SocialTransfer: Transferring Social Knowledge for Cold-Start Crowdsourcing

Zhou Zhao*, James Cheng, Furu Wei, Ming Zhou, Wilfred Ng, Yingjun Wu

Department of Computer Science and Engineering, Hong Kong University of Science and Technology
Department of Computer Science and Engineering, The Chinese University of Hong Kong
Microsoft Research Asia, Beijing, China
School of Computing, National University of Singapore, Singapore
{zhaozhou, wilfred}@cse.ust.hk, jcheng@cse.cuhk.edu.hk,
{fuwei,mingzhou}@microsoft.com, yingjun@comp.nus.edu.sg

ABSTRACT

An essential component of building a successful crowdsourcing market is effective task matching, which matches a given task to the right crowdworkers. In order to provide high-quality task matching, crowdsourcing systems rely on past task-solving activities of crowdworkers. However, the average number of past activities of crowdworkers in most crowdsourcing systems is very small. We call the workers who have only solved a small number of tasks cold-start crowdworkers. We observe that most of the workers in crowdsourcing systems are cold-start crowdworkers, and crowdsourcing systems actually enjoy great benefits from cold-start crowdworkers. However, the problem of task matching with the presence of many cold-start crowdworkers has not been well studied. We propose a new approach to address this issue. Our main idea, motivated by the prevalence of online social networks, is to transfer the knowledge about crowdworkers in their social networks to crowdsourcing systems for task matching. We propose a SocialTransfer model for cold-start crowdsourcing, which not only infers the expertise of warm-start crowdworkers from their past activities, but also transfers the expertise knowledge to cold-start crowdworkers via social connections. We evaluate the SocialTransfer model on the well-known crowdsourcing system Quora, using knowledge from the popular social network Twitter. Experimental results show that, by transferring social knowledge, our method achieves significant improvements over the state-of-the-art methods.

Categories and Subject Descriptors
H.2.8 [Database Management]: Database Applications

*Part of the work was done when the author was an intern at Microsoft Research Asia

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CIKM ’14, November 3–7, 2014, Shanghai, China.
Copyright 2014 ACM 978-1-4503-2598-1/14/11 ...$15.00.
http://dx.doi.org/10.1145/2661829.2661871.

Figure 1: Cold-Start Crowdworkers in Quora

General Terms
Algorithms, Design, Experiments

Keywords
Crowdsourcing, Social Network

1. INTRODUCTION

The benefits of crowdsourcing have been widely recognized today [26, 27, 1] and we have witnessed many successful systems such as Amazon Mechanical Turk [21] for generic tasks, LabelMe [11] for image annotation, and Quora [28] for question answering. As crowdsourcing systems are gaining more and more popularity, their rapid growth has also made it increasingly difficult for crowdworkers to find tasks that are suitable for them to perform in the market.

In this paper, we study the problem of task matching query in crowdsourcing systems, which is to find the right crowdworkers to solve a given task. In order to provide high-quality task matching, crowdsourcing systems rely on histories of past task-solving activities of crowdworkers. However, when new crowdworkers join a system, no prior activity can be observed. In fact, a vast majority of existing crowdworkers in a real-life crowdsourcing system, including many that have joined the system for a relatively long period of time, do not have sufficient prior task-solving records for inferring high-quality task matching. To give an example, we aggregate the question-solving activities of the Quora crowdworkers in Figure 1(a), from which we can observe that the participation in the question-answering activities of most crowdworkers fall into the Long Tail part of the power-law curve, i.e., the majority of the crowdworkers have performed less than 10 tasks. These crowdworkers who have
only performed a small number of tasks are called cold-start crowdworkers.

Interestingly, the crowdsourcing market also enjoys the great benefits brought by cold-start crowdworkers. We have observed the excellent performance of cold-start crowdworkers on an extensive number of tasks. However, the task matching for cold-start crowdworkers still remains a challenging problem.

To attain high-quality task matching, we need to find sufficient knowledge for cold-start crowdworkers, but knowledge about them in the crowdsourcing system is scarce. Fortunately, in this big data era, we may find other sources to obtain the required knowledge. In particular, we propose a novel idea of transferring knowledge from online social networks for crowdsourcing task matching. Indeed, with the prevalence of online social networks today, it is not difficult to find the activities of crowdworkers, including both warm-start and cold-start crowdworkers, as well as their connections, in various online social networks (e.g., Facebook, Twitter, etc.). The crowdworkers broadcast messages, post blogs, and follow the activities of other users in social networks. Thus, a vast amount of data related to crowdworkers, such as workers’ tweet contents and worker-to-worker social connection, can be obtained. For example, we observe that more than one third of the crowdworkers in Quora has a Twitter account and crowdworkers are followed by other crowdworkers.

How can we transfer rich information hidden in online social networks to achieve high-quality task matching in crowdsourcing systems?

We focus on the social activities of the Quora crowdworkers in a popular social networking site, Twitter, and propose a SocialTransfer graph in order to transfer the social knowledge of both cold-start and warm-start crowdworkers to solve task matching. We extract the Twitter accounts of the crowdworkers from their profiles in the Quora system. We mainly utilize two types of social knowledge of the crowdworkers, namely worker’s tweets and worker-to-worker social connections. We model the social interests of the crowdworkers from their tweets and infer the similarity of the crowdworkers by their worker-to-worker connections.

