Personalized Tag Recommendation Using Graph-based Ranking on Multi-type Interrelated Objects

Ziyu Guan[1], Jiajun Bu[1]*, Qiaozhu Mei[2], Chun Chen[1], Can Wang[1]

[1]Zhejiang Key Laboratory of Service Robot
College of Computer Science, Zhejiang University
Hangzhou, China, 310027
{guanzh, bjj, chenc, wcan}@zju.edu.cn

[2]Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801
qmei2@uiuc.edu

ABSTRACT

Social tagging is becoming increasingly popular in many Web 2.0 applications where users can annotate resources (e.g. Web pages) with arbitrary keywords (i.e. tags). A tag recommendation module can assist users in tagging process by suggesting relevant tags to them. It can also be directly used to expand the set of tags annotating a resource. The benefits are twofold: improving user experience and enriching the index of resources. However, the former one is not emphasized in previous studies, though a lot of work has reported that different users may describe the same concept in different ways. We address the problem of personalized tag recommendation for text documents. In particular, we model personalized tag recommendation as a "query and ranking" problem and propose a novel graph-based ranking algorithm for interrelated multi-type objects. When a user issues a tagging request, both the document and the user are treated as a part of the query. Tags are then ranked by our graph-based ranking algorithm which takes into consideration both relevance to the document and preference of the user. Finally, the top ranked tags are presented to the user as suggestions. Experiments on a large-scale tagging data set collected from Del.icio.us have demonstrated that our proposed algorithm significantly outperforms algorithms which fail to consider the diversity of different users' interests.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, experimentation.

Keywords

Social tagging, recommender systems, personalization, ranking.

*Corresponding author

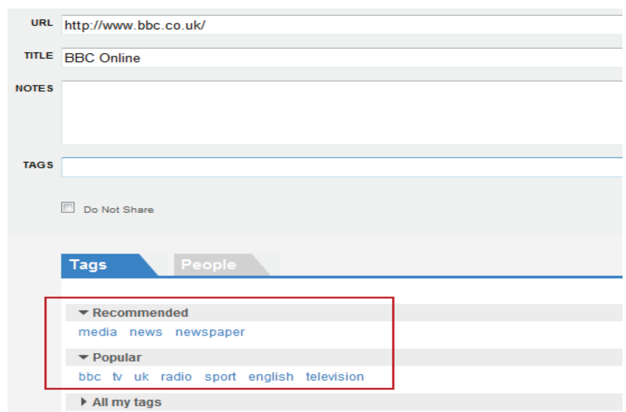


Figure 1: Tag recommendation in Del.icio.us.

1. INTRODUCTION

Tagging refers to the behavior of bookmarking resources with keywords (tags). In recent years, social tagging is becoming more and more popular in many Web 2.0 applications where users can freely annotate various resources, such as Web pages [7], academic publications [6], and multimedia objects [8]. Tag recommendation, an actively pursued research topic in tagging [22, 18, 19], is concerned with suggesting relevant tags to the users, which they could potentially use to bookmark the Web resources they visited. The motivation of tag recommendation is twofold. From the system's perspective, it aims at expanding the set of tags annotating a resource [18], thus enriches the index of resources. From the user's perspective, like all other recommendation systems, the target is to improve the experience of the user in her tagging process. In existing work, however, the latter perspective is not emphasized. Figure 1 shows the recommendation system provided by Del.icio.us [7]. When a user issues a URL, the system shows both *popular* and *recommended* tags for the URL.

Using a simple strategy, the popular tags are those frequently used by other users to annotate this URL, while the recommended tags are the intersection of this user's tag vocabulary and all the tags annotated to this URL. Such a strategy resembles collaborative filtering [1] in that it exploits collaborative knowledge and does not require the content of documents. However, the tag recommendation problem in reality is far more challenging. On one hand, since the popularity distribution of URLs in a social tagging system like Del.icio.us follows the power law [15], which indicates that most URLs are only bookmarked once or twice, it is very likely that a user issues a URL that few users, or even no user has ever bookmarked. In that case, the aforementioned strategy can hardly work. There is a need to further explore the interrelation of the Web documents (e.g., URLs) as well as the tags used to bookmark them.

On the other hand, different users may have very different preferences on the tags they would select to bookmark a document. Previous work shows that people trying to convey the same idea often disagree on how to describe it [17], due to different personal habits and different levels of expertise in related domains (the notion of *basic levels* [10]). For example, a *mobile phone* also can be called *cell phone*, *cellular phone* or *cellular telephone*. The ESP Game [21] demonstrates how difficult it is for two people to agree on even simple descriptive words for a picture. Therefore, it is desirable to develop personalized recommendation systems for social tagging, which could improve user experience and encourage users to annotate more resources.

1.1 General Idea

Neither of the two challenges is well addressed in literature. This paper addresses the problem of personalized tag recommendation for text documents. We model personalized tag recommendation as a "query and ranking" problem and propose a novel algorithm for *Graph-based Ranking of Multi-type interrelated Objects* (GRoMO) for this purpose. Specifically, we construct an affinity graph on the documents and a bipartite graph between documents and tags by using the annotation relationships. When a user issues a tagging request, both the document and the user are treated as query inputs. The tags are then ranked by the proposed graph-based ranking algorithm which considers both relevance to the document and preference of the user. Finally, the top ranked tags are presented to the user for selection.

1.2 Connection to Personalized Search

As one may see, the problem here bears some similarities with other paradigms of personalization, such as personalized search [20, 16]. Indeed, the goal here is to recommend a personalized list of tags given a user and a document, and personalized search can be viewed as recommending a personalized list of documents given a user and a query. However, there are fundamental differences between these two problems, which makes existing techniques of personalized search incapable to be applied to personalized tagging: 1) the "queries" (i.e., documents to be bookmarked) in tagging are much more informative than the queries in search, which makes it easy to compute the interrelation (e.g., similarity) between "queries." This brings in a new opportunity to explore the underlying structure of the "queries" in tagging, which is hard to achieve in search. Meanwhile, the "documents" (i.e., candidate tags) in tagging are far less informative than documents in search. This prevents us from exploring the well studied content-based methods in personalized search, such as feedback. 2) Techniques of personalized search usually rely on the initial set of relevant documents returned by the search engine. Indeed, most personalized search systems are developed by reranking the top documents in the list. In the tagging context, however, the "documents" (tags) are an open set. There is no "search engine" to obtain an initial list of relevant documents (tags) for a query (document), especially for a previously unseen query (document). The relevance of a tag can be only judged by the particular user, who could well tag a document with an acronym only used by herself.

