A Transfer Learning based Framework of Crowd-Selection on Twitter

Zhou Zhao, Da Yan, Wilfred Ng and Shi Gao
Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
Hong Kong, China
{zhaozhou,yanda,wilfred,gaoshi}@cse.ust.hk

ABSTRACT

Crowd selection is essential to crowd sourcing applications, since choosing the right workers with particular expertise to carry out crowdsourced tasks is extremely important. The central problem is simple but tricky: given a crowdsourced task, who are the most knowledgable users to ask? In this demo, we show our framework that tackles the problem of crowdsourced task assignment on Twitter according to the social activities of its users. Since user profiles on Twitter do not reveal user interests and skills, we transfer the knowledge from categorized *Yahoo ! Answers* datasets for learning user expertise. Then, we select the right crowd for certain tasks based on user expertise. We study the effectiveness of our system using extensive user evaluation. We further engage the attendees to participate a game called "Whom to Ask on Twitter". This helps understand our ideas in an interactive manner. Our crowd selection can be accessed by the following url `http://webproject2.cse.ust.hk:8034/tcrowd/`.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications

General Terms

Algorithms, Experiments

Keywords

Crowdsourcing, Microblogs

1. INTRODUCTION

In recent years, the studies of crowdsourcing techniques [6] have attracted a lot of attention due to their effectiveness in real-life applications, such as image tagging and natural language processing. Earlier approaches usually randomly select workers for certain tasks on well designed platforms such as *Amazon MTurk*[1] and *Crowdflower*[2] [7, 8]. More recently, a new trend that utilizes social networks as crowd platforms and asks questions to their users [2, 5, 3, 9] emerges. The crowd selection procedure in these works is based on the trustiness of the users. Therefore, a set of reliable users are selected for addressing certain tasks.

However, in many cases, selecting workers based on their skills for certain tasks is a better solution. Consider a crowdsourced query, "What is the complexity of sorting the elements in an array?". Crowd selection based only on the trustiness of the workers does not always work in this case. Since the selected workers may not have expertise in the area of computer science. Therefore, they may not be able to answer this query.

In this demo, we propose a web-based system for crowd selection on Twitter. Basically, we focus on addressing the following challenging issues:

- **Limited Expertise Information.** The user expertise information is very limited on Twitter, where most of its users do not explicitly state their interests and skills. Since the users may tweet or re-tweet the messages (or commonly referred to as tweets), we have to infer the user expertise based on the tweets.
- **Large Volume of Tweets.** The tweet messages are in large quantity and arriving in high speed. Therefore, it is infeasible for humans to label the tweets.
- **Requiring Online Crowd Selection.** Many crowdsourcing applications need online crowd selection. For example, a mobile crowd search system requires to return the results in real time.

Many existing works have studied the problem of crowd selection on microblogs [2, 5, 3, 9]. However, they select the workers for crowdsourced tasks based on only trustiness. In contrast, our crowd selection is based on worker expertise. In this work, we train a Naïve Bayes model based on categorized *Yahoo!Answer* Q ! A datasets. Next, we transfer the trained model to build a new bayesian model for user expertise inference on Twitter. Then, we rank the workers based on their expertise level on the crowdsourced task and recommend them to user.

The rest of this demo is organized as follows. Section 2 introduces the framework of our system. Section 3 then explains the crowd selection procedure in our system. Section

[1] `https://www.mturk.com/mturk/welcome`
[2] `http://crowdflower.com/`

KDD'13, August 11–14, 2013, Chicago, Illinois, USA.

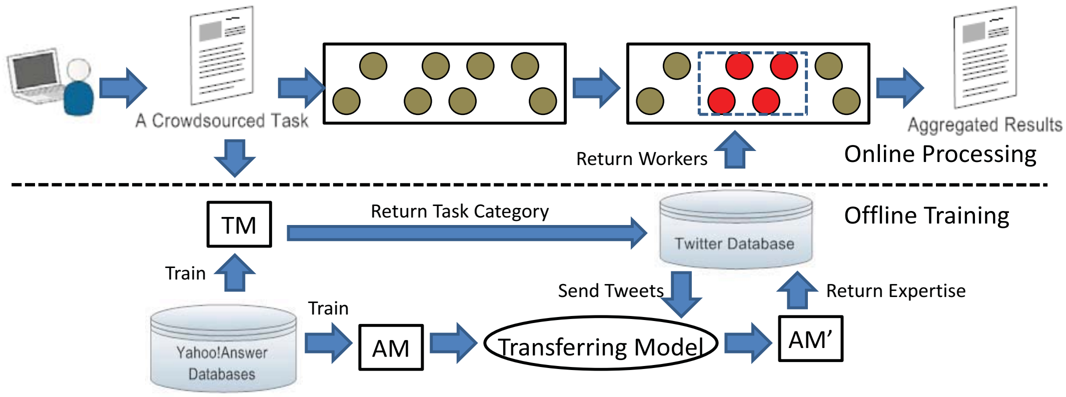

Figure 1: System Overview of Our Twitter-Based Framework

4 discusses the demonstration plan and we conclude the paper in Section 5.

