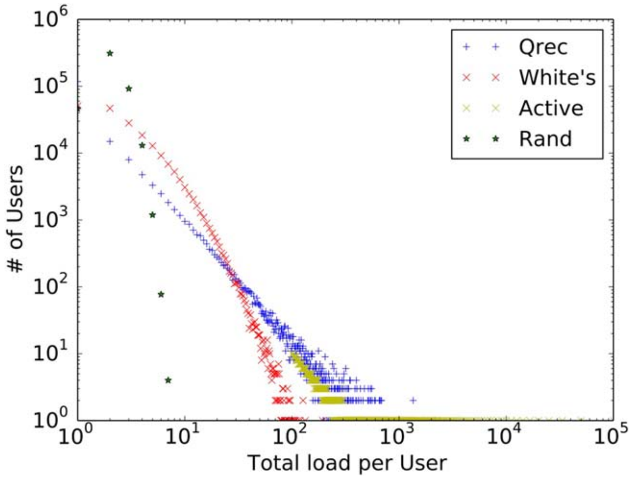

FIG. 8. Stack Overflow: The load distribution on users. *QRec* reflects the "power law" nature of users' activities on CQA. [Color figure can be viewed at wileyonlinelibrary.com]

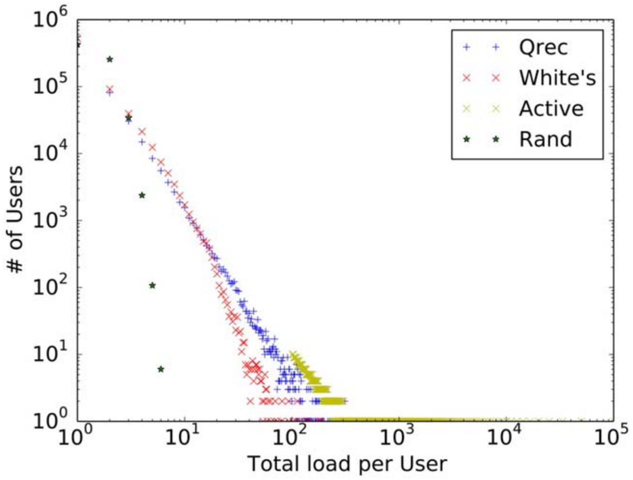

FIG. 9. Yahoo! Answers: The load distribution on users. *QRec* reflects the "power law" nature of users' activities on CQA. [Color figure can be viewed at wileyonlinelibrary.com]

Discussion

We will discuss how and why our method works. The results showed that a user's topic interest and activity level (or social capital) are useful features when finding potential answerers. We will explain the reason of using the $log(activity\_level)$ and the importance of using the topic interests.

To calculate the score between a user profile and a question, we used $log(activity\_level)$ instead of the *activity level* because of the skewed distribution of user activity. Figure 2 shows that some users answer a larger number of questions. These active users will always have a very high score if multiplying with *activity level* even if *tf-idf* is low. If we do not use the $log(activity\_level)$, the most active users will get high scores in majority of new questions. Thus, using the *log* value is important.

In addition, applying users' topical expertise is better than using the raw content because of the brevity of questions in CQA. Many CQA sites, including those used in the work reported here, contain massive amounts of highly diverse content. Our analysis shows that there are thousands of topics in Stack Overflow and Yahoo! Answers. Users are normally interested in a small number of topics, and they have a higher tendency to answer a question that pertains to topics with which they are familiar. To confirm this hypothesis, we did a test to measure a user's *answer-lift*. Given a user *u*, a question *q* belongs to topic *T*, *answer-lift* as defined in Equation 8.

$$AnswerLift(u,q) = \frac{P(u\,answers\,q \mid u\,answered\,T\,before)}{P(u\,answers\,q \mid u\,not\,answered\,T\,before)} \tag{8}$$

In general, the *answer-lift* measures the ratio of the proportion of users who answer questions that pertain to their topics of expertise to those who answer unrelated questions. We randomly picked 10,000 answers in the site and found that the answer lifts were 2.81 and 2.37 in Stack Overflow and Yahoo! Answers respectively. These results support our method's effectiveness in using the interest history to predict the future actions.