All these typical characteristics of tagging have brought in new challenges as well as opportunities to its personalization problem. Our algorithm can naturally address these problems by leveraging the underlying structure of documents and the annotation relationships between documents and tags collectively contributed by users.

The rest of the paper is organized as follows: the next section outlines related work. Formal definition of the problem and our novel graph-based ranking algorithm GRoMO are presented in section 3. In section 4 we show how to use this algorithm to achieve personalized tag recommendation. Experiments are described in section 5, and finally, Section 6 concludes our work.

2. RELATED WORK

In this section we briefly review previous work related to ours: automatic tag recommendation and graph-based ranking algorithms.

2.1 Tag Recommendation

Xu *et al.* exploit collaborative tagging information to recommend tags [22]. Their recommendation algorithm favors tags used by a large number of people on the target document (high authority) and attempts to minimize the overlap of concepts among the recommended tags to allow for high coverage of multiple facets. This algorithm is similar to the recommendation strategy employed by Del.icio.us, which cannot handle new documents. The P-TAG algorithm [5] automatically generates personalized tags for Web pages. The generated tags are relevant to the textual content of target Web page as well as the documents residing on the surfer's Desktop. However, the problem is different from ours in that they focus on extracting personalized keywords as tags from Web pages to automate the tagging process while we concern the problem of personalized tag recommendation using collaborative tagging data. Sigurbjörnsson and Van Zwol study tag recommendation in Flickr [18]. When a user submits a photo and enters some tags, an ordered list of candidate tags is derived for each of those entered tags, based on tag co-occurrence. These lists of candidate tags are then properly merged to form the final recommendation list. Their approach depends on user entered tags and cannot be directly applied to resources. Furthermore, since they only exploit co-occurrence data, there may exist the problem of *topic drift*. A personalized, interactive tag recommendation algorithm is introduced in [9], which provides a special treatment for personal tagging data. It also depends on tag co-occurrence based on user entered tags. Thus, the aforementioned disadvantages also apply in such an algorithm. Song *et al.* developed a clustering-then-classifying framework for tag recommendation [19]. They explore spectral clustering on bipartite graph to simultaneously group tags, documents and words into clusters. Then a two-way poisson mixture model is trained on the obtained clusters. Given a query document, the algorithm computes its posterior probabilities over those clusters, and then the tags are ranked by considering both a static score and the corresponding posterior probability. Their approach could not generate personalized suggestions, and only the top-ranked tags in each cluster could ever be recommended.

2.2 Graph-based Ranking

Our work is also related to graph based ranking. There have been several developments in theory and algorithms for learning on graph data [2, 3, 4, 12, 13, 23]. They are all developed within the Laplacian-based regularization framework.

Zhou *et al.* propose a manifold ranking algorithm which ranks data objects with respect to the intrinsic manifold structure among the data objects [23]. The ranking function is obtained by preserving the local structure. In other words, two similar objects should have similar ranks. A regularization framework is thus established for this purpose. Agarwal [2] models preference training data as a (directed) weighted graph and then minimizes the empirical rank-

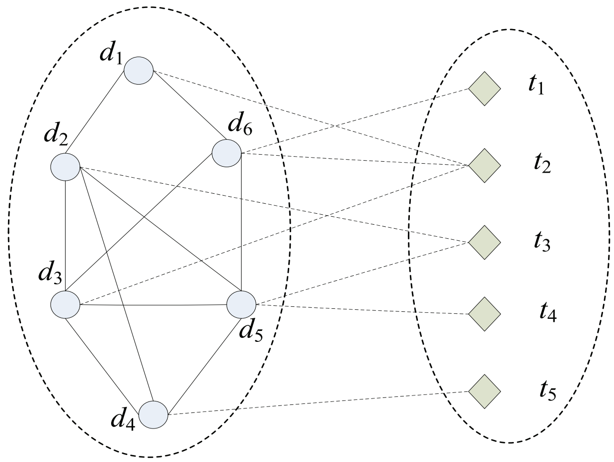


Figure 2: An illustration of the ranking problem that we consider. The solid lines represent the affinity relationships between objects in $\mathcal{D}$. The dotted lines denote the relationships between objects from $\mathcal{D}$ and those from $\mathcal{T}$.

ing error regularized by the Laplacian smoothness constraint which ensures that the ranking scores are similar for closely-connected objects. For tag recommendation, we need to deal with multi-type interrelated data objects. Therefore, the existing ranking algorithms can not be directly applied.

3. GRAPH-BASED RANKING OF MULTI-TYPE INTERRELATED DATA OBJECTS

3.1 Notation and Problem Definition

We have two types of objects, documents and tags, denoted by $\mathcal{D}$ and $\mathcal{T}$, respectively, an affinity graph $G_{\mathcal{D}}$ for $\mathcal{D}$ and a bipartite graph $H_{\mathcal{D},\mathcal{T}}$ describing annotation relationships between $\mathcal{D}$ and $\mathcal{T}$. Figure 2 illustrates the situation described above. The left dotted ellipse represents $\mathcal{D}$ and the right one represents $\mathcal{T}$. The solid lines represent the affinity relationships among documents (e.g. we can use cosine similarity or Gaussian similarity as edge weights.), and the dotted lines denote the annotation relationships between documents and tags (e.g. in Fig. 2, if tag t_2 is totally used 3 times to annotate d_6, then we can simply set the corresponding edge weight to 3.). The problem is, given query documents from $\mathcal{D}$ and/or query tags from $\mathcal{T}$, to rank documents and tags, respectively, according to their relevance to the queries. Let $\mathbf{W}$ be a $\mid \mathcal{D} \mid \times \mid \mathcal{D} \mid$ affinity matrix corresponding to $G_{\mathcal{D}}$ and $\mathbf{R}$ be a $\mid \mathcal{D} \mid \times \mid \mathcal{T} \mid$ affinity matrix corresponding to $H_{\mathcal{D},\mathcal{T}}$. Let $\mathbf{f} = [f_1, \ldots, f_{|\mathcal{O}|}]^T$ and $\mathbf{g} = [g_1, \ldots, g_{|\mathcal{T}|}]^T$ denote the ranking vectors for documents and tags, respectively. We define a query vector $\mathbf{y}_d = [y_{d1}, \ldots, y_{d|\mathcal{D}|}]^T$ in which $y_{di} = 1$ if $d_i \in \mathcal{D}$ is a query. $\mathbf{y}_t$ is defined similarly for $\mathcal{T}$. Then the goal is to infer $\mathbf{f}$ and $\mathbf{g}$ from $\mathbf{W}$, $\mathbf{R}$, $\mathbf{y}_d$ and $\mathbf{y}_t$. This definition is quite general, where given a query of $\mathbf{y}_d$ and/or $\mathbf{y}_t$, we can rank documents and tags according to $\mathbf{f}$ and $\mathbf{g}$, respectively.