2. SYSTEM DESIGN

We first present a general process of our system. Then, we introduce the transfer learning based algorithm.

2.1 General Process

The general process of our system is illustrated in Figure 1, which can be divided into two phases: training bayesian models and processing crowdsourced tasks.

A bayesian model is trained offline. First, we train a Naïve Bayes model based on categorized tasks from *Yahoo!Answer* website [1], denoted by *TM*. Then, we train another Naïve Bayes model based on the categorized answers from *Yahoo!Answer* website [1], denoted by *AM*. We utilize *TM* to categorize the crowdsourced tasks. However, we find that the trained *AM* model cannot be directly applied for user expertise inference, since the domain of tasks in *Yahoo!Answer* is different from the domain of tweets in Twitter. Therefore, we transfer the trained *AM* model to build a new bayesian model *AM'* for user expertise inference. After obtaining the user expertise, we store the data into the databases. We will have more technical details of building the new model in the next section.

The crowdsourced tasks are processed online. Consider a crowdsourced task processing in our system. First, the user u inputs a crowdsourced task t. The *TM* model categorizes the input task. We consider all the "followings" and "followers" of user u as the candidate crowd for the task. The system queries the expertise level of the candidate crowd from the databases. Then, the system ranks these workers in the candidate crowd based on the expertise level and recommends it to the user u. Finally, the user u tweets the task to the recommended workers. The system keeps collecting the answers from twitter workers.

2.2 Transfer Learning

We transfer the trained *AM* model to a new *AM'* model based on the technique in [4]. Our algorithm first estimates the initial parameters under the categorized answers D_c from *Yahoo!Answer*, and then uses an EM algorithm to revise the model *AM* for the tweets of users D_u, which are uncategorized.

In our Naïve Bayes model *AM*, an answer $a \in D_c$ is associated with a data instance. The answer a can be represented as a bag of words, where each word w comes from a word corpus W. Each answer a corresponds to a category c. We denote the model parameter of *AM* by $\theta_{D_c} = \{p_{D_c}(c), p_{D_c}(w|c)\}$.

The Naïve Bayes model *AM* estimates the conditional probability $p(c|a)$, given by

$$\begin{aligned} p_{D_c}(c|a) &\propto p_{D_c}(c) \cdot p_{D_c}(a|c) \\ &= p_{D_c}(c) \prod_{w \in a} p_{D_c}(w|c). \end{aligned} \tag{1}$$

We estimate $p_{D_c}(w|c)$ by Laplacian smoothing, given by

$$p_{D_c}(w|c) = \frac{1 + n_{D_c}(w,c)}{|W| + n_{D_c}(c)}, \tag{2}$$

where $n_{D_c}(w,c)$ is the number of times of the word w in the category c and $n_{D_c}(c)$ is the number of words whose category is c in the categorized data D_c.

Now, we build a new model *AM'*, which is based on uncategorized tweets D_u. We denote the model parameter of *AM'* by $\theta_{D_u} = \{p_{D_u}(c), p_{D_u}(w|c)\}$. We consider each tweet in D_u by d. We utilize an EM algorithm which maximizes the loglikehood $\mathcal{L}(\theta_{D_u}|D_c, D_u)$ by iterating through the following two steps:

- **E-Step:** We estimate the posterior probability of the category of the tweets in D_u, given by

$$p_{D_u}(c|d) \propto p_{D_u}(c) \prod_{w \in d} p_{D_u}(w|c). \tag{3}$$

- **M-Step:** We estimate the parameter of the model *AM'*, given by

$$p_{D_u}(c) \propto \sum_{i \in \{c,u\}} p_{D_u}(D_i) \cdot p_{D_u}(c|D_i) \tag{4}$$

$$p_{D_u}(w|c) \propto \sum_{i \in \{c,u\}} p_{D_u}(D_i) \cdot p_{D_u}(c|D_i) \cdot p_{D_u}(w|c, D_i). \tag{5}$$

We estimate $p_{D_u}(w|c, D_i)$ by

$$p_{D_u}(w|c, D_i) = \frac{1 + n_{D_u}(w,c,D_i)}{|W| + n_{D_u}(c, D_i)} \tag{6}$$

Figure 2: An Example of Estimated User Expertise

where

$$n_{D_u}(w, c, D_i) = \sum_{d \in D_i} |d| \cdot p_{D_u}(w|d) \cdot p_{D_u}(c|d), \quad (7)$$

$$n_{D_u}(c, D_i) = \sum_{d \in D_i} |d| \cdot p_{D_u}(c|d). \quad (8)$$

where $|d|$ is the number of words in the tweet d.