We illustrate our main idea using an example of a SocialTransfer graph in Figure 2. We extracted four task matching queries \( Q = \{q_1, q_2, q_3, q_4\} \) and five crowdworkers \( V = \{v_1, v_2, \ldots, v_5\} \) from Quora. First, the past task-solving activities of the crowdworkers on the three task matching queries are indicated by the edges between queries \( Q \) and crowdworkers \( V \) in the rectangle on the left in Figure 2. The quality scores of the answers given by the crowdworkers for each query in \( Q \) are shown in Table S. A score of 1 indicates a positive performance of the crowdworker on the task, while a score of -1 indicates a negative performance of the crowdworker on the task. The question mark means that the crowdworker did not participate in the task. The quality score is based on the thumb-ups and thumb-downs of the users who posted the tasks or viewed the answers in Quora.

Second, we show the social following relations (in Twitter) of the crowdworkers in the rectangle on the right in Figure 2. Each crowdworker in Twitter is associated with the worker’s tweets and social relations. A crowdworker can follow other crowdworkers or other Twitter users. The process of building a SocialTransfer graph is to construct a heterogeneous graph by integrating the social network graph of the crowdworkers into the task assignment bipartite graph.

The main idea of using a SocialTransfer graph to address the problem of task matching for cold-start crowdworkers is as follows. First, for cold-start crowdworkers with sufficient tweet information, we infer the expertise of the crowdworkers for different types of tasks by transferring the knowledge of the tweets. Second, for cold-start crowdworkers with limited tweet information, we infer their expertise by transferring the social knowledge from some warm-start crowdworkers. For example, \( v_1 \) is a cold-start crowdworker in Figure 2. Apart from inferring the expertise of \( v_1 \) from his tweets, we can also transfer the knowledge from \( v_3 \), who is a warm-start crowdworker, if \( v_1 \) and \( v_3 \) share many common followings in Twitter.

Task Matching for Cold-Start Crowdsourcing. There exists some work addressing the cold-start problem in user-item recommendation systems [23, 13, 24, 14, 40, 34, 10, 15]. However, most of them are not applicable in addressing the problem of task matching in cold-start crowdsourcing. Even though matching a task to the right crowdworker is analogous to recommending an item to a user, there are fundamental differences between them. First, the existing work incorporates the social relations of users to recommend the best item among a set of items to a user. In our work, we map a question to the right crowdworker but there is no relation among the questions; thus, the existing cold-start recommendation techniques cannot be applied to solve our problem. Second, existing recommendation techniques focus on recommending existing items to users, while task match-
ing in crowdsourcing aims to find the right crowdworkers to solve new tasks.

The main contributions of our work are summarized as follows:

- We propose a new approach to address the problem of task matching in cold-start crowdsourcing by transferring the knowledge embedded in online social networks to crowdsourcing systems.
- We develop a SocialTransfer model for Cold-start crowdsourcing (STC) that not only infers the expertise of the warm-start crowdworkers, but also transfers the knowledge to the cold-start crowdworkers via social connections.
- We devise an efficient inference algorithm to build our STC model and propose a task matching algorithm in crowdsourcing systems based on STC.
- We evaluate the proposed SocialTransfer model on a real-life crowdsourcing system Quora. We demonstrate that, by transferring social knowledge, our approach effectively addresses the task matching problem in cold-start crowdsourcing and significantly outperforms the state-of-the-art task matching techniques.

The rest of the paper is organized as follows. Section 2 introduces the notations and formulates the problem. We propose a novel model that transfers social knowledge for cold-start crowdsourcing in Section 3. We report the experimental results in Section 4 and survey the related work in Section 5. We conclude the paper in Section 6.

2. NOTATIONS AND PROBLEM DEFINITION

We first define some frequently used notations and then formulate the problem of task matching in crowdsourcing systems. The summary of the notation is given in Table 1.

**Definition 1. (Task Matching Queries Q)** The set Q contains a collection of existing processed task matching queries in a crowdsourcing system. We denote by Q the set \( \{q_1, q_2, \ldots, q_M\} \), where \( q_i \) is a task matching query and \( M \) is the number of queries. For each query \( q_i \), we denote its related task description by \( w_{q_i} \), which is a bag of words.

For example, the description of a task matching query in Quora can be a posted question, while the description of a query in Amazon Mechanical Turk can be several instructions.

**Definition 2. (Crowdworkers V)** The set V consists of all the crowdworkers in a crowdsourcing system. We denote by V the set \( \{v_1, v_2, \ldots, v_N\} \), where \( v_i \) is a crowdworker and \( N \) is the number of crowdworkers.

**Definition 3. (Task Matching Graph G_Q)** The bipartite task matching graph, \( G_Q = (Q \cup V, A) \), stores the existing matchings of queries Q and crowdworkers V in a crowdsourcing system. The set of vertices, \( Q \cup V \), is the union of queries and crowdworkers. The set of edges, A, is the set of matchings between queries in Q and crowdworkers in V.

For example, the graph \( G_Q \) for the matchings between queries \( Q = \{q_1, q_2, q_3, q_4\} \) and crowdworkers \( V = \{v_1, v_2, \ldots, v_5\} \) is given in Figure 2. The edge \( A(q_i, v_j) = 1 \) if the task in query \( q_i \) is assigned to crowdworker \( v_j \), otherwise \( A(q_i, v_j) = 0 \).

