A Probabilistic Model for Personalized Tag Prediction

Dawei Yin Zhenzhen Xue Liangjie Hong Brian D. Davison
Department of Computer Science & Engineering, Lehigh University
Bethlehem, PA, USA
{day207, zhx208, lih307, davison}@cse.lehigh.edu

ABSTRACT

Social tagging systems have become increasingly popular for sharing and organizing web resources. Tag recommendation is a common feature of social tagging systems. Social tagging by nature is an incremental process, meaning that once a user has saved a web page with tags, the tagging system can provide more accurate predictions for the user, based on the user's incremental behavior. However, existing tag prediction methods do not consider this important factor, in which their training and test datasets are either split by a fixed time stamp or randomly sampled from a larger corpus. In our temporal experiments, we perform a time-sensitive sampling on an existing public dataset, resulting in a new scenario which is much closer to "real-world".

In this paper, we address the problem of tag prediction by proposing a probabilistic model for personalized tag prediction. The model is a Bayesian approach, and integrates three factors—an ego-centric effect, environmental effects and web page content. Two methods—both intuitive calculation and learning optimization—are provided for parameter estimation. Pure graph-based methods which may have significant constraints (such as every user, every item and every tag has to occur in at least p posts) cannot make a prediction in most "real world" cases while our model improves the F-measure by over 30% compared to a leading algorithm on a publicly-available real-world dataset.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.2.8 [**Database Management**]: Database applications—*Data Mining*; H.4.0 [**Information Systems Applications**]: General; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; H.1.2 [**Models and Principles**]: User/Machine Systems—*Human Information Processing*

General Terms

Algorithms, Experimentation, Measurement, Performance

KDD'10, July 25–28, 2010, Washington, DC, USA.

Keywords

social tagging, tag recommendation, personalized tag prediction

1. INTRODUCTION

Collaborative tagging systems, also known as social bookmarking systems, have become increasingly popular for sharing and organizing web resources. In collaborative tagging systems, users add metadata in the form of descriptive terms, called tags, to describe web resources. Social bookmarking has already showed its value in many areas, such as query expansion [2], web search [1], personalized search [17, 21], web resource classification [23] and clustering [14]. A better understanding and prediction of tags on web pages is quite meaningful, especially in those areas.

Tag recommenders can assist users with the tagging process by suggesting a set of tags that users are likely to use for a web resource. Personalized tag recommenders which take a user's previous tagging behaviors into account when making suggestions usually have better performance compared with general tag recommenders. In short, the goal of a personalized tag recommender is to predict tags for each user specifically and effectively, given a web resource.

Personalized tag prediction has become a popular research topic. The two main directions for these systems are content-based approaches and graph-based approaches. Content-based methods, which usually model users' preferences from textual information (e.g., web pages, academic papers and tags), can predict tags for new users and new web resources. Graph-based approaches, while often having stronger assumptions than content-based ones, typically provide better performance. For example, one such assumption is the CORE p [9] requirement, in which every user, every item and every tag has to occur at least p times in the training set. However, in most cases, such an assumption is not realistic. Actually, tag recommenders are often asked to recommend tags when the system knows nothing about the web resource or the user. Our analysis will show that in a real world scenario, the probability of a web resource being new to a tag recommender is more than 0.9. Comparing both kinds of approaches, the content-based approach has the advantage that it can predict tags for any user and any web resource, while the overall performance is not as good as graph-based approach. Graph-based approaches which require CORE p can have significantly better performance, but can only predict tags for certain groups of users and web resources, preventing them from being widely applicable. Thus, a better tag recommender should be able to recommend tags for new users or new web resources, and still have reasonably good performance. Our tag recommender has such functionality, in part by incorporating various factors. We believe that in the real world, when a user is tagging web pages, at least the

following three factors will affect the choice of tags which he will finally use.

The ego-centric effect. A given user will have some specific interests and will tend to bookmark similar items with similar tags based on the user's vocabulary. Assume for example that a user is interested in "development" and he has already tagged many web pages about development by using "C++", "java", and "tutorials". When he bookmarks a new web page, intuitively, this item will be relevant to "development" with high probability. That is, the content of this new web page is very likely to be similar to web pages that the user tagged previously. In addition, on this web page, the user will also tend to use similar tags which he used before. We name this effect, which is from the user himself, the ego-centric effect.