3.2 Regularization Framework

We define three diagonal matrices $\mathbf{D}$, $\mathbf{D}_d$ and $\mathbf{D}_t$. The size of $\mathbf{D}$ and $\mathbf{D}_d$ is $\mid \mathcal{D} \mid \times \mid \mathcal{D} \mid$. $\mathbf{D}_t$ has size $\mid \mathcal{T} \mid \times \mid \mathcal{T} \mid$. The (i,i)-elements of $\mathbf{D}$, $\mathbf{D}_d$ and $\mathbf{D}_t$ equal to the sum of the i-th row of $\mathbf{W}$, the sum of the $i\text{-}th$ row of $\mathbf{R}$ and the sum of the $i\text{-}th$ column of $\mathbf{R}$, respectively.

$\mathbf{f}$ and $\mathbf{g}$ should be as consistent as possible with the given information, that is, $\mathbf{W}$, $\mathbf{R}$, $\mathbf{y}_d$ and $\mathbf{y}_t$. This leads to the following cost function associated with $\mathbf{f}$ and $\mathbf{g}$:

$$
\begin{aligned}
Q(\mathbf{f},\mathbf{g}) &= \frac{1}{2}\mu \sum_{i,j=1}^{|\mathcal{D}|} W_{ij} \left(\frac{1}{\sqrt{D_{ii}}} f_i - \frac{1}{\sqrt{D_{jj}}} f_j \right)^2 \\
&\quad + \eta \sum_{i=1}^{|\mathcal{D}|} \sum_{j=1}^{|\mathcal{T}|} R_{ij} \left(\frac{1}{\sqrt{D^d_{ii}}} f_i - \frac{1}{\sqrt{D^t_{jj}}} g_j \right)^2 \\
&\quad + \alpha \sum_{i=1}^{|\mathcal{D}|} (f_i - y_{di})^2 + \beta \sum_{i=1}^{|\mathcal{T}|} (g_i - y_{ti})^2, \qquad (1)
\end{aligned}
$$

where D_{ii}, D^d_{ii} and D^t_{ii} are the (i,i)-elements of $\mathbf{D}$, $\mathbf{D}_d$ and $\mathbf{D}_t$, respectively. The first and second terms of the right-hand side in the cost function are the *smoothness* constraints. The first term means that a good ranking of documents should assign similar ranking scores to similar documents. The second term means if a tag is strongly associated with a document (e.g. a tag is applied to a document many times), then they should have similar ranking scores. Please note that in the second term the ranking scores of documents and tags are normalized by $\sqrt{D^d_{ii}}$ and $\sqrt{D^t_{jj}}$, respectively. In other words, the scores are normalized by the popularity of corresponding nodes. The explanation is as follows: the documents annotated by a generally popular tag such as “design” or “2008” may not share a common topic; the large set of tags annotating a popular document is likely to contain irrelevant tags or even spam. By normalization, we can to some extent suppress popular documents and tags from dominating result rankings. The normalization in the first term is necessary for the optimization problem to be solvable. The third and fourth terms measure the difference between the obtained ranking scores and the pre-given labels which needs to be minimized. The trade-off among these terms is controlled by the regularization parameters μ, η and α and β, where $0 < \mu, \eta, \alpha, \beta < 1$ and $\mu + \eta + \alpha + \beta = 1$.

We define matrices

$$\mathbf{S}_W = \mathbf{D}^{(-1/2)} \mathbf{W} \mathbf{D}^{(-1/2)}, \qquad (2)$$

$$\mathbf{S}_R = \mathbf{D}_d^{(-1/2)} \mathbf{R} \mathbf{D}_t^{(-1/2)}. \qquad (3)$$

With simple algebraic formulations, the first term can be rewritten as follows:

$$
\begin{aligned}
& \frac{1}{2} \sum_{i,j=1}^{|\mathcal{D}|} W_{ij} \left(\frac{1}{\sqrt{D_{ii}}} f_i - \frac{1}{\sqrt{D_{jj}}} f_j \right)^2 \\
= & \frac{1}{2} \sum_{i,j=1}^{|\mathcal{D}|} W_{ij} \left(\frac{f_i^2}{D_{ii}} - 2 \frac{f_i f_j}{\sqrt{D_{ii}} \sqrt{D_{jj}}} + \frac{f_j^2}{D_{jj}} \right) \\
= & \sum_{i=1}^{|\mathcal{D}|} f_i^2 - \sum_{i,j=1}^{|\mathcal{D}|} f_i (S_{W,ij}) f_j \\
= & \mathbf{f}^T (\mathbf{I} - \mathbf{S}_W) \mathbf{f}. \qquad (4)
\end{aligned}
$$

Similarly, the second term can be computed as follows:

$$\begin{aligned}
&\sum_{i=1}^{|\mathcal{D}|}\sum_{j=1}^{|\mathcal{T}|} R_{ij}\left(\frac{1}{\sqrt{D^d_{ii}}}f_i - \frac{1}{\sqrt{D^t_{jj}}}g_j\right)^2 \\
= &\sum_{i=1}^{|\mathcal{D}|}\sum_{j=1}^{|\mathcal{T}|} R_{ij}\left(\frac{f_i^2}{D^d_{ii}} - 2\frac{f_i g_j}{\sqrt{D^d_{ii}}\sqrt{D^t_{jj}}} + \frac{g_j^2}{D^t_{jj}}\right) \\
= &\sum_{i=1}^{|\mathcal{D}|} f_i^2 + \sum_{j=1}^{|\mathcal{T}|} g_j^2 - 2\sum_{i=1}^{|\mathcal{D}|}\sum_{j=1}^{|\mathcal{T}|} f_i(S_{R,ij})g_j \\
= &\ \mathbf{f}^T\mathbf{f} + \mathbf{g}^T\mathbf{g} - 2\mathbf{f}^T\mathbf{S}_R\mathbf{g}. \qquad (5)
\end{aligned}$$

