Automatically Grouping Questions in Yahoo! Answers

Yajie Miao*, Lili Zhao*, Chunping Li*, Jie Tang**
Tsinghua National Laboratory for Information Science and Technology (TNList)
**School of Software, Tsinghua University*
***Department of Computer Science and Technology, Tsinghua University*
yajiemiao@gmail.com, zhaoll07@mails.tsinghua.edu.cn, {cli, jietang}@tsinghua.edu.cn

Abstract

In this paper, we define and study a novel problem which is referred to as Community Question Grouping (CQG). Online QA services such as Yahoo! Answers contain large archives of community questions which are posted by users. Community Question Grouping is primarily concerned with grouping a collection of community questions into predefined categories. We first investigate the effectiveness of two basic methods, i.e., K-means and PLSA, in solving this problem. Then, both methods are extended in different ways to include user information. The experimental results with real datasets show that incorporation of user information improves the basic methods significantly. In addition, performance comparison reveals that PLSA with regularization is the most effective solution to the CQG problem.

1. Introduction

With the blooming of Web 2.0, user-generated contents (UGC) such as Wikipedia, YouTube and Flickr begin to flourish. One type of UGC sites are the Community Question Answering (CQA) services, which enable users to post or answer questions on various subjects. Yahoo! Answers is now becoming the most popular CQA portal. Since its launch in 2005, Yahoo! Answers has attracted millions of users, and has stored a tremendous number of community questions in its database.

In Yahoo! Answers, this archive of community questions are organized in the form of hierarchical categories. However, the maintenance of this category structure highly relies on the efforts of users. When submitting new questions, users are required to assign category tags to their questions, though Yahoo! Answers provides category recommendations. Figure 1 gives an example of category selection in Yahoo! Answers.

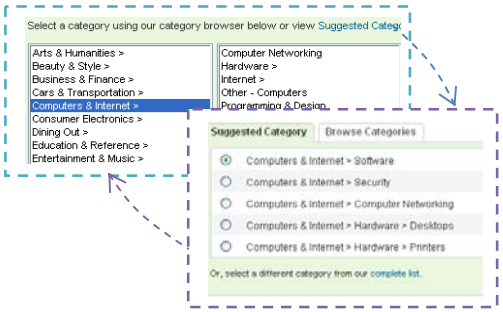


Figure 1. **Category selection in Yahoo! Answers.**

Recently, there has been a growing amount of research on CQA, and these research pertains to various aspects such as *User Satisfaction Prediction* [14], *Community Question Retrieval* [1, 4], *Question-Answer Relationship* [2], etc. However, there are yet no mechanisms with which we can group community questions automatically. In this paper, we study a novel problem which is referred to as Community Question Grouping (CQG). Given a set of community questions, the task of CQG is to automatically group these questions into predefined categories. This general problem subsumes interesting applications in that Yahoo! Answers users do not have to label their questions manually. Also, Community Question Grouping can potentially identify some new or emerging categories, and thus the existing category structure can be enriched and refined continuously.

As for solutions to CQG, we first examine whether the existing methods can solve this problem effectively. Particularly, we adapt two commonly accepted methods, i.e., K-means and PLSA, to the CQG problem. For both methods, the textual contents of community questions are utilized as grouping features. Then, we take user information in Yahoo! Answers into account. Users tend to post or answer questions on areas that they are most interested in or familiar with. As a result, the users *involved* in a community question

provide valuable indication about which category it belongs to. Based on this, we extend K-means and PLSA respectively, and propose three new methods, i.e., User K-means, User PLSA and Regularized PLSA. In User K-means, user information is exploited to enrich the textual representation of community questions. Differently, User PLSA combines text and user information in a unified probabilistic framework, and Regularized PLSA smoothes the estimation process of PLSA with *question graphs* which are derived from user information.

To the best of our knowledge, this study is the first attempt to address the problem of Community Question Grouping. We conduct extensive experimental studies to evaluate the proposed methods on three datasets: *Hardware*, *Internet* and *Science*. The experimental results show that the incorporation of user information improves the basic K-means and PLSA methods significantly. Also, we observe that Regularized PLSA performs most effectively, while at the same time achieving desirable efficiency.

The rest of the paper is organized as follows. In Section 2, we formally define the problem. After that, we present the methods in Section 3. Then, we show and discuss the experimental results in Section 4. In Section 5, we review some previous work which is related with this study. Finally, we have the conclusion and future work in Section 6.