Environmental effects. A user may be influenced by other users. When a user is tagging some web page, he may adopt tags which are used frequently by other similar users. For instance, a user may often use the tag "java" previously, but never use the tag "tutorial". Suppose that there is another user who is similar to this user—say, they both frequently use tag "java"—but in addition frequently uses the tag "tutorial". In this case, when the user is trying to bookmark an item which is similar to the items where the other user already has tagged as "tutorial", the probability of this user using both "java" and "tutorial" is higher (even if this user never used the tag "tutorial" before). In addition, some users may discover resources within the tagging system; that is, they find interesting items which other users have already tagged. In this case, the probability of this user using the same tags will be very high (most graph-based recommenders adopt this strategy). Another aspect is that some current tagging systems allow users to set up relationships with other users, e.g., delicious [22, 20]. This also strengthens the influence of neighboring users. We call all of these kinds of effects environmental effects.

Item content. For an item which already exists in past bookmarks, we can get some prediction hints from the tags which have already been used on this item. However, we have found that tag prediction systems may need to face new items more than 90 percent of the time. Thus, strictly graph-based recommenders will not work on these new items. When facing a new item, content analysis is necessary. Even if the item is not a new item, content analysis is still quite useful, because other items with similar topics will provide hints for tag prediction.

Besides proposing a probabilistic model for tag prediction, we also analyze the task of tag prediction itself. In previous literature, there are generally two kinds of approaches to evaluate social tagging systems. One is to randomly split a dataset into training and test sets without considering time information, and the other is to use a fixed time point to split the whole dataset into training and test sets. However, neither of the two methods well represent real world scenarios. The first case is not reasonable because the training set contains the posts with earlier timestamps and also posts with later timestamps. The second case is different from real systems in the sense that a real tagging system is likely to operate in an online mode (i.e., an incremental mode). Actually, for a given post, all posts in the whole data with earlier timestamps can be and should be used as training data. So the training data is different for each test post. This approach also has the benefit of utilizing as much information as possible.

In this paper, our contributions are as follows:

- We perform time-sensitive sampling on an existing public dataset, and propose a new use case of tag prediction which is closer to real world cases.
- We present a novel probabilistic model for personalized tag prediction.
- We demonstrate on public data that our method significantly outperforms a strong existing method when performing online tag prediction.

In Section 2, we briefly review some recent work on tag recommendation and related topics. Section 3 precisely defines the problem and notation. Section 4 introduces a general probabilistic model for tag prediction and extends it to personalized prediction. Section 5 reports online analysis. Section 6 reports our experiments. Section 7 concludes and outlines future work.

2. RELATED WORK

Personalized tag recommendation, as a special case of collaborative filtering, is a recent topic in recommender systems. The two main directions for these systems are content-based approaches and graph-based approaches.

Content-based methods, which usually encode users' preferences from textual information (e.g., web pages, academic papers, tags), can predict tags for new users and new items. One state-of-the-art content-based tag recommendation system [12] utilized several tag sources including item content and user history to build both profiles for users and tags. New tags are checked against user profiles, which are rich, but imprecise sources of information about user interests. The result is a set of tags related to both the resource and the user. Depending on the character of processed posts, this set can be an extension of the common tag recommendation sources, namely resource title and resource profile.

Graph-based approaches, which usually have stronger assumptions than content-based ones (e.g., requiring every user, every item and every tag to occur in at least p posts), can provide better performance. Early work like FolkRank, introduced by Hotho et al. [8], is an adaptation of PageRank that can generate high quality recommendations which are shown empirically to be better than other previous proposed collaborative filtering models [9]. Guan et al. [6] proposed a framework based on graph Laplacian to model interrelated multi-type objects involved in the tagging system. Tags are ranked by a graph-based ranking algorithm which takes into consideration both relevance to the document and preference of the user. Recently, factorization models (also considered as graph-based approaches) show very successful evaluation results on personalized tag recommendation problems. In [19], Symeonidis et al. proposed a method based on Higher-Order-Singular-Value-Decomposition (HOSVD)—which corresponds to a Tucker Decomposition (TD) model optimized for square-loss.In [15], Rendle et al. proposed a better learning approach for TD models, which is to optimize the model parameters for the AUC ranking statistic (area under the ROC curve). Rendle and Schmidt-Thieme [16] further presented a special case of the TD model with linear runtime for both learning and prediction. In experiments on real world datasets, they showed that the model outperforms TD largely not only in run-time but also can achieve better prediction quality. Non-personalized tag recommenders—i.e., for a given item they recommend to all users the same tags—have also attracted a lot of attention (e.g., [7, 18]). However, in [15], the authors showed that personalized tag recommendation systems empirically outperform the theoretical upper bound for any non-personalized tag recommender. Garg et. al[5] propose a personalized interactive tag suggestion system which suggests tags based on the ones that a user entered most recently. They employ a naive Bayes classifier which is only based on tag co-occurrences.

Graph-based methods are also popular in other fields of collaborative filtering. Many of the best performing models (e.g., [11, 10] on the Netflix Challenge[1]) are based on matrix factorization. Also for related task of item prediction, factorization models are known to outperform models like k-nearest neighbors collaborative filtering or the Bayesian model URP [13].