**Table 1: Definition of Notations**

<table>
<thead>
<tr>
<th>Group</th>
<th>Notation</th>
<th>Notation Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>( G_Q )</td>
<td>A SocialTransfer graph</td>
</tr>
<tr>
<td></td>
<td>( Q )</td>
<td>Matching query set</td>
</tr>
<tr>
<td></td>
<td>( V )</td>
<td>Crowdworker set</td>
</tr>
<tr>
<td></td>
<td>( A )</td>
<td>Task matchings</td>
</tr>
<tr>
<td></td>
<td>( S )</td>
<td>Feedback scores</td>
</tr>
<tr>
<td></td>
<td>( B )</td>
<td>Common followings</td>
</tr>
<tr>
<td></td>
<td>( W )</td>
<td>Words in queries and tweets</td>
</tr>
<tr>
<td>Model</td>
<td>( \Theta )</td>
<td>Dirichlet distribution</td>
</tr>
<tr>
<td></td>
<td>( \lambda_B, \lambda_V )</td>
<td>Priors</td>
</tr>
<tr>
<td></td>
<td>( \tilde{\Theta} )</td>
<td>Normal distribution</td>
</tr>
<tr>
<td></td>
<td>( \tilde{\Lambda} )</td>
<td>Latent query factor</td>
</tr>
<tr>
<td></td>
<td>( \tilde{\Theta}_Q )</td>
<td>Latent crowdworker factor</td>
</tr>
<tr>
<td></td>
<td>( \tilde{\Theta}_V )</td>
<td>Topic proportions</td>
</tr>
</tbody>
</table>

**Definition 4. (Feedback Score S)** The scores \( S \) are the feedbacks given by users and are used for measuring the quality of the tasks performed by crowdworkers. For each task matching \( (q_i, v_j) \) in graph \( G_Q \), there is a feedback score \( S(q_i, v_j) \) for it.

The feedback score \( S \) can be the satisfactoriness rate of the task sponsor (i.e., \( S(q_i, v_j) \in [0, 1] \)) in Amazon Mechanical Turk or the thumb-ups/downs for the answers by crowdworkers in Quora (i.e., \( S(q_i, v_j) \in Z \)).

Let \( F(v_j) \) be the set of followings of a crowdworker \( v_j \) in a social network, i.e., the user \( v_j \) follows the set of users in \( F(v_j) \) in the social network.

**Definition 5. (Social Graph G_V)** The social graph \( G_V = (V, B) \) stores the activities of crowdworkers in a social network such as workers’ tweet content and worker-to-worker social connection. For each crowdworker \( v_i \in V \), we denote its tweet content by \( w_{v_i} \), which is a bag of words. For each edge \( (v_i, v_j) \in B \), it indicates that crowdworkers \( v_i \) and \( v_j \) share some common followings in the social network, i.e., \( F(v_i) \cap F(v_j) \neq \emptyset \). We set the weight of the edge \( (v_i, v_j) \) to be \( |F(v_i) \cap F(v_j)| \), i.e., the ratio of the common followings between \( v_i \) and \( v_j \) to their total followings in the social network.

Note that the weight of the edges in \( B \) is within the range \([0, 1]\).

Now, we introduce a SocialTransfer graph that transfers the social knowledge of crowdworkers in a social network to address the cold-start crowdsourcing problem, in order to improve the quality of task matching.

**Definition 6. (SocialTransfer Graph G_ST)** The SocialTransfer graph \( G_{ST} = (Q \cup V, A \cup B) \) models the activities of crowdworkers in both the task matching graph \( G_Q \) and the social graph \( G_V \). The set of vertices in \( G_{ST} \) is the union of the existing task matching queries Q and crowdworkers V. The set of edges in \( G_{ST} \) contains both the set of existing matchings modeled in \( A \) and the set of pairwise common followings of crowdworkers modeled in \( B \).

Note that \( G_{ST} \) is not (and need not be) a tripartite graph, though we show a tripartite graph for the example in Figure 2 for simplicity of discussion there.
We now define the problem as follows.

**Problem 1.** *(Task Matching Query in Cold-Start Crowdsourcing)* Consider a crowdsourcing system with many cold-start crowdworkers and a SocialTransfer graph $G_{ST} = (Q \cup V, A \cup B)$. Given a new task matching query $q_i$, find the crowdworkers with high predicted feedback score to solve the given task.

### 3. SOCIALTRANSFER MODEL

In this section, we propose our model, STC, to tackle the problem of task matching in cold-start crowdsourcing. We first introduce the background of task matching in crowdsourcing and present the idea of transferring social knowledge with cold-start crowdsourcing. Then, we summarize our proposed STC model and devise an effective inference algorithm to build STC model. Finally, we present a task matching algorithm in crowdsourcing.

#### 3.1 Background of Task Matching

We first give the definitions of latent query factor and latent crowdworker factor for task matching queries $\vec{Q}$ and crowdworkers $\vec{V}$. Then, we propose a score generative model for the already processed tasks based on latent query factor and latent crowdworker factor.

**Definition 7.** *(Latent Query Factor)* Given the dimension of latent space $K$, the latent factor of the queries in crowdsourcing systems is denoted by a $K$-dimensional vector of real values. For each task matching query $q_i \in Q$, we denote by $\vec{q}_i$ the latent query factor, which is given by

$$\vec{q}_i = \left( q_{i,1}, \ldots, q_{i,K} \right) \in \mathbb{R}^K.$$  

In this paper, we consider that the prior distribution of the latent query factor is based on the latent topic of the task description $w_q$. We employ a well-known latent Dirichlet allocation (LDA) [2] to discover the latent topic of the task description, from which we choose the normal distribution for the prior distribution of latent query factor, given by

$$\vec{q}_i \sim N(\theta_{q_i}, \lambda_{q_i}^{-1}),$$

where $N(\cdot)$ is a multivariate normal distribution and $\theta_{q_i}$ is the latent topic of question $q_i$. We consider that $\theta_{q_i}$ is its mean and $\lambda_{q_i}^{-1}$ is standard deviation.