2. Problem definition

In this section, we give the formal definition of the CQG problem. In Yahoo! Answers, one community question usually consists of three parts, i.e., the *subject* (a brief statement of the question), the *content* (additional detailed descriptions of the question) and the answers posted by other users (see Figure 2).

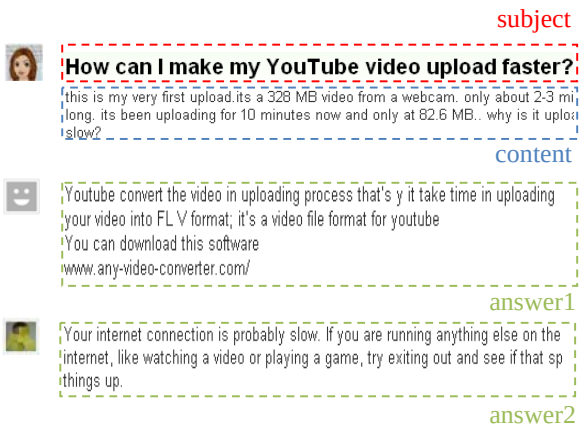


Figure 2. **A community question in Yahoo! Answers.**

We first consider the textual information of community questions. In Yahoo! Answers, the life cycle of a question is initiated when a user submits the subject (and content) to the community. However, we argue that the answers posted by others comprise an integral part of the entire *question thread* in the sense that they provide lexical or semantic extensions for the original question. We define the *associated text* of a community question as follows.

Definition (**Associated Text**): The associated text of a community question is the concatenation of its subject, content and answers.

For a specific community question, some users get involved in it either by posting the question or by providing answers to it. Figure 3 shows the number of users who appear in 1,2,3,4,5,6 categories respectively, in the three experimental datasets. An observation is that users tend to be concerned with only one or two areas due to their personal professions and interests. Because of this *concentration*, user involvement can serve as additional features which indicate the category that a question belongs to. We define the *involved users* of a community question as follows.

Definition (**Involved Users**): The involved users of a community question include both the user who posts the question and the users who issue answers.

Formally, we denote a collection of community questions with $Q=\{q_1,q_2,\cdots\}$. When considering the associated text, each question $q\in Q$ can be described as a word vector as follows.

$$q=\{c(w_1,q),c(w_2,q),\cdots,c(w_M,q)\}, \qquad (1)$$

where $c(w_h,q)$ is the number of occurrences of word w_h in the associated text of question q, and M is the total number of words in the collection Q.

Similarly, the question q is represented in the form of a user vector, that is,

$$q=\{f(u_1,q),f(u_2,q),\cdots,f(u_N,q)\}, \qquad (2)$$

where $f(u_t,q)$ is the number of times that user u_t gets involved in q (one user may answer the same question more than once), and N is the total number of users involved in the collection Q.

Then the goal of CQG is to partition the whole collection Q into disjunct groups, i.e.,

$$Q=\{G_1,G_2,\cdots,G_k\}, \qquad (3)$$

where k is the number of groups, and each group G_i corresponds to a specific category in Yahoo! Answers.

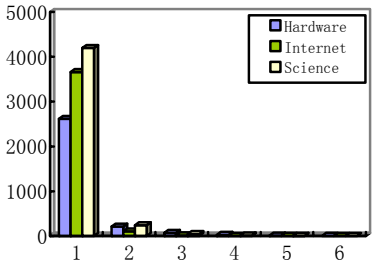


Figure 3. **Relations between user numbers and category numbers.**

3. Grouping methods

In this section, we formulate Community Question Grouping as a clustering task. We first use basic clustering methods, i.e., K-means and PLSA, for solving this problem. Then, we extend both methods to include user information in different manners. The details of these methods are described as follows.

3.1. K-means

3.1.1. Basic K-means. In Section 2, each community question has been represented with a word vector. Then we apply K-means [8], a standard clustering algorithm proved to be effective in many clustering tasks, to a set of such vectors. The distance between data points and centroids is computed as the *cosine similarity* of word vectors.