Then we can rewrite Equation (1) in the corresponding matrix-vector form:

$$\begin{aligned}
Q(\mathbf{f},\mathbf{g}) = &\ \mu\mathbf{f}^T(\mathbf{I}-\mathbf{S}_W)\mathbf{f} + \eta(\mathbf{f}^T\mathbf{f} + \mathbf{g}^T\mathbf{g} - 2\mathbf{f}^T\mathbf{S}_R\mathbf{g}) \\
&+\alpha(\mathbf{f}-\mathbf{y}_d)^T(\mathbf{f}-\mathbf{y}_d) + \beta(\mathbf{g}-\mathbf{y}_t)^T(\mathbf{g}-\mathbf{y}_t). \qquad (6)
\end{aligned}$$

Then the optimal rankings are achieved when $Q(\mathbf{f},\mathbf{g})$ is minimized:

$$\langle\mathbf{f},\mathbf{g}\rangle = \arg\min_{\mathbf{f},\mathbf{g}} Q(\mathbf{f},\mathbf{g}). \qquad (7)$$

Differentiating $Q(\mathbf{f},\mathbf{g})$ with respect to $\mathbf{f}$, we have

$$\frac{\partial Q}{\partial\mathbf{f}} = [(1-\beta)\mathbf{I} - \mu\mathbf{S}_W]\,\mathbf{f} - \eta\mathbf{S}_R\mathbf{g} - \alpha\mathbf{y}_d = 0. \qquad (8)$$

Differentiating $Q(\mathbf{f},\mathbf{g})$ with respect to $\mathbf{g}$:

$$\frac{\partial Q}{\partial\mathbf{g}} = (\beta+\eta)\mathbf{g} - \eta\mathbf{S}_R^T\mathbf{f} - \beta\mathbf{y}_t = 0. \qquad (9)$$

Substituting equation (9) into equation (8), we obtain the closed form solution for $\mathbf{f}^*$:

$$\mathbf{f}^* = \left[(1-\beta)\mathbf{I} - \mu\mathbf{S}_W - \frac{\eta^2}{\beta+\eta}\mathbf{S}_R\mathbf{S}_R^T\right]^{-1} \times \left(\alpha\mathbf{y}_d + \frac{\beta\eta}{\beta+\eta}\mathbf{S}_R\mathbf{y}_t\right). \qquad (10)$$

It can be proved that the matrix $\left[(1-\beta)\mathbf{I} - \mu\mathbf{S}_W - \frac{\eta^2}{\beta+\eta}\mathbf{S}_R\mathbf{S}_R^T\right]$ is invertible. We omit the proof due to space limitation. Once $\mathbf{f}^*$ is obtained, $\mathbf{g}^*$ can then be computed as

$$\mathbf{g}^* = \frac{\eta}{\beta+\eta}\mathbf{S}_R^T\mathbf{f}^* + \frac{\beta}{\beta+\eta}\mathbf{y}_t. \qquad (11)$$

Although the closed form is achieved, in some practical cases, the iterative form might be preferable. We can devise an iterative algorithm like HITS algorithm [14] from Equation (8) and (9). Without loss of generality, suppose $\mathbf{f}(0) = \mathbf{y}_d$ and $\mathbf{g}(0) = \mathbf{y}_t$. In the t-th iteration, we first use $\mathbf{f}(t)$ and $\mathbf{g}(t)$ computed in the last iteration to compute $\mathbf{g}(t+1)$:

$$\mathbf{g}(t+1) = \frac{\eta}{\beta+\eta}\mathbf{S}_R^T\mathbf{f}(t) + \frac{\beta}{\beta+\eta}\mathbf{y}_t, \qquad (12)$$

and then $\mathbf{f}(t+1)$ is computed from $\mathbf{f}(t)$ and $\mathbf{g}(t+1)$:

$$\mathbf{f}(t+1) = \frac{\mu}{1-\beta}\mathbf{S}_W\mathbf{f}(t) + \frac{\eta}{1-\beta}\mathbf{S}_R\mathbf{g}(t+1) + \frac{\alpha}{1-\beta}\mathbf{y}_d. \qquad (13)$$

We can see that $\mathbf{f}(t)$ and $\mathbf{g}(t)$ reinforce each other in each iteration. Substituting equation (12) into equation (13), we have

$$\begin{aligned}
\mathbf{f}(t+1) \ = &\ \frac{1}{1-\beta}\left(\mu\mathbf{S}_W + \frac{\eta^2}{\beta+\eta}\mathbf{S}_R\mathbf{S}_R^T\right)\mathbf{f}(t) \\
&+\frac{\alpha}{1-\beta}\mathbf{y}_d + \frac{\beta\eta}{(1-\beta)(\beta+\eta)}\mathbf{S}_R\mathbf{y}_t. \qquad (14)
\end{aligned}$$

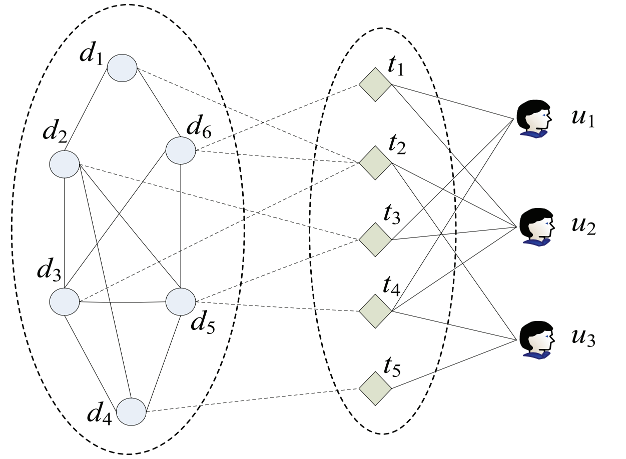


Figure 3: An illustration of the problem of personalized tag recommendation.

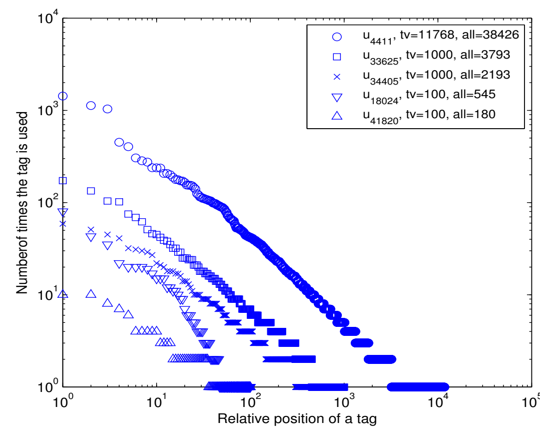


Figure 4: Frequency of tag usage as a function of the relative position (descending order by frequency) for five users.