**Definition 8.** *(Latent Crowdworker Factor)* The latent factor of the crowdworkers in crowdsourcing systems is denoted by a $K$-dimensional vector of real values. For each crowdworker $v_j \in V$, we denote by $\vec{v}_j$ the latent crowdworker factor, which is given by

$$\vec{v}_j = \left( v_{j,1}, \ldots, v_{j,K} \right) \in \mathbb{R}^K.$$  

We also choose the normal distribution as the prior for latent crowdworker factor. For the crowdworkers with activities in social networks, the prior distribution of them is given by

$$\vec{v}_j \sim N(\theta_{v_j}, \lambda_{v_j}^{-1}),$$

where $\theta_{v_j}$ is the latent topic of $v_j$’s tweets and $\lambda_{v_j}^{-1}$ is standard deviation. For other crowdworkers without activities, the prior distribution of them is given by

$$\vec{v}_j \sim N(0, \lambda_{v_j}^{-1}).$$

Using the definitions of latent query factor and latent crowdworker factor, we introduce the score generative model for processed tasks below.

**Definition 9.** *(Score Generative Model)* Given latent query factor $\vec{Q}$ and latent crowdworker factor $\vec{V}$, the feedback score on processed tasks is given by

$$S \sim N(\vec{Q}^T \vec{V}, \lambda_{S}^{-1})$$

where $N(\cdot)$ is a multivariate normal distribution with mean $\vec{Q}^T \vec{V}$ and standard deviation $\lambda_{S}^{-1}$.

For the score of each processed task $(q_i, v_j)$, we consider that its feedback score is generated by

$$S(q_i, v_j) \sim N(\vec{q}_i^T \vec{v}_j, \lambda_{S}^{-1}) = N(\sum_{k=1}^{K} q_{i,k} v_{j,k}, \lambda_{S}^{-1}). \quad (2)$$

We now justify the score generative model as follows: We generate the latent query factor by the latent topic of the task description and the latent crowdworker factor by the latent topic of workers’ tweets. Thus, we consider that the latent crowdworker factor is the topical expertise of the crowdworkers. Therefore, the score of the processed task is proportional to the dot-product of latent query factor and latent crowdworker factor.

#### 3.2 Knowledge Transfer via Social Network

We now introduce the idea of transferring social knowledge to improve the inference of latent crowdworker factor. We mainly utilize two types of social knowledge, workers’ tweets and worker-to-worker social connections.

We consider that the latent topic of workers’ tweets represents the topical interests of the crowdworkers. For the cold-start crowdworkers, it is difficult to infer the latent crowdworker factor from their few task-solving activities in crowdsourcing systems. Thus, we transfer the discovered topical interests as an external source to improve the inference quality of the latent factor of the cold-start crowdworkers by Formula 1.

We also notice that the crowdworkers with similar topical interests have a lot of common following users or entities in social networks. Therefore, we utilize the common followings of the crowdworkers to regularize the inference of topical interests.

Consider two crowdworkers $v_i$ and $v_j$ in a social network. If both of them follow a number of common followings (i.e., the weight of the edge $(v_i, v_j)$, $B_{(v_i, v_j)}$, in the social graph $G_V$ is large), then their topic interests are similar (i.e., $|\theta_{v_i} - \theta_{v_j}|$ is small). Based on this assumption, we propose a prior distribution for the topic interests of crowdworkers with social regularization, which is a product of the Dirichlet distribution and the Normal distributions. The prior distribution for the topic interests of crowdworker $v_i$ is given by

$$\theta_{v_i} \sim \text{Dir}(\alpha) \prod_{j \neq i} N(\theta_{v_i} - \theta_{v_j} | 0, B_{(v_i, v_j)}^{-1})^{\lambda_B}, \quad (3)$$

where the product of normal distributions of the topic interests of crowdworker $v_i$ and other crowdworkers $v_j$ are used for regularization.
3.3 Summary of STC Model

In this subsection, we present the generative process for STC model. The graphical representation of STC is given in Figure 3.

We notice that the contents of task matching queries and tweets are essentially represented by different kinds of languages. The study [8] shows that the words in task matching queries are formal, while the words in tweets are short, informal and abbreviated vocabularies. Particularly, tweets and task matching queries often have different words to describe the same topic. Therefore, we devise a dual language model for task matching queries and tweets based on different topic-word distributions $\beta^t$ and $\beta^v$, respectively.

Referring to Figure 3, the generative process for the words in questions and tweets, latent query factor $\mathbf{Q}$, and latent crowdworker factor $\mathbf{V}$, and the feedback score for task-solving activities $S$ in STC is described as follows.

1. For each task matching query $q$:
   (a) Draw topic proportions $\theta_q \sim Dir(\alpha)$
   (b) For each word $w_{q,i,j} \in w_q$
      i. Draw a topic assignment $z_{q,i,j} \sim Mult(\theta_q)$
      ii. Draw a word $w_{q,i,j} \sim Mult(\beta_{z_{q,i,j}})$
   (c) Draw latent query factor $\mathbf{q}^t \sim N(\theta_q, \lambda_Q^{-1})$

2. For each crowdworker $v$:
   (a) Draw topic proportions $\theta_v$ by Equation 3
   (b) For each word $w_{v,i,j} \in w_v$
      i. Draw a topic assignment $z_{v,i,j} \sim Mult(\theta_v)$
      ii. Draw a word $w_{v,i,j} \sim Mult(\beta_{z_{v,i,j}})$
   (c) Draw latent crowdworker factor $\mathbf{v}^t \sim N(\theta_v, \lambda_V^{-1})$

3. For each task matching $(q, v)$ in the processed task:
   (a) Draw feedback score $S_{(q,v)} \sim N(\mathbf{q}^T \mathbf{v}^t, \lambda_S^{-1})$

3.4 Inference Algorithm

In this section, we propose an inference algorithm that infers the latent query factor $\mathbf{Q}$, latent crowdworker factor $\mathbf{V}$, topic-word distributions $\beta^t$ and $\beta^v$ in STC.