3.1.2. User K-means. Besides textual contents, another important form of information embedded in community questions is their involved users. These users can act as additional features to enrich the textual representation of the questions. Therefore, the community question q is reformulated as a *word-user vector*.

$$q = \{c(w_1,q), c(w_2,q), \cdots, c(w_M,q), \\ \alpha \cdot f(u_1,q), \alpha \cdot f(u_2,q), \cdots, \alpha \cdot f(u_N,q)\}, \quad (4)$$

where α is a parameter to indicate the importance of user features in the overall representation. Accordingly, we apply K-means to these expanded vectors for question grouping.

3.2. PLSA

3.2.1. Basic PLSA. Probabilistic latent semantic analysis (PLSA) [9] has been applied to clustering with promising results [6, 7]. For the CQG problem, our idea is to use a unigram language model (a multinomial word distribution) to model a group (topic). To be consistent with previous literature, we still define the k unigram language models as $\Theta = \{\theta_1, \theta_2, \cdots, \theta_k\}$ which capture individual groups. Then each word w_h in question q is generated from a two-stage process: first, a group θ_j is chosen conditionally for the question according to $\pi_{q,j}$; second, the word w_h is generated from θ_j according to the conditional probability $p(w_h \mid \theta_j)$. From a statistical perspective, the question collection Q is the observed data, and its log-likelihood is described as

$$L(Q) = \sum_{q \in Q} \sum_{w_h \in V} \{c(w_h,q) \times \log \sum_{j=1}^{k} [\pi_{q,j} \cdot p(w_h \mid \theta_j)]\}, \quad (5)$$

where V is the whole set of words in Q.

We perform Maximum Likelihood Estimation using the EM algorithm. The latent variable z_{q,w_h} is defined as the group from which the word w_h in question q is generated. During the estimation process, the model parameters are updated iteratively as follows[1].

E-step: $$p(z_{q,w_h} = j) = \frac{\pi_{q,j}^{(n)} p^{(n)}(w_h \mid \theta_j)}{\sum_{j'=1}^{k} \pi_{q,j'}^{(n)} p^{(n)}(w_h \mid \theta_{j'})}$$

M-step: $$\pi_{q,j}^{(n+1)} = \frac{\sum_{w_h \in V} c(w_h,q)\, p(z_{q,w_h} = j)}{\sum_{j'}^{k} \sum_{w_h \in V} c(w_h,q)\, p(z_{q,w_h} = j')}$$

$$p^{(n+1)}(w_h \mid \theta_j) = \frac{\sum_{q \in Q} c(w_h,q)\, p(z_{q,w_h} = j)}{\sum_{w'_h \in V} \sum_{q \in Q} c(w'_h,q)\, p(z_{q,w'_h} = j)} \quad (6)$$

where $p(z_{q,w_h} = j)$ represents the probability that the word w_h in question q is generated from the j^{th} group.

3.2.2. User PLSA. In order to take advantage of involved users, we propose a User PLSA method which combines text and user features in a unified probabilistic framework. In addition to the word distribution, each group is also represented by a multinomial distribution of users. Formally, the k user distributions are defined as $\Psi = \{\psi_1, \psi_2, \cdots, \psi_k\}$. Then, besides the word generation process in PLSA, there exists a user generation process. These two processes are *linearly* combined together, and the log-likelihood of the question collection is reformulated as follows.

$$L(Q \mid \Lambda) = \sum_{q \in Q} \Bigg\{ \beta \cdot \sum_{w_h \in V} [c(w_h,q) \times \log \sum_{j=1}^{k} (\pi_{q,j} \cdot p(w_h \mid \theta_j))] \\ + (1-\beta) \cdot \sum_{u_t \in S} [f(u_t,q) \times \log \sum_{j=1}^{k} (\pi_{q,j} \cdot p(u_t \mid \psi_j))] \Bigg\} \quad (7)$$

[1] The inference of Basic PLSA and User PLSA can be found in est.pdf at https://sourceforge.net/projects/yahoodataset/

where S is the set of users appearing in the question collection, $p(u_t \mid \psi_j)$ is the probability of choosing user u_t when given the j^{th} group, and β is a combination factor to trade off the text and user features. The other parameters have the same meanings as in Equation (5). Besides z_{q,w_h}, we introduce another latent variable z_{q,u_t} which denotes the group from which the user u_t in question q is generated. Similarly, the EM algorithm is used for estimation of the parameter set Λ. The updating formulas for the iterative estimation process are as follows.