2.3 Estimate Expertise Level

The tweets are categorized after the algorithm converges in log-likehood. An example of the categorized tweets for the users in our databases is illustrated in Figure 2. The level of user expertise on the task category c equals to the number of produced tweets which is associated with category c, given by

$$l_c(u) = \sum_{d_u} I(f(d_u) = c). \quad (9)$$

where d_u is the tweet produced by user u and $f(d_u)$ is the estimated category by using model *AM'*. $I(f(d_u) = c)$ is an indicate function. When the user inputs the crowdsourced task with category c, the system ranks the workers in the candidate crowd based on their level of expertise level on that category.

3. CROWDSOURCED TASK PROCESSING

We now explain the crowdsourced task processing of our system. The demonstration video of the system can be found in `http://www.youtube.com/watch?v=PeMaw-gifpU`.

To enable the crowdsourced task processing, the system asks for the authority of users' account so that it is able to

- read all the tweets in the timeline.
- read all the followings and followers.
- tweet the messages from the account.

On the other hand, to protect the privacy of the account, our system does not

- read the private messages.
- read the password of the account.

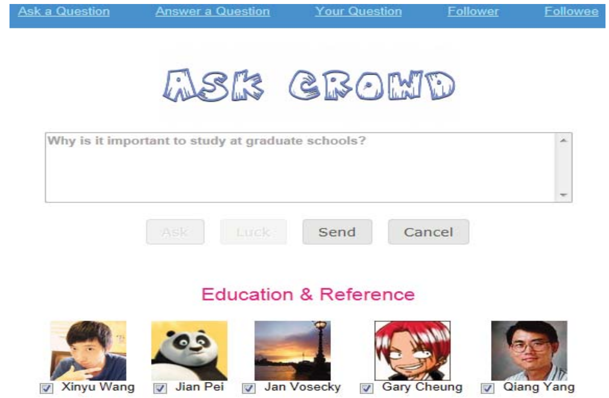

Figure 3: Ask the Crowd

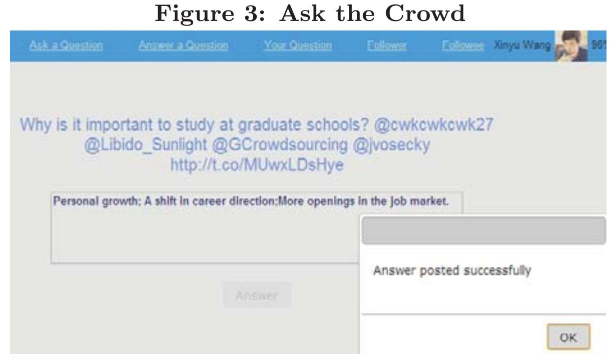

Figure 4: Answer the Task

When a new user comes, the system first reads all the tweets from his/her account and store them in the databases. Next, the system also stores the tweets from his/her followings and followers who are not in our databases. Then, the system builds the *AM'* model to infer the expertise level of the new user, his/her followings and followers, which are also stored in the databases.

Now, consider a crowdsourced task "Why is it important to study at graduate schools?"

Ask the Crowd. When the system receives this task, it first sends to the task model *TM* and the model *TM* returns the category of this task. We can see that this task is classified in the "Education& Reference" category in Figure 3. Next, the system issues the query for the followers and followings of the user on Twitter from the databases. We consider the queried followers and followings as the candidate crowd. Then, the databases return the workers in the candidate crowd ranked by the level of expertise on the category of "Education& Reference". Figure 3 shows the top five workers ranked category of "Education& Reference" from the left to the right. We present the selected workers by both name and head portrait such that the user can recognize them well.

The user can choose the selected workers and click the "send" button. After that, the system generates the task sheet and Twitter the sheet to the workers by @ them.

Answer the Task. After the task being posted, the workers can receive the generated sheet immediately. The workers are informed by the notification of Twitter, since the system sends the task sheet by "@" them. The workers can fill up the task sheet and click the "send"button. Then,