Based on the generative process of STC, the joint distribution over feedback score $S$, topic proportions $\Theta$, latent query factor $\mathbf{Q}$ and latent crowdworker factor $\mathbf{V}$ can be factorized, which is given by

$$p(S, \mathbf{V}, \mathbf{Q}, \Theta, Z, W|\lambda_Q^{-1}, \lambda_V^{-1}, \lambda_S^{-1}, \lambda_B^{-1}, \alpha, \beta, A, B)$$

$$= p(\theta_Q|\alpha)p(\theta_V|\alpha, B)p(Z|\Theta)p(W|Z, \beta)p(\mathbf{V}|\theta_V, \lambda_V^{-1})$$

$$\times p(\mathbf{Q}|\theta_Q, \lambda_Q^{-1})p(S|\mathbf{Q}, \mathbf{V}, A, \lambda_S^{-1})$$

where

$$p(\theta_Q|\alpha) = \prod_{\theta_q} Dir(\alpha)$$

$$p(\theta_V|\alpha, B) = \prod_{\theta_v} Dir(\alpha)$$

$$\times \prod_{v_i, v_j \in V} N(\theta_{v_i} - \theta_{v_j}|0, B_{(v_i,v_j)})^{\lambda_B}$$

$$p(Z|\Theta) = \prod_{l \in \{v,q\}} \prod_{v_i,j \in Z} Mult(\theta_l)$$

$$p(W|Z, \beta) = \prod_{l \in \{v,q\}} \prod_{v_i,j \in W} \prod_{w_{i,j} \in w_l} \beta^{l,i,j}_{v,w_{i,j}}$$

$$p(\mathbf{Q}|\theta_Q, \lambda_Q^{-1}) = \prod_{q \in \mathbf{Q}} N(\mathbf{q}_q|^t|\theta_q, \lambda_Q^{-1})$$

$$p(\mathbf{V}|\theta_V, \lambda_V^{-1}) = \prod_{v \in \mathbf{V}} N(\mathbf{v}_v|^t|\theta_v, \lambda_V^{-1})$$

$$p(S|\mathbf{Q}, \mathbf{V}, A, \lambda_S^{-1}) = \prod_{A, (q_i, v_j) = 1} N(S_{(q_i,v_j)}|\mathbf{q}_q^T \mathbf{v}_v^j, \lambda_S^{-1})$$

We solve the inference problem, by finding a maximum a posterior (MAP) configuration of the latent query factor $\mathbf{Q}$ and latent crowdworker factor $\mathbf{V}$ conditioning on the feedback score $S$, and the words in queries and tweets $W$. That is, we aim to find

$$\left(\mathbf{Q}^*, \mathbf{V}^*\right) = \arg\max_{\mathbf{Q}, \mathbf{V}} p(\mathbf{Q}, \mathbf{V}|S, W, A, B, \lambda_S^{-1}, \lambda_Q^{-1}, \lambda_V^{-1}, \lambda_B^{-1}, \alpha, \beta)$$

(4)

Maximization a posterior configuration is equivalent to maximizing the complete log likelihood $\mathcal{L}$ and $\mathbf{V}$, given by

$$\mathcal{L} = -\frac{\lambda_V}{2} \sum_{v_i \in V} (\mathbf{v}_v^i - \theta_{v_i})^T (\mathbf{v}_v^i - \theta_{v_i})$$

$$-\frac{\lambda_Q}{2} \sum_{q_j \in Q} (\mathbf{q}_q^j - \theta_{q_j})^T (\mathbf{q}_q^j - \theta_{q_j})$$

$$+ \sum_{l \in \{v,q\}} \sum_{w_{i,j} \in w_l} \sum_{z_{i,j} \in z_{i,j}} \log(\sum_{k=1}^{K} \theta_{t,k} \beta_{z_{i,j}=k,w_{i,j}})$$

$$-\frac{\lambda_B}{2} \sum_{v_i,j \neq v_i} B_{(v_i,v_j)}(\theta_{v_i} - \theta_{v_j})^T (\theta_{v_i} - \theta_{v_j})$$

$$-\frac{\lambda_S}{2} \sum_{A, (q_i,v_j) = 1} (S_{(q_i,v_j)} - \mathbf{q}_q^T \mathbf{v}_v^j)^2$$

(5)

where we omit the constant terms and set the prior parameters $\alpha$ to a vector of 1.
Algorithm 1 Task Matching Query Processing Algorithm

Input: A task matching query \( q \), word distribution \( \beta^V \) and crowdworkers \( V \)

Output: A ranking of crowdworkers \( V' \)

1: Set \( \theta_q \propto \text{Uniform distribution} \\
2: for \( i : 1 \to \tau_{\text{max}} \) do \\
3: for each word \( w_{vq} \in w_q \) do \\
4: for \( k : 1 \to K \) do \\
5: Estimate variational parameter \( \phi_{vq,k} \propto \theta_{q,k}^2 \) \\
6: Sample \( \theta_q \propto \prod_{i=1}^{T} \sum_{k=1}^{K} \phi_{vq,k} \phi_{q,k} \) \\
7: Estimate the latent query factor \( q \sim N(\theta_q, \lambda_q^{-1}) \) \\
8: Rank crowdworkers based on the relevance \\
9: return A ranking of crowdworkers \( V' \)

Then, we infer the latent crowdworker factor \( \vec{V} \), the latent query factor \( \vec{Q} \), the latent topics of tweets, task matching queries \( \Phi \) and word distributions \( \beta^V \) and \( \beta^Q \). We estimate these parameters by taking the derivative of the complete log likelihood \( L \) in Equation 5 and set it to zero.