Recent research also show that users are much more likely to use their recently used tags. Zhang et. al [24] investigate the recurrence dynamics of social tagging. Time information is also important to recommend high-quality tags to users.

3. DEFINITIONS

In a social tagging system, users can bookmark web pages by assigning tags to them. The system can also retrieve the content of a web page which the user is bookmarking and based on content, the system can recommend to the user some personalized tags. The task of recommending tags to users is called tag recommendation.

A similar task is tag prediction which needs to predict the tags which the user will use on some bookmarks. This can also be personalized; that is, given a user and a set of bookmarks without tags, the algorithm should predict which tags the user will use on each bookmark. In order to predict or recommend tags for a specific user precisely, the recommender should first understand the user well. Because different users have different preferences and interests, for some users, the bookmarks the user saves may tend to be similar or in the same topic. In addition, on similar bookmarks, the tags which different users use may be similar. But for other users, even if they save the same or similar web pages, they may use different tags because of different perspectives and different preferences.

Here, we formalize the definitions. Let U be the set of all users, I be the set of all items (they sometimes are also called objects, resources, or web pages in other literature) and T be the set of all tags. For past tagging information, we have existing ternary relations S, and $S \subseteq U \times I \times T$. Thus, each single record $(u, i, t) \in S$ means that user u has tagged an item i with the tag t. Here, we also define P_s as all the past distinct user-item combination:

$$P_s = \{(u, i) | \exists t \in T : (u, i, t) \in S\}$$

Thus, when the current user u_c is trying to add an item i_c, the task is to recommend a list of tags to the potential post (u_c, i_c), based the past posts S, which we also call training data.

4. PROBABILISTIC MODEL

The tag prediction problem can be treated as the reverse of web search. In web search, users submit a list of terms as a query, and then the relevant web pages i will be retrieved and the web pages can be ranked by $P(i|t)$, the probability of the page i being relevant to the query t. Here, the list of terms can be considered as a list of tags. Without considering personal information (non-personalized tag prediction), the general tag prediction could be that given a web page i, retrieve a list of potential tags. The tags can be ranked by $P(t|i)$. According to Bayesian theory, we have

$$P(t|i) = \frac{P(i|t) \cdot P(t)}{P(i)} \tag{1}$$

In Equation 1, $P(t|i)$ means the probability of using tag t given an item i. $P(i|t)$ means the frequency of item i in a set of items which are tagged by t. $P(t)$ is the prior probability of tag t. If the tag t appears more frequently, it will hold a higher prior probability. If the item i exists in past posts which can be considered as the training data, then $P(i|t)$ can be easily estimated by simply using the number of occurrence of (i, t)—$N_{i,t}$. However, if the item i does not exist in the past posts, that is, i is a new item, it is difficult to estimate the probability $P(i|t)$. One possible solution is to use the content of the item.

The content of item i can be represented by a language model. The most straightforward model is a unigram language model. The item i is treated as a bag of words $W = \{w | w \text{ appears in item } i\}$. Here, if the word-independence assumption is made, the probability of item i given the tag t will be:

$$P(i|t) = \prod_{w \in W_i} P(w|t) \tag{2}$$

According to Equation 2, we know that the probability $P(i|t)$ can be broken down into the production of word-level probabilities $\prod_{w \in W_i} P(w|t)$. $P(w|t)$ means the likelihood of the word w appearing in the item's content, given a tag t. Given a item i, the number of occurrence of w is denoted as $N_{w,i}$. Given a tag t, the number of occurrence of w is denoted as $N_{w,t}$ which can be calculated as follows:

$$N_{w,t} = \sum_{i \in I} N_{w,i} \cdot N_{i,t}$$

To estimate $P(w|t)$, we can assume that words obey the following distribution:

$$P(w|t) = \frac{N_{w,t}}{N_t}$$

Then, maximum likelihood estimation (MLE) can be used to estimate the parameter N. To maximize the probability of the word w, we have:

$$N_t = \sum_w N_{w,t}$$

By combining Equations 1 and 2, general tag prediction can be expressed as:

$$P(t|i) = \frac{\prod_{w \in W_i} P(w|t) \cdot P(t)}{P(i)} \tag{3}$$

4.1 Personalized Tag prediction

While we have shown how to perform general tag prediction, personalized tag prediction is more preferable. In social tagging systems, individual users may have specific interests and tend to bookmark similar web pages by using similar tags. For different users, the prior probability of tags is often different, and the language model of tags is also different. Even if two users bookmark the same item, the tags they use can also be different because of their various interests, perspectives and preferences. Rendel et al. [15] show that personalized tag prediction systems empirically outperform the theoretical upper bound for any non-personalized tag recommender. In our probabilistic model, the general tag prediction can be simply extended to personalized prediction by involving the ego-centric effect. Given a user u, the personalized tag prediction can be:

$$P(t|i, u) = \frac{P(i|t, u) \cdot P(t|u)}{P(i|u)} \tag{4}$$

Here, $P(t|i, u)$ means that given a user u, the probability that tag t is applied to the item i. $P(i|t, u)$ means the likelihood of item i given a tag t and user u. $P(t|u)$ is also the conditional prior probability of tag t, given the user u. It can be easily understood that Equation 4 is based on a set of past posts S—that is, for the specific user u_c, $S_{u_c} = \{(u_c, i, t) | (u_c, i, t) \in S\}$. Similar to non-personalized tag prediction, to incorporate the content of items, replacing $P(i|t, u)$ in Equation 4, the personalized tag prediction will

[1] http://www.netflixprize.com/

become:

$$P(t|i,u) = \frac{\prod_{w \in W_i} P(w|t,u) \cdot P(t|u)}{P(i|u)} \quad (5)$$

However, in this model, if a user has not yet used a tag, we cannot rank it. Because that if tag t has not been used by user u, the prior probability $P(t|u) = 0$, and we cannot get the $P(i|t,u)$ either. According to Equation 5, for this new tag t, the $P(t|d,u)$ will be always 0. Thus, the candidate tags will be constrained to the set of tags which the user has used before. Obviously, such candidate tags are often quite limited.

In addition, when users are trying to bookmark some web pages, the three factors mentioned previously will affect the tags which the user will finally use: the ego-centric effect, environmental effects and item content. In Equation 5, the ego-centric effect is modeled by the whole equation and item content is modeled by $\prod_{w \in W_i} P(w|t,u)$. To model environmental effects, we involve the probability of neighbor effects $P(u_k|u)$, that is, given the current user u, the probability of user u_k affecting user u. When $u_k = u$, then $P(u|u)$ represents the exact weight of the ego-centric effect. Thus,

$$\sum_{u_k} P(u_k|u) = 1 \quad (6)$$

When we integrate the environmental effects into Equation 5, we get

$$P(t|i,u) = \frac{\prod_{w \in W_i} \sum_{u_k} P(w|t,u_k)P(u_k|u) \times \sum_{u_k} P(t|u_k)P(u_k|u)}{\sum_{u_k} P(i|u_k)P(u_k|u)} \quad (7)$$

This equation enlarges the tag candidates for tag prediction and also integrates the the environmental effects. Given a user u and an item i, the probability of tag t being used will be $P(t|i,u)$. Our algorithm will rank the tags by the value $P(t|i,u)$. Because the evidence $P(i|u)$ is the same for all tag candidates, then

$$\begin{aligned} & P(t|i,u) \\ & \propto \prod_{w \in W_i} \sum_{u_k} P(w|t,u_k)P(u_k|u) \times \sum_{u_k} P(t|u_k)P(u_k|u) \\ & \propto \sum_{w \in W_i} \log \sum_{u_k} P(w|t,u_k)P(u_k|u) + \log \sum_{u_k} P(t|u_k)P(u_k|u) \end{aligned} \quad (8)$$

We refer to Equation 8 as,

$$\begin{aligned} y_{u,i,t} = \sum_{w \in W_i} & \log \sum_{u_k} P(w|t,u_k)P(u_k|u) \\ & + \log \sum_{u_k} P(t|u_k)P(u_k|u) \end{aligned} \quad (9)$$

Then, given a user u and an item i, our algorithm will rank the tags by the value $y_{u,i,t}$. If we define the probability $P(u|u)$ as α or $p_{u,u}$ and environmental effects $P(u_k|u)$ as $p_{u_k,u}$, then split the ego-centric effect part and environmental effects part and Equation 9 can be rewriten as

$$\begin{aligned} y_{u,i,t} = \sum_{w \in W_i} & \log(\sum_{u_k \neq u} p_{u_k,u} \cdot P(w|t,u_k) + \alpha \cdot P(w|t,u)) \\ & + \log(\sum_{u_k \neq u} p_{u_k,u} \cdot P(t|u_k) + \alpha \cdot P(t|u)) \end{aligned} \quad (10)$$

where $\sum_{u_k \neq u} p_{u_k,u} + \alpha = 1$ and $\sum_{u_k \neq u} p_{u_k,u}$ can be also called the weight of environmental effects and α can be called the weight of ego-centric effect or ego weight. To avoid zero probability, for $P(w|t,u_k)$, we use simple Laplace smoothing in our experiments.

4.2 Parameter Estimation and Optimization

In our model, Equation 9, we have already introduced the unigram language model for $P(w|t,u_k)$. Another $P(t|u_k)$ can be calculated through the number of occurrence of tag t within the posts of user u_k. The hard problem is to estimate the ego-centric effect and environmental effects $P(u_k|u)$.