This form of iteration involves $\mathbf{f}$ only and is more efficient for computation. By a similar analysis as in [23], it can be shown that $\mathbf{f}(t)$ converges to $\mathbf{f}^*$:

$$\mathbf{f}^* = \lim_{t\to\infty}\mathbf{f}(t). \qquad (15)$$

Once $\mathbf{f}^*$ is obtained, we can then compute $\mathbf{g}^*$ using equation (11).

3.3 Intuitive Interpretation of GRoMO

We can intuitively interpret the GRoMO algorithm as heat diffusion among vertices of the graphs through edges until a stationary state is established. The stronger the edge is (i.e. W_{ij} and R_{ij}), the more heat is transferred between the vertices connected by the edge. There are two types of diffusion: 1) diffusion among objects in $\mathcal{D}$ (first term of the right-hand side of Equation (1)); 2) diffusion between $\mathcal{D}$ and $\mathcal{T}$ (second term); The third and fourth terms assure that there are heat sources (i.e. queries) on graphs. Parameters μ, η, α and β control the relative importance of two types of diffusion and two types of sources. Hence, objects with many strong paths (paths that have many strong edges, re-weighed by the μ or η) from important query objects will have high ranking scores.

4. PERSONALIZED TAG RECOMMENDATION

In this section, we propose to solve the problem of personalized tag recommendation using the GRoMO algorithm introduced. Figure 3 shows a sketch of the problem. We continue to use $\mathcal{D}$ and $\mathcal{T}$ to represent the sets of documents and tags, respectively. We are given n users $\mathcal{U} = \{u_1, \ldots, u_n\}$ and their tagging history $\mathcal{B} = \{(u_i, d_j, t_k)\}$ where (u_i, d_j, t_k) means user u_i has used tag t_k to annotate document d_j. The task is, given a user-document pair (u, d), to rank tags considering both relevance to d and tag preference of u. There are mainly two issues we need to address: the construction scheme of the affinity graphs and personalization of the recommended tags. We also summarize the whole algorithm at the end of this section.

4.1 Affinity Graph Construction

We need to construct matrices $\mathbf{W}$ and $\mathbf{R}$. $\mathbf{R}$ is constructed as follows: obtain the users' tagging history $\mathcal{B}$ and set $R_{ij} =| \{u_k \mid u_k \in \mathcal{U}$ and $(u_k, d_i, t_j) \in \mathcal{B}\} |$. $\mathbf{W}$ is constructed using similarity measures between documents. Cosine similarity is used:

$$\mathrm{sim}(d_i, d_j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\|\mathbf{x}_i\| \|\mathbf{x}_j\|}, \tag{16}$$

where $\mathbf{x}_i$ is the feature vector representation of d_i. $\mathbf{W}$ is formed by setting $W_{ij} = \mathrm{sim}(d_i, d_j)$ if i is among the k most similar documents of j or j is among the k most similar documents of i. All the other elements of $\mathbf{W}$ are set to zero.

4.2 Personalization

The recommended tags should be biased to the user's tag vocabulary. For example, in Figure 3, both t_1 and t_2 are used to annotate d_6. Suppose they have similar meaning. We should recommend t_1 for u_1 and t_2 for u_3 when d_6 is involved. For u_2, both t_1 and t_2 should be recommended since u_2 may use synonyms when annotating documents.

To retrieve relevant tags, the query vector $\mathbf{y}_d$ is set so that only the entry corresponding to the query document d is 1:

$$y_{di} = \begin{cases} 1 & d_i = d \\ 0 & \text{otherwise} \end{cases} . \tag{17}$$

As a document becomes popular (i.e. annotated many times by users), the distribution of frequencies of tags annotating it gradually forms power law [11]. If we use document d alone as a query, the most frequently used tags may become dominant, which may not conform to the user's preference. We propose to utilize the query vector $\mathbf{y}_t$ to achieve personalization. By setting tags used by the user as queries we can promote tags that are not only relevant to the query document but also preferred by the user. The remaining question is then how to distribute query weights to each tag used by the user. We investigate tag usage patterns of users in Del.icio.us (the dataset is described in section 5), and selectively show the tag usage patterns of five users in Figure 4 (in log-log scale). The x-axis represents relative positions, from the most frequent tag to the least frequent tag, and the y-axis is the corresponding frequency of the tag. All users are represented by the IDs in our database. "u$_i$, tv= j, all= l" means user with ID i has a tag vocabulary of size j and has used them l times in total. We can see that the larger the all/tv ratio, the better the points fit a power law. This means there are a small number of tags used frequently by a user, while a large number of tags are only used once or twice. Therefore, if a user decides to use previously used tags to annotate a new document, the frequently used tags should be used rather than those in the long tail. However, assigning weights to each tag proportional to the frequency by which it was used by the user tends to bias to the most frequently used tags. We use log frequencies:

$$y_{ti} = \begin{cases} \frac{\left[\log(frequency_{u,t_i})+1\right]}{\sum_{t_j \in \mathcal{T}_u}\left[\log(frequency_{u,t_j})+1\right]} & t_i \in \mathcal{T}_u \\ 0 & \text{otherwise} \end{cases} , \tag{18}$$

where y_{ti} denotes the i-th entry of $\mathbf{y}_t$ corresponding to $t_i \in \mathcal{T}$ and $frequency_{u,t_i}$ is the number of times user u has used tag t_i. $\mathcal{T}_u$ is the tag vocabulary of user u. We add 1 to the log frequency to avoid zero weights.

The personalized tag recommendation algorithm is summarized in Algorithm 1. We employ the closed form of GRoMO in our experiments. In the online recommendation phase, we first use Equation (10) to compute the ranking vector $\mathbf{f}^*$ of documents. Then the ranking vector $\mathbf{g}^*$ of tags is computed using Equation (11). Finally, the top ranked tags are presented to the user.