We first report the inference of latent query factor \( \vec{Q} \) and latent crowdworker factor \( \vec{V} \), which is given by

\[
\vec{V}_i \leftarrow (\lambda_S \vec{Q} \vec{Q}^T + \lambda_V I_K)^{-1} (\lambda_S \vec{Q} S_{vi} + \lambda_V \theta_{v_i}) \\
= \vec{Q} S_{vi} + \frac{\lambda_V}{\lambda_S} \theta_{v_i} \\
(6)
\]

\[
\vec{Q}_j \leftarrow (\lambda_S \vec{V} \vec{V}^T + \lambda_Q I_K)^{-1} (\lambda_S \vec{V} S_{vj} + \lambda_Q \theta_{q_j}) \\
= \vec{V} S_{vj} + \frac{\lambda_Q}{\lambda_S} \theta_{q_j} \\
(7)
\]

where \( S_{vi} \) is a diagonal matrix of the feedback scores for the workers on query \( q_i \) and \( S_{vj} \) is also a diagonal matrix of the feedback scores for all tasks done by worker \( v_j \).

We now justify the estimation of latent crowdworker factor and latent query factor as follows. The inference of latent crowdworker factor \( \vec{V} \) is based on both its topical interests from the tweets \( \theta_v \) and the latent factor of matched queries \( \vec{Q} S_{v} \) for worker \( v \). Similarly, the inference of latent query factor \( \vec{Q} \) is based on the latent topic from this query \( \theta_q \) and the latent factor of the crowdworkers worked for this query \( \vec{V} S_{q} \).

We then present the inference for the topical interests of crowdworkers \( \theta_v \), latent topics of the task matching queries \( \theta_q \), word distributions \( \beta^V \) and \( \beta^Q \), respectively. Unfortunately, we find that it is difficult to directly take the derivative of the complete log likelihood \( L \) with respect to \( \theta_V \) and \( \theta_Q \), due to the decoupling between word distributions \( \beta^V \), \( \beta^Q \) and topic assignment \( Z \). To tackle this problem, we introduce a new variational parameter \( \Phi \) for topic assignment \( Z \) and derive a lower bound, denoted by \( L' \). Then we estimate the parameters \( \theta_Q, \theta_V, \Phi, \beta^V \) and \( \beta^Q \) on \( L' \).

We derive the lower bound \( L' \) with respect to the topical interests of crowdworkers \( \theta_v \) by Jensen’s Inequality. The derivation for the lower bound \( L'(\theta_v) \) with respect to the latent topic of queries \( \theta_q \) is similar to the derivation of \( L' \). The derivation of \( L'(\theta_v) \) is given by

\[
L'(\theta_v) \geq -\frac{\lambda_V}{2} (\vec{v}_i - \theta_{v_i})^T (\vec{v}_i - \theta_{v_i}) \\
- \sum_{w_{v_j} \in w_{v_i}} \sum_{k=1}^{K} \phi_{v_j,k} \log \phi_{v_j,k} \\
+ \sum_{w_{v_j} \in w_{v_i}} \sum_{k=1}^{K} \phi_{v_j,k} \log (\theta_{v_k} \beta^V_{v_j,v_k} = k, w_{v_j}) \\
- \frac{\lambda_B}{2} \sum_{v_j \neq v_i} B(v_i,v_j) (\theta_{v_i} - \theta_{v_j})^T (\theta_{v_i} - \theta_{v_j}) = L'(\theta_v).
\]

We estimate the parameters \( \phi \) and \( \beta^V \) by taking the derivative of \( L' \) with respect to them. Therefore, we report the inference of the parameters \( \phi_{v_i,k} \) and \( \beta^V_{k,w} \) by

\[
\phi_{v_i,k} \propto \theta_{v_i,k} \beta^Q_{k,w} \\
(8)
\]

\[
\beta^Q_{k,w} \propto \sum_{w_{v_j} \in W} \sum_{w_{v_j} \in w_{v_j}} \phi_{v_j,k} 1[w_{v_j} = w] \\
(9)
\]

3.5 Task Matching Algorithm on STC

We propose a task matching query processing algorithm based on our STC model in Algorithm 1. That is, given a new task matching query \( q \), we aim to find the right crowdworkers to solve this task. Algorithm 1 first estimates the latent topic and topic assignments of query \( q \) alternatively. After that, Algorithm 1 samples the latent query factor \( \vec{Q} \) from its latent topic \( \theta_q \). Finally, we choose the crowdworkers based on the relevance between the latent crowdworker factor \( \vec{V} \) and the latent query factor \( \vec{Q} \), (i.e. \( \vec{V}^T \vec{Q} \)).

4. EXPERIMENTAL EVALUATION

The main goal of this experimental evaluation is to validate the effectiveness of our proposed STC model. To show its competitive performance, STC is compared with other four state-of-the-art approaches for task matching in crowdsourcing systems, such as Vector Space Model (VSM) [31], AuthorityRank [3], Dual Role Model (DRM) [31] and Topic Sensitive Probabilistic Model (TSPM) [39]. We implemented our algorithm in C++ and tested on machines with Linux OS Intel(R) Core(TM2) Quad CPU 2.66Hz, and 32GB RAM.

4.1 Experimental Setup

4.1.1 Datasets

We first collect the data from a popular crowdsourcing system, Quora. Quora is a question-and-answer website where questions are posted and answered by its community of crowdworkers. Quora was launched to the public in June, 2010 and has become very successful in just a few years. We then crawled the data of the social activities of all 95,915 users in September 2010 and have become very successful in just a few years.