4.2.1 *Intuitively calculating* $P(u_k|u)$

Given a user u, to calculate the probability of another user's influence—$P(u_k|u)$, we consider that users can be represented in tag space. In the set of past posts S, each user has a set of tags, which describes the interests of the user. In other words, each user has a distribution of tags. The vector of tag occurrences can be used to represent to the user. For the user u_k,

$$V_{u_k} = [n_{u_k,t_0}, n_{u_k,t_1}, ...n_{u_k,t_i}, ...n_{u_k,t_m}]$$

Here, n_{u_k,t_i} means the number of times of user u_k uses tag t_i. For the user u_k, the prior probability of the tag t_j can be calculated by

$$P(t_j|u_k) = \frac{n_{u_k,t_j}}{\sum_{i=0}^{m} n_{u_k,t_i}}$$

If two users have similar interests, then they may have similar distributions of tags and there will be a higher probability of affecting each other. Here, for user u, if we assume that the similarity of interests between user u and user u_k is directly proportional to the probability of u_k affecting u—$P(u_k|u)$, then

$$P(u_k|u) \propto sim(u,u_k) = \frac{V_u \cdot V_{u_k}}{|V_u| \times |V_{u_k}|}$$

Where $sim(u,u) = 1$, the ego weight will be always larger than the weight of other individual users. After normalizing similarity, we can simply set the

$$P(u_k|u) = \frac{sim(u,u_k)}{\sum_{u_i} sim(u,u_i)} \quad (11)$$

We refer to this calculation as user-tag-user similarity. We can also manually cut off users by using a threshold. For experiments, the number of neighbor users can be set as runtime parameter k. Only the most similar k neighbor users will be counted.

4.2.2 *Learning algorithm*

Our intuitive estimation is only a rough method of estimating $P(u_k|u)$. In some cases, it may not be precise. For example, user u_k may use the same tags that user u used on item i, to tag another item i_k, while the content i and i_k are completely different. Thus, different users may use the same tags with different intents or perspectives for tagging web pages. Our previous method will over-estimate the probability $P(u_k|u)$ in this situation. To solve this problem, we design a learning algorithm to calculate $P(u_k|u)$ iteratively. For a post (u,i), the algorithm ranks tags by $y_{u,i,t}$. We use the similar objective function as in [15], which uses the "post-based ranking interpretation" and maximizes the ranking statistic AUC (area under the ROC-curve).

$$AUC(\hat{\theta},u,i) = \frac{1}{|T^+_{u,i}||T^-_{u,i}|} \sum_{t^+ \in T^+_{u,i}} \sum_{t^- \in T^-_{u,i}} H_{0.5}(y_{u,i,t^+} - y_{u,i,t^-}) \quad (12)$$

where

$$H_\beta = \begin{cases} 0, & x < 0 \\ \beta, & x = 0 \\ 1, & x > 1 \end{cases} \quad (13)$$

$T^+_{u,i}$ is the set of tags which the user u adds on the item i while $T^-_{u,i}$ is the set of tags which the user u does not add on the item i. The overall optimization task with respect to the ranking statistic AUC and the observed data is then:

$$\arg\max_{\hat{\theta}} \sum_{(u,i)\in P_s} AUC(\hat{\theta}, u, i) \quad (14)$$

Then, we use the continuous sigmoid function to replace H:

$$s(x) = \frac{1}{1+e^{-x}} \quad (15)$$

Then using gradient descent, AUC has to be differentiated with respect to all model parameters and for each post $(u,i) \in P_s$ the model parameters $P(u_k|u)$ are updated and renormalized.

$$\begin{aligned} &\frac{\partial}{\partial p_{u_k,u}} AUC(\hat{\theta}, u, i) \\ &= \frac{\partial}{\partial p_{u_k,u}} \frac{1}{|T^+_{u,i}||T^-_{u,i}|} \sum_{t^+\in T^+_{u,i}} \sum_{t^-\in T^-_{u,i}} s(y_{u,i,t^+} - y_{u,i,t^-}) \\ &= z \sum_{t^+\in T^+_{u,i}} \sum_{t^-\in T^-_{u,i}} w_{t^+,t^-} \frac{\partial}{\partial p_{u_k,u}} (y_{u,i,t^+} - y_{u,i,t^-}) \end{aligned}$$

with:

$$\begin{aligned} w_{t^+,t^-} &= s(y_{u,i,t^+} - y_{u,i,t^-})(1 - s(y_{u,i,t^+} - y_{u,i,t^-})) \\ z &= |T^+_{u,i}||T^-_{u,i}| \\ y_{u,i,t^+} - y_{u,i,t^-} &= (\sum_{w\in W_i} \log \sum_{u_k} P(w|t^+, u_k) P(u_k|u) \\ &\quad + \log \sum_{u_k} P(t^+|u_k) P(u_k|u)) \\ &\quad - (\sum_{w\in W_i} \log \sum_{u_k} P(w|t^-, u_k) P(u_k|u) \\ &\quad + \log \sum_{u_k} P(t^-|u_k) P(u_k|u)) \end{aligned}$$