Algorithm 1: Personalized Tag Recommendation

Input:
$\mathcal{U}$: the set of all users
$\mathcal{D}$: the set of all documents
$\mathcal{T}$: tag vocabulary
$\mathcal{B}$: the annotation history of all users

Offline Training

1. Construct the affinity matrix $\mathbf{W}$. Set W_{ij} to the cosine similarity between document i and j (Eq. (16)) if i is among the k most similar documents of j or j is among the k most similar documents of i. Set all other elements of $\mathbf{W}$ to zero.
2. Construct the affinity matrix $\mathbf{R}$, and set $R_{ij} = |\{u_k | u_k \in \mathcal{U}$ and $(u_k, d_i, t_j) \in \mathcal{B}\}|$.
3. Normalize $\mathbf{W}$ and $\mathbf{R}$ by $\mathbf{S}_W = \mathbf{D}^{(-1/2)}\mathbf{W}\mathbf{D}^{(-1/2)}$ (Eq. (2)) and $\mathbf{S}_R = \mathbf{D}_d^{(-1/2)}\mathbf{R}\mathbf{D}_t^{(-1/2)}$ (Eq. (3)), respectively.
4. Compute $\left[(1-\beta)\mathbf{I} - \mu\mathbf{S}_W - \frac{\eta^2}{\beta+\eta}\mathbf{S}_R\mathbf{S}_R^T\right]^{-1}$ (denoted by $\mathbf{A}^{-1}$ hereafter), with μ, η, α and β properly set.

Online Recommendation

5. Suppose user u issues document d. Set $\mathbf{y}_d$ such that the entry corresponding to d is 1 and all others equal 0 (eq. (17)). Set $\mathbf{y}_t$ according to equation (18).
6. Calculate the ranking vector $\mathbf{f}^*$ of documents:
$$\mathbf{f}^* = \mathbf{A}^{-1} \times \left(\alpha\mathbf{y}_d + \frac{\beta\eta}{\beta+\eta}\mathbf{S}_R\mathbf{y}_t\right)$$
7. Calculate the ranking vector $\mathbf{g}^*$ of tags:
$$\mathbf{g}^* = \frac{\eta}{\beta+\eta}\mathbf{S}_R^T\mathbf{f}^* + \frac{\beta}{\beta+\eta}\mathbf{y}_t.$$
8. Recommend top ranking tags to u.

---

5. EXPERIMENTS

5.1 Dataset

The dataset used in this paper is crawled from Del.icio.us. We use a user-centric strategy to collect data. In particular, we subscribed to 20 popular tags and harvested 47,355 distinct users extracted from the user field of each fetched bookmark, from Nov. 27th, 2008 to Dec. 2nd, 2008. We discarded users whose bookmarks were fewer than 30 or whose average number of tags per bookmark is less than 3. For the remaining 15,732 users, we crawled all their bookmarks from Del.icio.us (i.e. snapshots of the users' tagging data by the time the bookmark pages were crawled). About

8.9 million bookmarks and 4.4 million URLs were obtained. We then constructed a unweighted bipartite graph in which vertices were users and URLs and applied HITS algorithm [14] to find authoritative users. We selected 300 the most authoritative users and 12,677 URLs that were saved more than 6 times[1] by these users. The page content of these URLs were crawled. Among the successfully downloaded pages, we discarded non-HTML pages and non-English pages. Texts are extracted from the remaining Web pages and URLs are represented using normalized word frequency vectors. Finally, we ended up with a dataset containing 300 users, 11,795 URLs, 17,777 tags and 167,885 bookmarks, averaging 6.11 tags per bookmark.

5.2 Metrics and Compared Algorithms

For comparison, two variations of the Vector Similarity (VS) approach are employed: *Personalized Vector Similarity* (PVS) and *Global Vector Similarity* (GVS). PVS works as follows: for a query (u, d), calculate the similarity between d and each training document annotated by u using Equation (16), and then the similarity scores of s most similar documents are accumulated to the corresponding tags used by u to annotate them. The top ranked tags are recommended to u. GVS works similarly. In GVS, all training documents, tags and bookmarks are exploited, and when calculating the ranking score of tag t, the similarity of a related top-s document is weighted by the number of times tag t is applied to the document normalized by the total number of times of all tags applied to the document. Note that PVS uses completely the personal data of a query user and concerns only personalization, while GVS considers the tagging data of all users and returns the same recommendation for a document regardless of the query user. The parameter s is set empirically. When constructing $\mathbf{W}$, we empirically set $k = 50$. In experiments, we set the number of recommended tags to 10.

For evaluation, we sort each user's bookmarks by time and use the first $x\%$ (we test different values of x, from 50 to 90) from each user to form the training set. The last 10% bookmarks are treated as test data as well as the ground truth. Since we concern personalization, it is reasonable to use the users' bookmarks as ground truth. Note that we can still recommend tags to documents only appearing in the test set since we can recommend tags associated with similar documents. We use Normalized Discount Cumulative Gain (NDCG), precision and recall to evaluate recommendation algorithms. Consider a test instance (u, d). NDCG at position n is defined as

$$\mathrm{NDCG@n} = Z_n \sum_{i=1}^{n} (2^{r_i} - 1)/\log_2(i + 1), \qquad (19)$$

where r_i is the rating of tag at rank i. In our case, r_i is 1 if u actually used this tag to annotate d and 0 otherwise. Z_n is chosen so that the perfect ranking has a NDCG value of 1. Precision is defined as the number of correctly recommended tags divided by the number of all recommended tags. Recall is defined as the number of correctly recommended tags divided by the number of all tags u actually used for d.

5.3 Exploring Parameter Settings in GRoMO

GRoMO has four parameters (three free parameters) which control the relative importance of different types of score diffusion. To explore the influence of different parameter settings on the performance of GRoMO, we use the first 90% of each user's bookmarks for training and the rest for testing. Due to space limitation, we show only the results that reveal characteristics of parameters. From preliminary experiments, we found user used tags can easily reinforce one another through co-occurrence relationships bridged by documents. Although the total query weight assigned to user tags is equal to that of query document, the influence of the reinforcement phenomenon can easily dominate the score diffusion process and consequently user frequently used tags are ranked high regardless of the query document (i.e. overly biased to the user's preference). Therefore, we should keep β small. In particular, we fix two of $\{\mu, \eta, \alpha\}$ at 0.3 and vary β against the other one. Figure 5 shows the results. As we expect, when β varies against α or μ, the performance first increases then decreases. Since μ and α represent the influence from documents, Figure 5 illustrates that we indeed can achieve better performance by trading off between relevance and personalization. η controls the importance of score diffusion between documents and tags. Increasing η, however, can not only increase the score flowing from documents to tags, but also amplify the reinforcement phenomenon mentioned above. Nevertheless, varying β against η exhibits similar performance curve as varying β against μ or α. It seems β is more crucial. We select the best parameter setting for the remaining experiments: $\mu = 0.3$, $\eta = 0.17$, $\alpha = 0.5$ and $\beta = 0.03$.