We first collect the data posted between September 2012 and August 2013, and then crawled all the workers who answered these questions. In total, we collect 444,138 questions, 95,915 crowdworkers, 887,771 answers. We then collect the data of the social activities of all 95,915 users in
We denote by $R_{q}^{k}$ the set of crowdworkers ranked on $TopK$ by the algorithms. The measurement $Recall_{TopK}$ is given by

$$Recall_{TopK} = \frac{|\{q \in Q' | v \in R_{TopK}^{q}, q, v = 1\}|}{|Q'|}.$$  

**Cold-Start Rate.** We also investigate the types of the crowdworkers returned by an algorithm (i.e. cold-start crowdworkers or warm-start crowdworkers). In this experimental study, we consider the crowdworker who answered less than $\tau = 25$ questions as cold-start crowdworkers. We propose the measurement $Cold-Start Rate$ to illustrate the type of the crowdworkers ranked on top, which is given by

$$Cold-Start Rate = \frac{|\{q \in Q' | v \in R_{Top1}^{q}, \sum_{q \in Q'} A_{q, v} \leq \tau\}|}{|Q'|},$$

where $R_{Top1}^{q}$ is the set containing the crowdworker ranked the top.

### 4.2 Performance Results

We first compare the effectiveness of STC with the state-of-the-art methods on the six groups of testing questions. Next, we study the impact of various model parameters on STC model: dimension of latent space $K$, social regularization $\lambda_{B}$, and other model parameters $\lambda_{V}$ and $\lambda_{Q}$, respectively. Then, we illustrate the time cost of building our STC model.

#### 4.2.1 Performance Comparison

We compare STC with VSM, AuthorityRank, DRM and TSPM on six groups of queries and the results are presented in Figures 4(a) to 4(d).

We illustrate the Cold-Start Rate of the returned crowdworkers by all algorithms in Figure 4(a). The result shows that the rate of all the algorithms drops from $Q_{6}$ to $Q_{6}$. This is because the warm-start crowdworkers prefer solving common tasks to special tasks. We also find that the Cold-Start Rate of the crowdworkers found by STC is around 10% to 20% higher than other algorithms. The selection of the cold-start crowdworkers to solve the tasks is attributed to the effect of the knowledge transfer of STC.

Now, we show the benefits of the knowledge transfer of STC for cold-start crowdworkers using the measurements $Accu$, $Accu_{Top1}$ and $Recall_{TopK}$. We find that the performance of task matching can be greatly improved by STC.

Our STC has the excellent performance on both precision measurements: $Accu$ and $Accu_{Top1}$, for all the six groups of task matching queries. As shown in Figures 4(b) and 4(c), the performance of STC is 5% to 10% better than all the other algorithms. We notice that there is a number of cold-start crowdworkers having excellent performance on an extensive number of tasks in crowd sourcing systems. Using STC, the latent factor of cold-start crowdworkers can be inferred by knowledge transfer. Unlike the other classical algorithms, the high-quality cold-start crowdworkers can also be selected by STC to solve the tasks. Therefore, the precision of task matching is substantially improved.

The STC also greatly improves the performance of task matching in terms of $Recall_{TopK}$. Figure 4(d) shows that the recall of task matching is improved by 10% to 20% by STC. We find that the existing algorithms mostly select warm-start crowdworkers for solving the tasks. However, there is a significant number of the tasks that can be solved by

### Table 2: Summary of Datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Questions</th>
<th>Average #Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>444k</td>
<td>2</td>
</tr>
<tr>
<td>Q2</td>
<td>178k</td>
<td>3.5</td>
</tr>
<tr>
<td>Q3</td>
<td>86k</td>
<td>5.0</td>
</tr>
<tr>
<td>Q4</td>
<td>48k</td>
<td>6.7</td>
</tr>
<tr>
<td>Q5</td>
<td>30k</td>
<td>8.3</td>
</tr>
<tr>
<td>Q6</td>
<td>20k</td>
<td>10.0</td>
</tr>
</tbody>
</table>
cold-start crowdworkers. However, the existing algorithms do not select these cold-start crowdworkers who solve the tasks well. On the other hand, STC infers the latent factor of cold-start crowdworkers by knowledge transfer and can select both high-quality cold-start crowdworkers and warm-start crowdworkers to solve the tasks. Thus, the recall of the task matching is also improved.

4.2.2 Impact of Model Parameters

We investigate the impact of various model parameters including dimension of latent space $K$, social regularization $\lambda_B$, parameters $\lambda_V$ and $\lambda_Q$ for latent crowdworker factor and latent query factor, respectively.

**Dimension of latent space $K$.** We study the impact of dimension of latent space on the performance of STC by varying $K$ from 10 to 50, which is illustrated in Figures 5(a) to 5(d). Figure 5(a) shows the Cold-Start Rate of the returned crowdworkers by STC increases by 5% to 10% and then becomes convergent with respect to the dimension of latent space. By increasing the dimension of latent space, more knowledge is transferred from warm-start crowdworkers to cold-start crowdworkers. Figures 5(b) and 5(c) illustrate that the accuracy increases by 1% to 3% by varying the parameter $K$ and then converges. We conclude that the setting $K = 50$ is good enough to represent the latent factor of crowdworkers and queries. We notice that the recall increases significantly by 10% to 20% with respect to parameter $K$ in Figure 5(d). By transferring more knowledge to the cold-start crowdworkers, both cold-start crowdworkers and warm-start crowdworkers can be selected to solve the tasks such that the recall is greatly improved.