Conclusion

In the work reported here, we presented a large-scale study on two of the most successful CQA sites: Yahoo! Answers and Stack Overflow. Our analysis highlighted the similarities and differences between these two sites, as they serve different purposes and communities. We also addressed an important problem within CQA by proposing a new method, *QRec*, to find potential question answerers. The results showed that our method can achieve high correctness in both platforms. This could help CQA sites forward questions to suitable users. We expect that finding the correct answerers would allow a question to be answered more quickly and accurately. In both scenarios, users should be more satisfied with the CQA site, increase their engagement, and ultimately contribute to a larger and healthier overall community.

We expect that using topic interest can help us solve other important problems that concern user retrieval in CQA, such as grouping users or finding special types of users. For example, we could use the *QRec* algorithm proposed here with a slight modification to find people who are likely to quit the community, as they are either losing interest in the topics covered by the community or have not found enough activities that pertain to their interests. Another example is finding a moderator in CQA. Because of the popularity of its community, a CQA site must have a set of users who will monitor others' activities. Topic interest is a strong indicator of these potential moderators.

This work is not without its limitations. In our experiments, we only evaluated our method using questions that received at least one answer. Because of the limitation of our datasets, it is not possible to test our method on unanswered questions. In this work, we do not investigate why certain questions are not answered. Understanding the answerability of questions is studied in several works (Dror, Maarek, & Szpektor, 2013; L. Yang et al., 2011; Shah et al., 2012). Questions often go unanswered because they are spam, duplicates, or annoying. Furthermore, our method did not consider users' temporal activity, such as changes in topic interest over time. Different forms of data and experiments are needed to address such issues.

In our current work, we only consider the list of questions answered when constructing user profiles. A profile based on answering activity can be considered a user's *expertise*. Future work will incorporate the list of questions asked into user profiles. A profile based on asking activity is considered a user's *interest*. We expect that combining both user expertise and user interest will provide a better view of CQA.

References

Adamic, L. A., Zhang, J., Bakshy, E., & Ackerman, M. S. (2008). Knowledge sharing and yahoo answers: Everyone knows something. In International Conference on World Wide Web (WWW) (pp. 665–674). Beijing, China.

Andreas, J., Rohrbach, M., Darrell, T., & Klein, D. (2016). Learning to compose neural networks for question answering. In The Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, (NAACL) (pp. 1545–1554). San Diego, CA.

Arora, P., Ganguly, D., & Jones, G. J. F. (2015). The good, the bad and their kins: Identifying questions with negative scores in stackoverflow. In IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) (p. 1232–1239). Paris, France.

Bian, J., Liu, Y., Agichtein, E., & Zha, H. (2008). Finding the right facts in the crowd: Factoid question answering over social media. In International Conference on World Wide Web (WWW) (pp. 467–476). Beijing, China.

Bishop, C. M. (2006). Pattern recognition and machine learning (Information science and statistics). Heidelberg, Germany: Springer.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. J. Mach. Learn. Res., 3, 993–1022.

Carmel, D., Mejer, A., Pinter, Y., & Szpektor, I. (2014). Improving term weighting for community question answering search using syntactic analysis. In ACM International Conference on Conference on Information and Knowledge Management (CIKM) (pp. 351–360). Shanghai, China.

Chakrabarti, D., & Faloutsos, C. (2012). Graph mining: Laws, tools, and case studies. San Rafael, CA: Morgan & Claypool.

Chen, Z., Gao, B., Zhang, H., Zhao, Z., Liu, H., & Cai, D. (2017). User personalized satisfaction prediction via multiple instance deep learning. In Proceedings of the 26th International Conference on World Wide Web (WWW) (pp. 907–915). Perth, Australia.

Dror, G., Koren, Y., Maarek, Y., & Szpektor, I. (2011). I want to answer; who has a question? Yahoo! answers recommender system. In ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD) (pp. 1109–1117). San Diego, CA.

Dror, G., Maarek, Y., & Szpektor, I. (2013). Will my question be answered? predicting "question answerability" in community question-answering sites. In European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD)(Vol. 8190, p. 499–514). Prague, Czech Republic.

Ganu, G., & Marian, A. (2014). Personalizing forum search using multi-dimensional random walks. In International AAAI Conference on Web and Social Media (ICWSM) (pp. 140–149). Oxford, England.