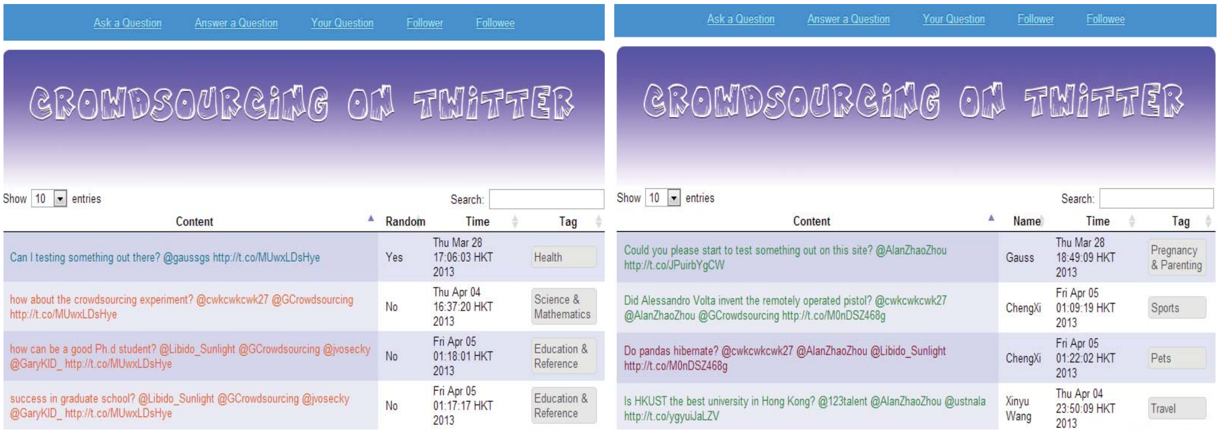

Figure 5: Task and Answer Records

the system generates the answer sheet, which is sent to our databases. Finally, our databases aggregate the collected sheets under the posted task.

The system also maintains the historical record of all the tasks and received answers. The users are able to search the posted tasks and its related answers from the system, illustrated in Figure 5.

4. DEMONSTRATION PLAN

In this demonstration, we plan to engage the attendees to participate in Crowd Selection by starting an interesting game called "Whom to Ask on Twitter". We utilize the *Yahoo!Answer* datasets as the initial data source.

We build two Naïve Bayes models *TM* and *AM* based on the categorized *Yahoo!Answer* dataset. We also encourage the audience to login into our system with their Twitter account. Then, the system transfers the *AM* model to the *AM'* model for user expertise inference. In the meant time, we explain how the system works. Then, we invite the audience to issue the crowdsourced task (i.e. technical questions raised during the conference presentation) to our system. The system shows the right crowd for asking these questions.

The system aims to match a specific task with the workers who have expertise to address it. Currently, we set the candidate crowd of a Twitter user to be his/her followers and followings. The reason is that the users far away have much less incentive to address the crowdsourced task. Later, we will extend the candidate crowd to be the users on Twitter by incorporating an effective incentive mechanism.

5. CONCLUSION

In this demo, we explore a new issue of "Whom to Ask on Twitter". Different from the general crowdsourced task processing, we focus on a specific task with needed expertise. We build a framework of crowd selection on Twitter. The system returns the workers ranked by their expertise level on the crowdsourced task category. Based on the algorithm, we devise a demonstration program on *Yahoo!Answer* data and engage the attendees to participate our system. We devise an interactive game "Whom to Ask on Twitter " to demonstrate the effectiveness of our system.

ACKNOWLEDGEMENTS: We thank the help from the UROP project students Xinyu Wang, Weikeng Chen and Zhijun Zhang at HKUST for helping the experiments. This work is partially supported by RGC GRF under grant number HKUST 617610 and 618509.

6. REFERENCES

[1] Yahoo!answer datasets. `http://answers.yahoo.com/`.

[2] A. Bozzon, M. Brambilla, and S. Ceri. Answering search queries with crowdsearcher. In *Proceedings of the 21st international conference on World Wide Web*, pages 1009–1018. ACM, 2012.

[3] C. C. Cao, J. She, Y. Tong, and L. Chen. Whom to ask?: jury selection for decision making tasks on micro-blog services. *Proceedings of the VLDB Endowment*, 5(11):1495–1506, 2012.

[4] W. Dai, G.-R. Xue, Q. Yang, and Y. Yu. Transferring naive bayes classifiers for text classification. In *Proceeding of The National Conference on Artifical Intelligence*, volume 22, page 540. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.

[5] X. Liu, M. Lu, B. C. Ooi, Y. Shen, S. Wu, and M. Zhang. Cdas: a crowdsourcing data analytics system. *Proceedings of the VLDB Endowment*, 5(10):1040–1051, 2012.

[6] V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy. Learning from crowds. *The Journal of Machine Learning Research*, 99:1297–1322, 2010.

[7] V. S. Sheng, F. Provost, and P. G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 614–622. ACM, 2008.

[8] Y. Tian and J. Zhu. Learning from crowds in the presence of schools of thought. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 226–234. ACM, 2012.

[9] Z. Zhao, W. Ng, and Z. Zhang. Crowdseed: query processing on microblogs. In *Proceedings of the 16th International Conference on Extending Database Technology*, pages 729–732. ACM, 2013.