And

$$\begin{aligned} &\frac{\partial}{\partial p_{u_k,u}} (y_{u,i,t^+} - y_{u,i,t^-}) = \\ &\sum_{w\in W_i} \frac{P(w|t^+, u_k)}{\sum_{u_k} P(t^+|u_k) P(u_k|u))} + \frac{P(t^+|u_k)}{\sum_{u_k} P(t^+|u_k) P(u_k|u)} \\ &- \sum_{w\in W_i} \frac{P(w|t^-, u_k)}{\sum_{u_k} P(t^-|u_k) P(u_k|u))} - \frac{P(t^-|u_k)}{\sum_{u_k} P(t^-|u_k) P(u_k|u)} \end{aligned}$$

Then, the derivation of $p_{u_k,u}$ is

$$\frac{\partial AUC}{\partial p_{u_k,u}} = z \sum_{t^+\in T^+_{u,i}} \sum_{t^-\in T^-_{u,i}} w_{t^+,t^-} Y_{t^+,t^-}$$

Table 1: Offline Statistics

Training Data		Test Data	
Total Posts	262,336	Total Posts	668
Total Records	914,162	Total Records	2,307
Total Users	2,677	New/Total Users	2/169
Total Items	234,764	New/Total Items	564/668
Total Tags	56,370	New/Total Tags	54/1,224

where

$$\begin{aligned} w_{t^+,t^-} &= s(y_{u,i,t^+} - y_{u,i,t^-})(1 - s(y_{u,i,t^+} - y_{u,i,t^-})) \\ z &= |T^+_{u,i}||T^-_{u,i}| \\ Y_{t^+,t^-} &= \frac{\partial}{\partial p_{u_k,u}} (y_{u,i,t^+} - y_{u,i,t^-}) \end{aligned}$$

Thus, for each post $(u,i) \in P_s$ the model parameters $P(u_k|u)$ are updated as follow.

$$\hat{p}_{u_k,u} \leftarrow \frac{\hat{p}_{u_k,u} + \gamma \cdot \frac{\partial AUC}{\partial p_{u_k,u}}}{\eta}$$

where η is a normalization factor $\eta = \sum_{u_j} (\hat{p}_{u_j,u} + \gamma \cdot \frac{\partial AUC}{\partial p_{u_j,u}})$ and γ is a learn rate.

4.3 Processing New Users

Our model is designed for personalized tag prediction, especially for existing users. However, in the real world, we still may face users who have not been seen by the tagging system previously. A simple method to predict tags for new users is to just use the general model Equation 3.

A better option is that instead of using the general model, we can build a language model for the new user u_{new} from the item i. Given a new user and an item (u_{new}, i), even if we do not know the past information of the user, we can still get some implication from the content of item i. For existing users, a similar language model is extracted from the items which the users tagged previously. Then the language models are used to represent users' interests. For user u_k,

$$W_{u_k} = [n_{u_k,w_0}, n_{u_k,w_1}, ...n_{u_k,w_i}, ...n_{u_k,w_m}]$$

Here, n_{u_k,w_i} means the number of times of user u_k has used the word w_i. Similarly,

$$P(u_k|u_{new}) \propto sim(u_{new}, u_k) = \frac{W_{u_{new}} \cdot W_{u_k}}{|W_{u_{new}}| \times |W_{u_k}|}$$

Where $sim(u_{new}, u_{new}) = 0$, for new users, there will be no ego effect. All the information should be from environmental effects and item content. After normalizing similarity, we can simply set the

$$P(u_k|u_{new}) = \frac{sim(u_{new}, u_k)}{\sum_{u_i} sim(u_{new}, u_i)} \quad (16)$$

We refer to this calculation as user-lan-user similarity. For new users, we cannot use learning algorithm to refine $P(u_k|u_{new})$.