E-step:

$$p(z_{q,w_h} = j) = \frac{\pi_{q,j}^{(n)} p^{(n)}(w_h \mid \theta_j)}{\sum_{j'=1}^{k} \pi_{q,j'}^{(n)} p^{(n)}(w_h \mid \theta_{j'})}$$

$$p(z_{q,u_t} = j) = \frac{\pi_{q,j}^{(n)} p^{(n)}(u_t \mid \psi_j)}{\sum_{j'=1}^{k} \pi_{q,j'}^{(n)} p^{(n)}(u_t \mid \psi_{j'})}$$

M-step:

$$\pi_{q,j}^{(n+1)} = \frac{\beta \sum_{w_h \in V} c(w_h, q) p(z_{q,w_h} = j) + (1-\beta) \sum_{u_t \in S} f(u_t, q) p(z_{q,u_t} = j)}{\sum_{j'=1}^{k} [\beta \sum_{w_h \in V} c(w_h, q) p(z_{q,w_h} = j') + (1-\beta) \sum_{u_t \in S} f(u_t, q) p(z_{q,u_t} = j')]}$$

$$p^{(n+1)}(w_h \mid \theta_j) = \frac{\sum_{q \in Q} c(w_h, q) p(z_{q,w_h} = j)}{\sum_{w'_h \in V} \sum_{q \in Q} c(w'_h, q) p(z_{q,w'_h} = j)}$$

$$p^{(n+1)}(u_t \mid \psi_j) = \frac{\sum_{q \in Q} f(u_t, q) p(z_{q,u_t} = j)}{\sum_{u'_t \in S} \sum_{q \in Q} f(u'_t, q) p(z_{q,u'_t} = j)} \quad (8)$$

where $p(z_{q,u_t} = j)$ represents the probability that the user u_t in question q is generated from the j^{th} group.

3.2.3. Regularized PLSA.

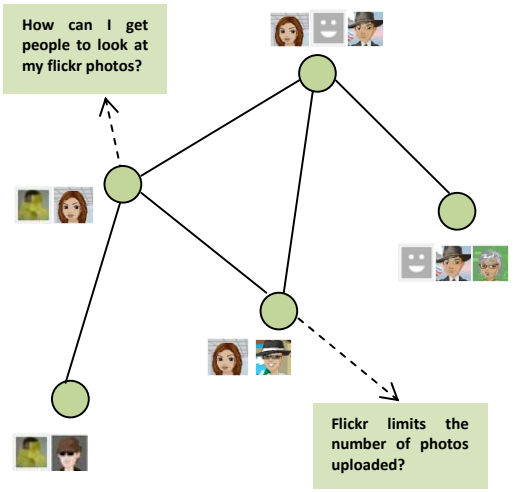


Figure 4. **Question graph.**

Because of the consideration of involved users, a graph structure actually exists in the question collection. In such a graph, each node represents a community question. There is an edge between two nodes if they have common involved users. This graph is defined as a *question graph* in this study, and an example is presented in Figure 4.

Intuitively, questions posted or answered by the same user are much likely to belong to the same category. This means that the nodes which are connected to each other on the question graph should have similar group distributions ($\pi_{q,j}$). Based on this, we propose another method, named Regularized PLSA, to improve PLSA with user information. In this method, the question graph is exploited to regularize the process of group modeling and parameter estimation. We let G denote the question graph and E denote the set of edges. The weight of each edge, i.e., $w(u,v)$, is the number of common involved users shared by the two corresponding questions. When using the *harmonic function*, the regularizer on the question graph can be formulated as

$$R(Q,G) = \frac{1}{2} \sum_{(u,v) \in E} w(u,v) \sum_{i=1}^{k} (\pi_{u,i} - \pi_{v,i})^2 . \quad (9)$$

By combining this regularizer with the basic PLSA model, we can give the overall data likelihood of the question collection Q as follows.

$$L(Q,G \mid \Lambda) = \mu \cdot L(Q \mid \Lambda) - (1-\mu) \cdot R(Q,G) , \quad (10)$$

where $L(Q \mid \Lambda)$ is the log-likelihood of the question collection being generated by the word distributions and therefore has the same form as Equation (5). So the regularized likelihood is further rewritten as

$$L(Q,G \mid \Lambda) = \mu \cdot \sum_{q \in Q} \sum_{w_h \in V} \{c(w_h, q) \times \log \sum_{j=1}^{k} [\pi_{q,j} \cdot p(w_h \mid \theta_j)]\} - \frac{(1-\mu)}{2} \sum_{(u,v) \in E} w(u,v) \sum_{i=1}^{k} (\pi_{u,i} - \pi_{v,i})^2 . \quad (11)$$

We can see that the introduction of the regularizer allows us to smooth the PLSA model on the question graph, and to make sure that neighboring nodes have similar weights. The parameter μ can be set to a value between 0 and 1 to control the balance between the original data likelihood and the smoothness of the question graph.