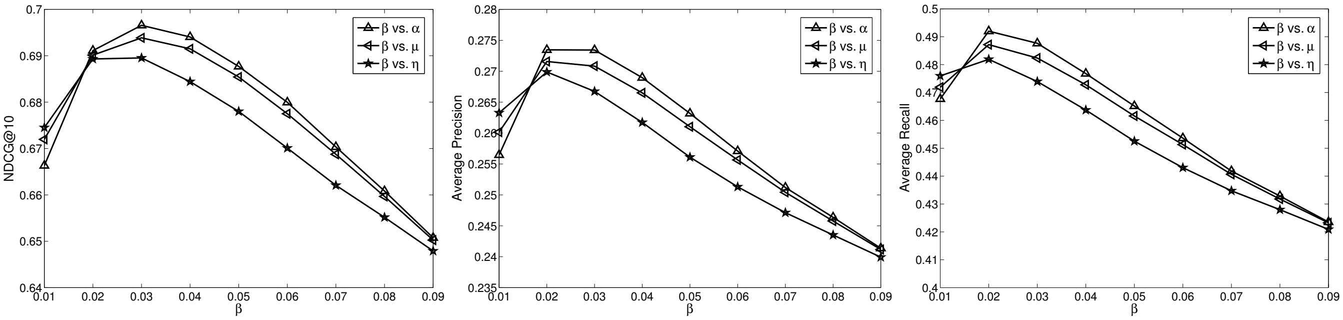
5.4 Performance Comparison

We compare GRoMO with the other two algorithms with respect to different amounts of training data. Specifically, we use the first 50%, 60%, 70%, 80% and 90% of each user's bookmarks as training data, respectively. At each run, the last 10% of each user's bookmarks are used for testing. The results are presented in Figure 6. GRoMO is clearly the winner. Our algorithm significantly outperforms both PVS and GVS (by t-test, $\alpha = 0.05$). GVS performs better than PVS. This can be explained by: 1) users do have some degree of consensus on which tags to apply (i.e. the power law distribution of tags annotating a document [11]); 2) considering the diverse interests of users, it may be difficult to find documents directly related to the current document from the user's personal tagging history. However, by combining both collaborative and personal data, we can achieve better performance on all evaluation metrics, as shown by the curves of GRoMO. There is a drop of performance for GRoMO on NDCG@10 when the training data changes from 80% to 90%, which is unexpected. Though precision and recall increase (Figure 6(b) and 6(c)), it seems the relevant tags tends to reside at lower positions within the top 10 tags compared to the case of 80% training data. We tune the parameters of GRoMO using 90% data as training data, but it performs even better on NDCG with less training data. We also report the performance of the three recommendation algorithms on NDCG@1, NDCG@3 and NDCG@5, as shown in Table 1. By Wilconxon test, in most cases our proposed algorithm significantly outperforms the other methods at significance level $\alpha = 0.01$. The 90% case is again due to the performance drop of GRoMO on NDCG. In general, we find that GVS can achieve competitive performance when training data is abundant. However, such a condition is rarely satisfied in real world.

Table 2 shows an example demonstrating that our algorithm can correctly adapt to the users' personal habits of tag usage. The page "http://www.brand-name-coupons.com/how-to-search-amazon-for-deals/" tells people how to find discounted deals in Amazon[2]. We show three users' annotations (the ground truth) in the last 10% testing data and the corresponding tag lists suggested by GRoMO

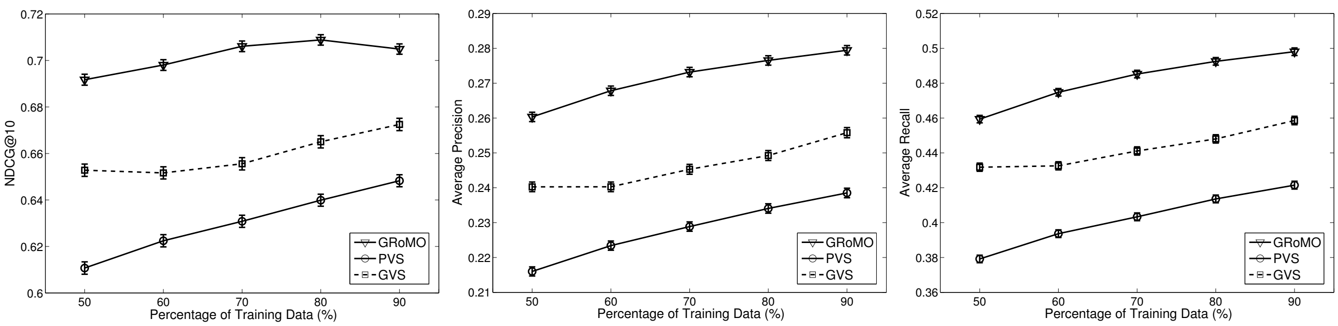
[1]Bookmarks with zero tags are discarded because they are of no use to both training and testing.

[2]http://www.amazon.com/



(a) (b) (c)

Figure 5: Exploring the influence of different parameter settings on the performance of GRoMO. 90% of each user's data is used for training. We fix two of $\{\mu, \eta, \alpha\}$ at 0.3 and vary β against the other one. The figures show performance measured by (a) NDCG@10, (b) Precision and (c) Recall.



(a) (b) (c)

Figure 6: Comparison of recommendation algorithms in terms of (a) NDCG@10 (b) Precision and (c) Recall. Different percentages of training data are considered. We use the last 10% bookmarks of each user for testing. The performance are averaged over all the test instances.

(trained on 90% data). In our dataset, we find that the majority of users annotating this URL use *bargains*, *discount* and *coupons*. Only a few users use *cheap* to express the same meaning. From Table 2 we can see that our algorithm correctly adapts to u_{37982}'s preference. We also find that in our dataset u_{37982} had already used *cheap* several times before he/she annotated this URL. The same analysis can be derived for u_{5472} with respect to *tips*. Note that in the recommended list for u_{5472} there are some irrelevant tags. They are mainly the user's most frequently used tags.