5. TIME-SENSITIVE SAMPLING

5.1 Dataset

In our experiments, we use the bookmark dataset of the ECML PKDD 09 Challenge Workshop[2]. The dataset S includes 2,679

[2]http://www.kde.cs.uni-kassel.de/ws/dc09/

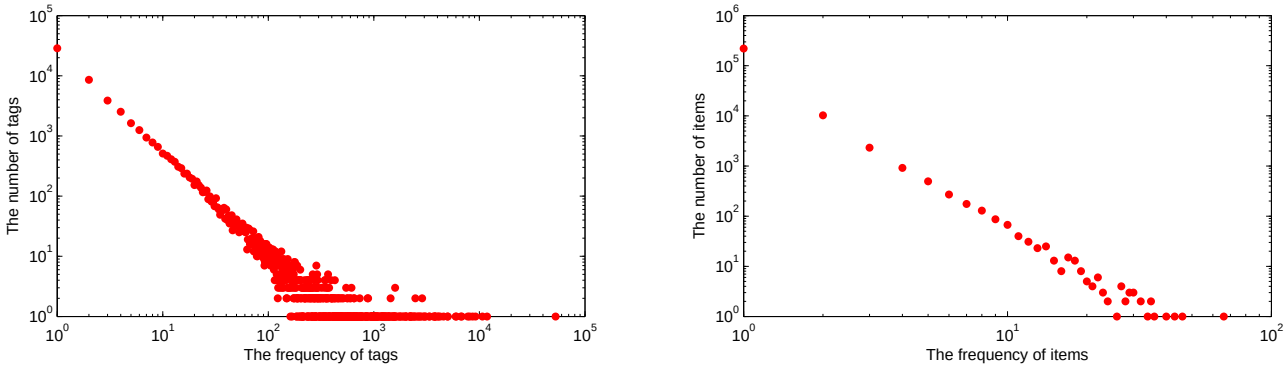


Figure 1: Dataset Statistics

Table 2: Online Statistics

Total Posts	668
Total Records	2307
Old/Total Users	627/668
New/Total Users	41/668
Old/Total Items	66/668
New/Total Items	602/668
Old/Total Tags	1,986/2,307
New/Total Tags	321/2,307

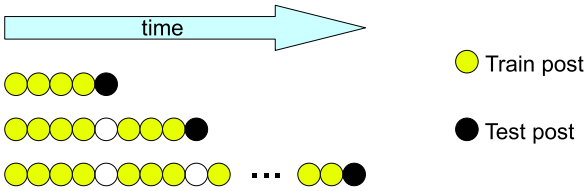


Figure 2: Online framework

users, 263,004 items, 56,424 tags, 262,336 posts and 1,401,104 records. All of the posts contain timestamps. We uniformly sample 668 posts along the time line as our test dataset S_{test} and the remaining posts constitute the training dataset S_{train}.

In Figure 1 we show the tag and item frequencies over the full dataset. In the plot on the right, the large vertical gap between the two leftmost points means that 93.6% items appear only once and only 6.4% of items appear more than once. Thus, most graph-based methods which require more than CORE-2 (users, tags and items appear at least twice) cannot work on it. For tags, 49.4% of tags appear only once; 50.6% of tags appear more than once.

In comparison, if we ignore time information and assume a traditional fixed training and test split (e.g., use the dataset as an "offline" dataset), a test post may have occurred prior to some training posts, effectively using the future to build a model to predict the past. Table 1 provides statistics regarding the training data and the number of "new" items seen in the test data. We find that there are only 2 new users out of 169 users and 54 new tags out of 1,224 tags in the test dataset. However, there are 564 new items out of 668 items even in the offline statistics. Here, "new" means that it does not exist in training data. While the offline analysis can give us some impression of the dataset, it is different from the real world, because in the real world, we cannot use future data as training data to recommend tags for users.

5.2 Online Evaluation

Besides the offline test, another testing method which is often used in tag prediction evaluation is that of fixing a time point—posts whose timestamp is earlier than that time will be used as training data while posts whose timestamps are later than that time will be used as test data. The ECML PKDD Challenge Workshop employed this approach. There are still some problems for this method. For example, if a user never tagged items before that time point and then tagged M posts after that time point, in this test mode, the all M posts of this user will be treated as the posts of a new user to training data. Thus, there will be too many "new user cases" which in the real world is actually existing users. In the real world, after the user tagged his first item, the system should know this user and be able to retrieve the list of tags which this user has previously used. In addition, for users who tagged items both before and after the time point, their interests may not always stay the same and may even change frequently; in the real world, the system can again retrieve the latest tags which can represent the latest interests of this user. Such information should also be considered to make better prediction of tags.

We introduce a better evaluation method which is much closer to the real world and call it the "online" framework in this paper. Figure 2 illustrates the online framework. Like online machine learning [3] which has been used widely, in our online mode[3], the tagging system operates in an incremental mode and the test posts are randomly sampled from the whole dataset along the timeline. In other words, for users and items in our test dataset, we only use the training posts which have earlier timestamps than the test posts, and the available training data is different for each test post. Under this setting, for items tagged early in time, fewer training data is available. The online statistics (shown in Table 2) demonstrate that we still face a new user (a user which is not in the training set) in 6.1% of the cases, and in 90.1% of the time users are trying to bookmark a new item. In addition, there is .139 probability that users would use new tags (which do not appear in the system before). Thus, in the real world, the principal difficulty is to process cases in which

[3]In this paper, online mode means a incremental mode of real tagging system rather than real-time tag prediction.

existing users which try to tag new items. Overall, this online mode is more like a real tag prediction system, permitting the system to learn user behaviors incrementally rather than existing evaluation procedures with a fixed dataset split.