In this case, the goal of parameter estimation is to maximize $L(Q \mid \Lambda)$ and minimize $R(Q,G)$, and eventually to maximize the likelihood $L(Q,G \mid \Lambda)$. A possible way to estimate the parameters is to maximize the Q *function* with the analytical *Newton's method* (also known as the *Newton-Raphson* method) in the M step. However, this strategy increases the time cost of each

iteration [3], and therefore makes the estimation process much expensive.

To solve this problem, we adopt a simple but efficient estimation algorithm proposed in [3]. Specifically, when an iteration (in Equation (6)) ends, we examine the group distributions of neighboring nodes, and smooth $\pi_{q,j}$ for many times until the value of the Q function begins to drop down. This means that each time of regularization makes the group distributions smoother on the question graph. In the experiments of this study, we observe that the Q function generally drops after the first time of regularization. Consequently, for computational efficiency, we only perform one-time regularization. At the end of the n^{th} iteration in PLSA (see Equation (6)), the group distributions are regularized as follows.

$$\pi^n_{q,j} = \lambda \cdot \pi^n_{q,j} + (1-\lambda) \cdot \frac{\sum_{(q,v)\in E} w(q,v) \cdot \pi^n_{v,j}}{\sum_{(q,v)\in E} w(q,v)}, \quad (12)$$

where λ is a controlling factor and a smaller λ means we would like to put more emphasis on regularization. After regularization in each iteration, we normalize the parameters to ensure $\sum_{j=1}^{k} \pi_{q,j} = 1$.

3.2.4. Group Assignment. We have the same group assignment strategy for Basic PLSA, User PLSA and Regularized PLSA. When the iterative estimation process converges, we assign question q into the group which has the largest $\pi_{q,j}$, and the group mapping function is

$$group(q) = \arg\max_j (\pi_{q,j}). \quad (13)$$

4. Experiments

In this section, we conduct experimental studies to test the effectiveness of the methods. Before going to the details, we first describe the datasets and evaluation metrics.

4.1. Datasets and evaluation metrics

With the Yahoo! APIs[2], we create three datasets by downloading questions from Yahoo! Answers. These questions have been issued over a period from November 2009 to January 2010. We only focus on the *resolved* questions, meaning questions that have been given their best answers. The first dataset is collected from the *Hardware* domain and covers 6 categories including Add-ons, Desktops, Monitors, etc. The second dataset is collected from the *Internet* domain and covers 7 categories such as FaceBook, Flickr, MSN, Wikipedia, etc. The last dataset is collected from the *Science&Mathematics* domain and covers 12 categories like Botany, Chemistry, Zoology, Biology, etc. Moreover, to minimize the impact of data imbalance, we try to make the categories in each dataset have a similar number of questions. To facilitate future research, we have released our datasets at https://sourceforge.net/projects/yahoodataset/.

For preprocessing, we perform user extraction on each dataset. Some statistics of the datasets are shown in Table 1.

Table 1. **Statistics of the datasets.**

Statistics	**The Datasets**		
	Hardware	**Internet**	**Science**
# of categories	6	7	12
# of questions	1334	1407	1905
# of users	2923	3779	4495

The quality of the generated groups is evaluated with three metrics: *Purity*, *Fscore* and *normalized mutual information* (NMI). These metrics have been widely used in the evaluation of clustering [12] and community detection [13].

With the actual category labels given by users submitting the questions, we can form the true group structure $Q' = \{G'_1, G'_2, \cdots, G'_k\}$. The group structure generated by the methods is represented by $Q = \{G_1, G_2, \cdots, G_k\}$. Purity measures the extent to which each group contains questions primarily from one category. Formally, the Purity of G_j, denoted as $Purity(G_j)$, equals the proportion of the questions in the dominant class to all the questions in the group. Then the overall Purity is obtained as the weighted sum of the individual Purity values, that is,

$$Purity(Q) = \sum_{j=1}^{k} \frac{|G_j|}{|Q|} \cdot Purity(G_j), \quad (14)$$

where $|G_j|$ is the number of questions in group G_j, and $|Q|$ is the total number of questions.