Gkotsis, G., Stepanyan, K., Pedrinaci, C., Domingue, J., & Liakata, M. (2014). It's all in the content: State of the art best answer prediction based on discretisation of shallow linguistic features. In ACM Web Science Conference (pp. 202–210). Bloomington, IN.

Gollapalli, S. D., Mitra, P., & Giles, C. L. (2013). Ranking experts using author-document-topic graphs. In ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL) (pp. 87–96). Indianapolis, IN.

Hoffman, M. D., Blei, D. M., & Bach, F. R. (2010). Online learning for latent dirichlet allocation. In Conference on Neural Information Processing Systems (NIPS) (p. 856–864). Vancouver, Canada.

Iyyer, M., Boyd-Graber, J. L., Claudino, L. M. B., Socher, R., & Daumé, H. III. (2014). A neural network for factoid question answering over paragraphs. In Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 633–644). Doha, Qatar.

Jurczyk, P., & Agichtein, E. (2007). Discovering authorities in question answer communities by using link analysis. In ACM International Conference on Conference on Information and Knowledge Management (CIKM) (pp. 919–922). Lisbon, Portugal.

Le, L. T., & Shah, C. (2016). Retrieving rising stars in focused community question-answering. In Asian Conference on Intelligent Information and Database Systems (ACIIDS) (pp. 25–36). Da Nang, Vietnam.

Le, L. T., Shah, C., & Choi, E. (2016). Evaluating the quality of educational answers in community question-answering. In ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL) (pp. 25–36). Salt Lake City, UT.

Le, L. T., Shah, C., & Choi, E. (2017). Bad users or bad content? breaking the vicious cycle by finding struggling students in community question-answering. In ACM SIGIR Conference on Human Information Interaction and Retrieval (CHIIR) (pp. 165–174). Tokyo, Japan.

Li, B., Jin, T., Lyu, M. R., King, I., & Mak, B. (2012). Analyzing and predicting question quality in community question answering services. In International Conference on World Wide Web (WWW) (pp. 775–782). Lyon, France.

Li, G., Zhu, H., Lu, T., Ding, X., & Gu, N. (2015). Is it good to be like wikipedia? Exploring the trade-offs of introducing collaborative editing model to Q&A sites. In ACM Conference on Computer-Supported Cooperative Work and Social Computing (CSCW) (pp. 1080–1091). Vancouver, Canada.

Luo, L., Wang, F., Zhou, M. X., Pan, Y., & Chen, H. (2014). Who have got answers? Growing the pool of answerers in a smart enterprise social qa system. In International Conference on Intelligent User Interfaces (IUI) (pp. 7–16). Haifa, Israel.

Macina, J., Srba, I., Williams, J. J., & Bielikova, M. (2017). Educational question routing in online student communities. In Proceedings of the Eleventh ACM Conference on Recommender Systems (RecSys) (pp. 47–55). Como, Italy.

Manning, C. D., Raghavan, P., & Schutze, H. (2008). Introduction to information retrieval. New York: Cambridge University Press.

Nam, K. K., Ackerman, M. S., & Adamic, L. A. (2009). Questions in, knowledge in? A study of naver's question answering community. In SIGCHI Conference on Human Factors in Computing Systems (CHI) (pp. 779–788). Boston, MA.

Pal, A., Wang, F., Zhou, M. X., Nichols, J., & Smith, B. A. (2013). Question routing to user communities. In ACM International Conference on Conference on Information and Knowledge Management (CIKM) (pp. 2357–2362). San Francisco, CA.

Preece, J., Nonnecke, B., & Andrews, D. (2004). The top five reasons for lurking: Improving community experiences for everyone. Computers in Human Behavior, 20(2), 201 - 223.

Qiu, X., & Huang, X. (2015). Convolutional neural tensor network architecture for community-based question answering. In International Joint Conference on Artificial Intelligence (IJCAI) (pp. 1305–1311). Buenos Aires, Argentina.

Qu, M., Qiu, G., He, X., Zhang, C., Wu, H., Bu, J., & Chen, C. (2009). Probabilistic question recommendation for question answering communities. In International Conference on World Wide Web (WWW) (pp. 1229–1230). Madrid, Spain.

Ravi, S., Pang, B., Rastogi, V., & Kumar, R. (2014). Great question! question quality in community Q&A. In International AAAI Conference on Web and Social Media (ICWSM) (pp. 426–435). The AAAI Press.