6. CONCLUSIONS

We address the problem of personalized tag recommendation in social tagging systems. We model it as a "query and ranking" problem and propose a novel graph-based ranking algorithm of Multi-type interrelated objects (GRoMO). When a user issues a tagging request, both the document and the user are treated as queries, accounting for relevance and personalization, respectively. After applying GRoMO, the top ranked tags are presented to the user. Although we consider text data in this paper, our algorithm is general and can be applied to any social tagging systems as long as a notion of similarity between resources is defined. We compare GRoMO with Personalized Vector Similarity and Global Vector Similarity on a dataset crawled from Del.icio.us. The results show that GRoMO is effective and outperforms the other algorithms. For future work, we would like to examine the efficiency issue of our algorithm. It would be also interesting to apply GRoMO to other types of tagging systems and other recommendation problems.

7. ACKNOWLEDGMENTS

This work was supported by National Key Technology R&D Program (2008BAH26B02 & 2007BAH11B06) and the Key S&T Projects of Zhejiang Province (2007C13019). We would like to thank Zhengguang Chen, Tianyu Li and Zhaohong Chen for algorithm implementation and detailed discussion.

8. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 2005.

[2] S. Agarwal. Ranking on graph data. In *Proceedings of the 23rd international conference on Machine learning*, 2006.

[3] D. Cai, X. He, and J. Han. Spectral regression for efficient regularized subspace learning. In *Proc. Int. Conf. Computer Vision (ICCV'07)*, 2007.

[4] D. Cai, X. He, and J. Han. SRDA: An efficient algorithm for large scale discriminant analysis. *IEEE Transactions on Knowledge and Data Engineering*, 20(1):1–12, January 2008.

[5] P. A. Chirita, S. Costache, W. Nejdl, and S. Handschuh. P-tag: large scale automatic generation of personalized annotation tags for the web. In *Proceedings of the 16th international conference on World Wide Web*, pages 845–854, 2007.

Table 1: Comparison of recommendation algorithms in terms of NDCG@1, NDCG@3 and NDCG@5. We perform Wilcoxon test at different significance levels. "*" and "" mean the performance is significantly better compared to others at significance level $\alpha = 0.05$ and $\alpha = 0.01$, respectively.**

	NDCG@1			NDCG@3			NDCG@5		
Training Data (%)	GRoMO	PVS	GVS	GRoMO	PVS	GVS	GRoMO	PVS	GVS
50	**0.5422****	0.4518	0.5136	**0.5357****	0.4617	0.5209	**0.5863****	0.5127	0.5593
60	**0.5434****	0.4645	0.5202	**0.5381****	0.4726	0.5122	**0.5897****	0.5234	0.5542
70	**0.5558****	0.4759	0.5301	**0.5474****	0.4825	0.5179	**0.5982****	0.5314	0.5593
80	**0.5534**	0.4873	0.5478	**0.5483***	0.4907	0.5322	**0.5990****	0.5398	0.5706
90	0.5397	0.4992	**0.5590****	0.5387	0.5006	**0.5455****	**0.5911**	0.5495	0.5795

Table 2: Three Users' annotations in the last 10% testing data for the URL "http://www.brand-name-coupons.com/how-to-search-amazon-for-deals.html". The 10 tags recommended by GRoMO (trained on 90% data) are denoted by "GRoMO Recommended". Tags with bold font indicate matches with the ground truth.

URL: http://www.brand-name-coupons.com/how-to-search-amazon-for-deals/		
UserID	Ground Truth	GRoMO Recommended
8414	amazon, bargains, Coupons, deals, discount, howto, shopping	**amazon**, **deals**, **bargains**, **shopping**, discounts, coupons, bargain, s3, search, **discount**
37982	amazon, cheap, coupons, sales, shopping	**amazon**, deals, **shopping**, bargains, **coupons**, discounts, s3, search, **cheap**, discount
5472	blog, howto, shopping, tips, tools	**shopping**, **tools**, free, web, design, reference, software, **howto**, **tips**, amazon

[6] CiteULike. http://www.citeulike.org.

[7] Del.icio.us. http://delicious.com.

[8] Flickr. http://www.flickr.com.

[9] N. Garg and I. Weber. Personalized, interactive tag recommendation for flickr. In *Proceedings of the 2008 ACM conference on Recommender systems*, 2008.

[10] S. A. Golder and B. A. Huberman. Usage patterns of collaborative tagging systems. *Journal of Information Science*, 32(2):198, 2006.

[11] H. Halpin, V. Robu, and H. Shepherd. The complex dynamics of collaborative tagging. In *Proceedings of the 16th international conference on World Wide Web*, 2007.

[12] X. He and P. Niyogi. Locality preserving projections. In *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2003.

[13] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang. Face recognition using laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005.

[14] J. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proc. 9th Annual ACM-SIAM Symposium Discrete Algorithms*, pages 668–677, January 1998.

[15] X. Li, L. Guo, and Y. E. Zhao. Tag-based social interest discovery. In *Proceedings of the 17th international conference on World Wide Web*, 2008.

[16] Q. Mei, D. Zhou, and K. Church. Query suggestion using hitting time. In *Proceedings of the 16th ACM International Conference on Information and Knowledge Management*.

[17] S. Sen, S. K. Lam, D. Cosley, D. Frankowski, J. Osterhouse, F. M. Harper, and J. Riedl. tagging, communities, vocabulary, evolution. In *Proceedings of the ACM 2006 conference on Computer supported cooperative work*, pages 181–190, 2006.

[18] B. Sigurbjörnsson and R. van Zwol. Flickr tag recommendation based on collective knowledge. In *Proceedings of the 17th international conference on World Wide Web*, 2008.

[19] Y. Song, Z. Zhuang, H. Li, Q. Zhao, J. Li, W. C. Lee, and C. L. Giles. Real-time automatic tag recommendation. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 515–522, 2008.

[20] J. T. Sun, H. J. Zeng, H. Liu, Y. Lu, and Z. Chen. Cubesvd: a novel approach to personalized web search. In *Proceedings of the 14th international conference on World Wide Web*, 2005.

[21] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326, 2004.

[22] Z. Xu, Y. Fu, J. Mao, and D. Su. Towards the semantic web: Collaborative tag suggestions. In *Collaborative Web Tagging Workshop at WWW2006*, 2006.

[23] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Scholkopf. Ranking on data manifolds. In *18th Annual Conference on Neural Information Processing Systems*, 2003.