Fscore is an extensively adopted metric in various areas of information retrieval. Fscore combines Precision and Recall together, and the overall Fscore value is computed in a similar manner with Purity.

NMI has recently gained popularity in cluster and community evaluation. The *mutual information* (MI) between the two structures Q and Q' is defined as

[2] http://developer.yahoo.com/answers/

$$MI(Q,Q') = \sum_{G_i,G'_j} p(G_i,G'_j)\log\frac{p(G_i,G'_j)}{p(G_i)\cdot p(G'_j)}, \quad (15)$$

and the NMI is calculated as

$$NMI(Q,Q') = \frac{MI(Q,Q')}{\max(H(Q),H(Q'))}, \quad (16)$$

where $H(Q)$ and $H(Q')$ are the entropies of the group partitions in Q and Q' respectively.

All the three metrics have values ranging from 0 to 1, and the higher their values, the better the grouping results.

4.2. Experimental results

The parameters are set in the following ways. The factors β and λ in User PLSA and Regularized PLSA are experimentally set to 0.8 and 0.7. For the parameter α in User K-means, we tune it from 0.1 to 2.0 with 0.1 as the step size, and choose the best one in terms of the evaluation metrics. All the iterative algorithms converge when the average relative difference of the objective parameters falls below 10^{-6}.

To minimize the influence of initialization, we run each method for ten times with different initial states. In particular, Basic PLSA, User PLSA and Regularized PLSA have different initial parameter values, while Basic K-means and User K-means have different initial centroid positions. Then the overall performance is evaluated by averaging the metric values over the ten runs. Table 2 shows the results for various methods.

For each dataset, incorporation of user information improves the grouping performance in any cases and on each evaluation metric. Each method, when integrating user information, outperforms its corresponding basic method, i.e., User PLSA and Regularized PLSA outperform Basic PLSA, and User K-means outperforms Basic K-means. The greatest improvement is achieved on the Internet dataset where Regularized PLSA outperforms Basic PLSA by approximately 35.7% on NMI.

When viewing K-means methods or PLSA methods as a whole, we observe that PLSA methods have better performance than K-means methods. Even the Basic PLSA method can outperform User K-means on most metrics, with the only exception on the Internet dataset where User K-means outperforms Basic PLSA on Fscore. This observation is especially true for the Science dataset. On this dataset, when considering NMI, we find that Basic PLSA performs nearly three times better than Basic K-means (NMI: 0.3712 vs 0.1233) and User K-means (NMI: 0.3712 vs 0.1314). In general, we can conclude that PLSA-based methods can generate better grouping results and thus can function as more effective solutions to the CQG problem. Then, in order to provide a more detailed perspective, we conduct extensive comparison among the three superior PLSA methods.

Among the three PLSA methods, Regularized PLSA performs best consistently on all the datasets. Both Regularized PLSA and User PLSA achieve improvements over Basic PLSA. To determine whether these improvements are statistically significant, we perform several single-tailed t-tests, and Table 3 presents the P-values for both User PLSA and Regularized PLSA compared to Basic PLSA. Both User PLSA and Regularized PLSA are able to improve Basic PLSA significantly at a 95% confidence level. However, in most cases, the P-values of Regularized PLSA are much smaller than those of User PLSA. This further confirms that Regularized PLSA is advantageous, compared with User PLSA as well as Basic PLSA. In Regularized PLSA, we regularize the values of parameters on the question graph which is

Table 2. **The grouping performance of various methods.**

Methods	Hardware			Internet			Science		
	Purity	**Fscore**	**NMI**	**Purity**	**Fscore**	**NMI**	**Purity**	**Fscore**	**NMI**
Basic K-means	0.3928	0.4076	0.1943	0.5610	0.5808	0.3407	0.2582	0.2454	0.1233
User K-means	0.3989	0.4149	0.2010	0.5897	0.6064	0.3721	0.2671	0.2518	0.1314
Basic PLSA	0.4520	0.4526	0.2471	0.5957	0.5955	0.4012	0.4979	0.4903	0.3712
User PLSA	0.4550	0.4577	0.2485	0.6060	0.6000	0.4063	0.5211	0.5147	0.3891
Regularized PLSA	0.5082	0.5013	0.2811	0.6986	0.6936	0.5444	0.5833	0.5922	0.4813