**Social regularization $\lambda_B$.** We investigate the impact of the social regularization $\lambda_B$ on the performance of STC, which is illustrated in Figures 6(a) to 6(d). We vary the value of the regularization term $\lambda_B$ from 0.01 to 1000. The Cold-Start Rate of the crowdworkers selected by STC first increases and then becomes convergent in Figure 6(a). The larger the parameter $\lambda_B$ becomes, the more knowledge is transferred to the cold-start crowdworkers. Thus, more cold-start crowdworkers are selected for solving the tasks. The performance of STC is first improved but then declines with respect to $\lambda_B$ on Accu, Accu$_{Top1}$, and Recall$_{TopK}$ in Figures 6(b) to 6(d). The inference of latent crowdworker factor by STC is based on both past task-solving activities in crowdsourcing systems and the activities of the crowdworkers in social networks. When the social regularization term $\lambda_B$ is small, the knowledge transfer improves the performance of inferring the latent crowdworker factor. When the value of social regularization term $\lambda_B$ becomes large, the inference of latent crowdworker factor is dominated by the activities of the crowdworkers in social networks. Thus, the performance of STC declines. To balance the inference from both past task-solving activities and social activities, we set the value of $\lambda_B$ as a new regularization term.

**Impact of Parameters $\lambda_V$ and $\lambda_Q$.** We study the impact of parameters $\lambda_V$ and $\lambda_Q$ on the performance of STC by varying $\lambda_V$ and $\lambda_Q$ from 0.01 to 1000 and the results are presented in Figures 7(a) to 7(d). The Cold-Start Rate does not vary for the parameters $\lambda_V$ and $\lambda_Q$ and hence the result is omitted. We observe that the performance of STC declines with respect to the large value of both parameters in Figures 7(a) to 7(d), which is similar to the effect of pa-
parameter $\lambda_D$. Therefore, we also set the value of $\lambda_V$ and $\lambda_Q$ as new regularization terms.

We remark that the overall performance of STC with three new regularization terms can also be improved by 5%, 5% and 10% on $\text{Accu}$, $\text{Accu}_{\text{Top1}}$ and $\text{Recall}_{\text{Top100}}$, respectively.

4.2.3 Running Time

We first study the running time of building our STC model on the convergence of model inference. We then investigate the time cost of model inference with respect to different dimensions of latent space $K$. We set the convergence threshold $\epsilon$ to $10^{-5}$ for inferring STC model. Figure 8(a) shows that the inference of STC model with latent space $K = 10$ converges after 20th iteration. Figure 8(b) illustrates that the time cost of inferring STC model scales linearly with respect to the dimensions of latent space $K$. Thus, it is efficient to build STC model on real crowdsourcing platforms such as Quora. After inferring the STC model, the time cost for matching the right crowdworkers can be computed in real time.

5. RELATED WORK

In this section, we briefly review some related work on crowdsourcing, transfer learning in the literature.

Crowdsourcing. Crowdsourcing has been widely used to solve challenging problems by human intelligence in comprehensive areas. Some successful applications that appear include CrowdDB [5] and CDAS [16].

Recently, the crowdsourcing techniques have been applied in several research areas such as database management, machine learning and information retrieval. The crowdsourcing techniques on entity resolution were studied in [29, 30]. CrowdScreen [22] applied the crowdsourcing techniques in decision making. Guo et al. [7] studied the problem of finding maximum element in the crowdsourcing databases. In [4], Davidson et al proposed the top-k and group-by queries on crowdsourcing databases. Kaplan et al [9] aimed to select the right question for planing queries. Marcus et al [17] studied the count query with the crowd. Gao et al [6] proposed crowdsourcing based online algorithm to find the ground truth. Zhao et al [35, 36] proposed crowd-selection to find the right workers to answer the questions. The work [12, 33, 20] proposed the assignment algorithms for tagging based on crowdsourcing platform.

Transfer Learning. Transfer learning techniques tackle the problem of data sparsity in the target domain by transferring supervised knowledge from other related domains, which achieve great success in a lot of applications such as classification, clustering.

Recently, Zhong et al. developed techniques [37, 38] that borrow the knowledge from the auxiliary historical activities of users in multiple networks to predict the user behavior in a target network. Mo et al. proposed cross-task crowdsourcing method [19] that transfers the knowledge cross different tasks. However, their approaches are difficult to be applied to task matching for cold-start crowdsourcing since the total number of auxiliary worker activities is limited. In this work, we enrich the profile of cold-start crowdworkers by external knowledge transfer.

6. CONCLUSIONS

We studied the problem of task matching for cold-start crowdsourcing by transferring knowledge from social networks to crowdsourcing systems. We developed a Social-Transfer graph to capture the relations between tasks, crowdworkers and social networks. We then proposed a Social-Transfer model for cold-start Crowdourcing called STC. The model not only infers the latent factor of warm-start crowdworkers from their past task-solving activities, but also transfers the knowledge from warm-start crowdworkers to cold-start crowdworkers via social networks. We evaluated the performance of STC on the popular crowdsourcing system Quora for question-answering, by transferring knowledge from Twitter. Our results show that compared with four state-of-the-art methods for task matching in crowdsourcing systems, STC selects more cold-start crowdworkers who are the best crowdworkers to solve the given tasks, and STC improves both the precision and recall of task matching. The results confirm that our approach effectively finds the high-quality cold-start as well as warm-start crowdworkers for crowdsourcing tasks.

Acknowledgments. We thank the reviewers for giving us many constructive comments, with which we have significantly improved our paper. This research is supported in part by SHIAE Grant No. 8115048, MSRA Grant No. 6903555, and HKUST Grant No. FSGRF13EG22.

7. REFERENCES