Shah, C., Oh, S., & Oh, J. S. (2009). Research agenda for social Q&A. Library & Information Science Research, 31(4), 205–209.

Shah, C., Radford, M., Connaway, L., Choi, E., & Kitzie, V. (2012). How much change do you get from 40$? Analyzing and addressing failed questions on social Q&A. In Annual Meeting of the Association for Information Science and Technology (ASIST) (pp. 1–10). Baltimore, MD.

Srba, I., Grznar, M., & Bielikova, M. (2015). Utilizing non-qa data to improve questions routing for users with low qa activity in cqa. In IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) (pp. 129–136). Paris, France.

Surowiecki, J. (2005). The wisdom of crowds. New York: Anchor.

Tong, H., Faloutsos, C., & Pan, J.-Y. (2006). Fast random walk with restart and its applications. In IEEE International Conference on Data Mining (ICDM) (p. 613–622). Hong Kong, China.

Wang, G., Gill, K., Mohanlal, M., Zheng, H., & Zhao, B. Y. (2013). Wisdom in the social crowd: An analysis of quora. In International Conference on World Wide Web (WWW) (pp. 1341–1352). Rio de Janeiro, Brazil.

Wang, X.-J., Tu, X., Feng, D., & Zhang, L. (2009). Ranking community answers by modeling question-answer relationships via analogical reasoning. In ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR) (pp. 179–186). Boston, MA.

White, R. W., & Richardson, M. (2012). Effects of expertise differences in synchronous social Q&A. In ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR) (pp. 1055–1056). Portland, OR.

White, R. W., Richardson, M., & Liu, Y. (2011). Effects of community size and contact rate in synchronous social Q&A. In SIGCHI Conference on Human Factors in Computing Systems (CHI) (pp. 2837–2846). Vancouver, Canada.

Wu, H., Wu, W., Zhou, M., Chen, E., Duan, L., & Shum, H.-Y. (2014). Improving search relevance for short queries in community question answering. In ACM International Conference on Web Search and Data Mining (WSDM) (pp. 43–52). New York, NY.

Xue, X., Jeon, J., & Croft, W. B. (2008). Retrieval models for question and answer archives. In ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR) (pp. 475–482). Singapore.

Yan, Z., & Zhou, J. (2015). Optimal answerer ranking for new questions in community question answering. Information Processing & Management, 51(1), 163–178.

Yang, L., Bao, S., Lin, Q., Wu, X., Han, D., Su, Z., & Yu, Y. (2011). Analyzing and predicting not-answered questions in community-based question answering services. In AAAI Conference on Artificial Intelligence (AAAI) (pp. 1273–1278). San Francisco, CA.

Yang, P., & Fang, H. (2013). Opinion-based user profile modeling for contextual suggestions. In ACM International Conference on the Theory of Information Retrieval (ICTIR) (pp. 80–83). Copenhagen, Denmark.

Yao, Y., Tong, H., Xie, T., Akoglu, L., Xu, F., & Lu, J. (2014). Joint voting prediction for questions and answers in cqa. In IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) (pp. 340–343). Paris, France.

Yarosh, S., Matthews, T., & Zhou, M. (2012). Asking the right person: Supporting expertise selection in the enterprise. In SIGCHI Conference on Human Factors in Computing Systems (CHI) (pp. 2247–2256).

DOI: 10.1002/asi

Yin, X., Han, J., & Yu, P. S. (2008). Truth discovery with multiple conflicting information providers on the web. In IEEE Transactions on Knowledge and Data Engineering (TKDE) (pp. 796–808).

Zhang, J., Kong, X., Luo, R. J., Chang, Y., & Yu, P. S. (2014). Ncr: A scalable network-based approach to co-ranking in question-and-answer sites. In ACM International Conference on Conference on Information and Knowledge Management (CIKM) (pp. 709–718).

Zhang, J., Tang, J., & Li, J.-Z. (2007). Expert finding in a social network. In International Conference on Database Systems for Advanced Applications (DASFAA) (p. 1066–1069). Springer.

Zhou, Z.-M., Lan, M., Niu, Z.-Y., & Lu, Y. (2012). Exploiting user profile information for answer ranking in cqa. In International Conference on World Wide Web (WWW) Companion (pp. 767–774).