Table 3. **P-values of Regularized PLSA and User PLSA compared to Basic PLSA.**

Methods	Hardware		
	Purity	**Fscore**	**NMI**
User PLSA	0.0325	0.0343	0.2330
Regularized PLSA	2.07e-5	5.06e-4	1.72e-5
Methods	**Internet**		
	Purity	**Fscore**	**NMI**
User PLSA	0.0144	0.0071	4.11e-4
Regularized PLSA	7.55e-6	2.93e-5	2.21e-8
Methods	**Science**		
	Purity	**Fscore**	**NMI**
User PLSA	0.0011	0.0023	1.23e-4
Regularized PLSA	0.0024	9.40e-4	8.01e-6

built with involved users. Therefore, in the CQG problem, parameter regularization is a more effective way to exploit the additional user information for better grouping results.

Moreover, we investigate the time efficiency of the three PLSA methods. We have mentioned that each method is performed ten runs with different initial states. Figure 5 gives the number of iterations each method needs to converge in different runs. The iteration numbers of Basic PLSA and User PLSA are quite close to each other, especially on the Internet and Science datasets. However, Regularized PLSA generally needs less iterations than both Basic PLSA and User PLSA to terminate the estimation process. This proves that regularization on the question graph can make the parameters converge more quickly, and Regularized PLSA is a more efficient method. In practice, we probably deal with a huge number of community questions. Also, it is more desirable to provide grouping results in a real-time fashion. As a result, this time efficiency can be viewed as another advantage of Regularized PLSA.

5. Related work

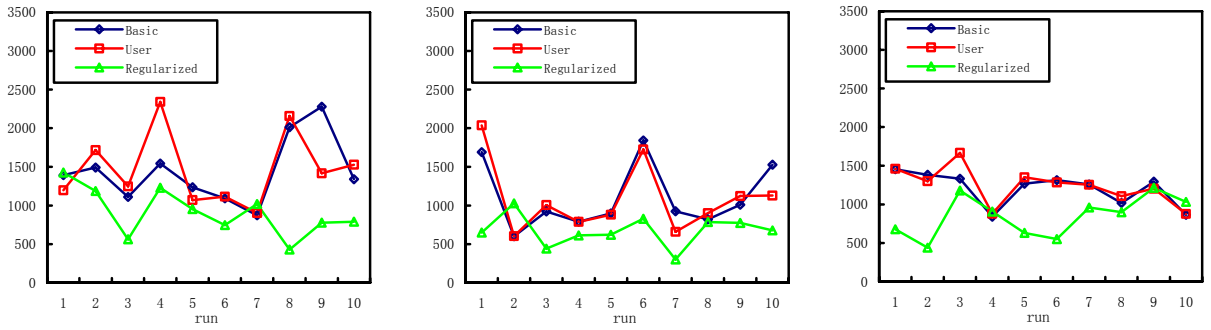
5.1. CQA research

Community Question Answering services have attracted intensive attention from research community. Liu et al. [14] introduced the problem of predicting information seeker satisfaction in question answering communities and proposed a general model for this problem. Xue et al. [1] combined a translation-based language model with a query likelihood approach for the task of finding relevant question-answer pairs when users post new questions. Wang et al. [2] proposed an *analogical reasoning* approach for measuring the linkage between questions and answers in Yahoo! Answers. By exploring the previous data, their approach can discover potential linkage within a new question-answer pair. Wang et al. [4] proposed a *syntactic tree matching* method which retrieves similar questions from CQA databases for other unaddressed questions.

Our study differs fundamentally from these work in that we deal with a different problem Community Question Grouping. What is more, we mainly focus on combining text and user information together in order to obtain better grouping performance.

5.2. Applications of PLSA

Another line of related work focuses on adapting the general PLSA framework to various specific applications. Cohn et al. [10] proposed the PHITS model which is actually the application of PLSA in document citation and link analysis. Further, they combined PLSA and PHITS jointly, and proposed the PLSA-



(b) Hardware. **(a) Internet.** **(c) Science.**

Figure 5. **Iteration numbers of the PLSA methods.**

PHITS [11] algorithm which considers content and link information in a unified framework. Also based on PLSA, Zhai et al. [6] proposed a *Cross-Collection Mixture* model to perform cross-collection and within-collection clustering, and therefore model common themes and collection-specific themes simultaneously. Lu et al. [5] adapted the PLSA model to integrate the opinions posted on blogs or forums with the opinions expressed by expert reviews. Also, Lu et al. [7] extended the basic PLSA model and developed the Structured PLSA method, which aims to generate decomposed views of short comments and product reviews for users. In spirit, our Regularized PLSA method is in line with the general framework proposed by Mei et al. [3], in which they regularized topic models with network structures. The difference is that our method is set in a totally new application scenario, i.e., Community Question Grouping in Yahoo! Answers. Also, in our experimental studies, we put emphasis on time efficiency, an aspect which is not investigated in their work.

6. Conclusion and future work

In this paper, we give the formal definition of the Community Question Grouping problem which aims at grouping questions in Yahoo! Answers automatically. We first apply the basic K-means and PLSA methods to this problem. Then we propose extensions of these basic methods which can leverage user information and achieve better grouping results. The experimental results show that user information indeed takes effects in boosting the performance of the basic K-means and PLSA. Also we conclude that Regularized PLSA is the most effective solution to the CQG problem.

For future work, we will examine whether incorporating other forms of information in Yahoo! Answers such as user ratings and best answers can improve the grouping results. Moreover, we will consider methods which can discover new or emerging categories from a collection of community questions.

7. Acknowledgments

Yajie Miao, Lili Zhao and Chunping Li are supported by National Natural Science Funding of China under Grant No. 90718022 and National 863 Project under Grant No. 2009AA01Z410.

Jie Tang is supported by National High-tech R&D Program (No. 2009AA01Z138).

8. References

[1] X. Xue, J. Jeon, and W. B. Croft, "Retrieval Models for Question and Answer Archives", In *Proceedings of SIGIR'08*, pages 475-482, 2008.

[2] X. J. Wang, X. Tu, D. Feng, and L. Zhang, "Ranking Community Answers by Modeling Question-Answer Relationships via Analogical Reasoning", In *Proceedings of SIGIR'09*, pages 179-186, 2009.

[3] Q. Mei, D. Cai, D. Zhang, and C. Zhai, "Topic Modeling with Network Regularization", In *Proceedings of WWW'08*, pages 101-110, 2008.

[4] K. Wang, Z. Ming, and T. S. Chua, "A Syntactic Tree Matching Approach to Finding Similar Questions in Community-based QA Services", In *Proceedings of SIGIR'09*, pages 187-194, 2009.

[5] Y. Lu, and C. Zhai, "Opinion Integration Through Semi-supervised Topic Modeling", In *Proceedings of WWW'08*, pages 121-130, 2008.

[6] C. Zhai, A. Velivelli, and B. Yu, "A Cross-Collection Mixture Model for Comparative Text Mining", In *Proceedings of SIGKDD'04*, pages 743-748, 2004.

[7] Y. Lu, C. Zhai, and N. Sundaresan, "Rated Aspect Summarization of Short Comments", In *Proceedings of WWW'09*, pages 131-140, 2009.

[8] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations", In *Proceeding of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281-297, 1967.

[9] T. Hofmann, "Probabilistic latent semantic indexing", In *Proceedings of SIGIR'99*, pages 50-57, 1999.

[10] D. Cohn, and H. Chang, "Learning to Probabilistically Identify Authoritative Documents", In *Proceedings of ICML'00*, pages 167-174, 2000.

[11] D. Cohn, and T. Hofmann, "The Missing Link - A Probabilistic Model of Document Content and Hypertext Connectivity", In *Proceedings of NIPS'01*, 2001.

[12] X. Hu, X. Zhang, C. Lu, E. K. Park, and X. Zhou, "Exploiting Wikipedia as External Knowledge for Document Clustering", In *Proceedings of SIGKDD'09*, pages 389-396, 2009.

[13] T. Yang, R. Jin, Y. Chi, and S. Zhu, "Combining Link and Content for Community Detection: A Discriminative Approach", In *Proceedings of SIGKDD'09*, pages 927-935, 2009.

[14] Y. Liu, J. Bian, and E. Agichtein, "Predicting Information Seeker Satisfaction in Community Question Answering", In *Proceedings of SIGIR'08*, pages 483-490, 2008.