6. EXPERIMENTS

In this section, we describe the details of datasets and experiments. We also compare our approach with two other algorithms.

6.1 Dataset and Evaluation Method

The dataset used in our experiments has been already described in Section 5. All of the posts contain timestamps. We randomly sample 668 posts along the time line as our test dataset S_{test} and the remaining ones are the training dataset S_{train}. We use the common evaluation scheme of F-measure in Top N lists, where $N = 5$ is mainly used as our measurement. The precision, recall and F-measure are calculated as follows.

$$\mathrm{Prec}(S_{test}, N) = \underset{(u,i)\in P_{S_{test}}}{\mathrm{avg}} \frac{|\mathrm{Top}(u,i,N) \cap \{t|(u,i,t) \in S_{test}\}|}{N}$$

$$\mathrm{Rec}(S_{test}, N) = \underset{(u,i)\in P_{S_{test}}}{\mathrm{avg}} \frac{|\mathrm{Top}(u,i,N) \cap \{t|(u,i,t) \in S_{test}\}|}{|\{t|(u,i,t) \in S_{test}\}|}$$

$$\mathrm{F}_1(S_{test}, N) = \frac{2 \times \mathrm{Prec}(S_{test}, N) \times \mathrm{Rec}(S_{test}, N)}{\mathrm{Prec}(S_{test}, N) + \mathrm{Rec}(S_{test}, N)}$$

6.2 Comparison

From our analysis, in the real world, the graph-based method cannot work on most posts. Most graph-based algorithms require that users, tags and items appear at least twice in training set. We compare our approach with Liczak's method [12], which took the first place in the "content-based" recommendation task, and took third place in "graph-based" recommendation task in ECML PKDD Discovery Challenge[4]. They have two versions respectively for the "content-based" task and the "graph-based" task. In this paper, we call their "content-based" version *LHKM-C* and their "graph-based" version *LHKM-G*, corresponding to the authors' initials of [12]. For LHKM-C and LHKM-G, we use the same parameters as they used in the Challenge Workshop. For our model, we only use the most similar 30 neighbors for each test user. In the $P(u_k|u)$ part, we use user-tag-user similarity mode to estimate ego-centric effect and environmental effects for existing users and user-lan-user similarity mode for new users.

In Figure 3(a) and 3(b), we show the comparison between online and offline tests. For each we also show the difference between performances whens recommending various number of tags (known as Top N). We see that as expected, the results of the offline test are always better than the results of the online test, because in the offline test, more training data (even future data) can be used. The results of LHKM-G are slightly better than the results of the LHKM-C, because in LHKM-G, "graph-information" is used. Our method outperforms both LHKM-C and LHKM-G. In offline test, the F-measure of our model is around 11% higher than LHKM-G and 14% higher than LHKM-C. In online test, the F-measure of our model is 12% higher than LHKM-C and LHKM-G. In the following experiments, all the evaluation of F-measure in Top N lists will be based on $N = 5$.

6.3 Optimization Analysis

In this section, we use gradient descent to optimize parameters which can more accurately represent the environmental effects and ego-centric effects. We run the learning algorithm on the offline test. In our optimization, although it shows some improvement on the results, it is very time-consuming. For each user, we only use 10 training items to optimize the environmental effects of 30 neighbors and the ego-centric effect—α. The learning rate is set to 1.

There are two versions: the first is opt-Alpha which only tries to optimize α, the second is opt-Alpha+30N which tries to optimize α and all 30 neighbors, that means, total 31 parameters will be optimized. The initial values of $P(u_k|u)$ are the same as the section 6.2, using user-tag-user similarity for old users and user-lan-user similarity for new users. Figure 3(c) shows the results of the iterative learning algorithm. The x-axis is the number of iterations and y-axis is F-measure. As expected, both optimization methods can improve the results of initial value a little (2-3%) and opt-Alpha+30N always outperforms opt-Alpha. This is because in opt-Alpha+30N, 31 parameters can be optimized while in opt-Alpha, only Alpha is optimized. From Figure 3(c), we also notice that after 1 or 2 times iteration, both opt-Alpha+30N and opt-Alpha get the best results and then the F-measure decreases slightly and converges. We hypothesize that this situation may be caused by overfitting. Another possible reason is that the learning procedure is time consuming, and we only use 10 items to optimize the parameters. Some users tagged thousands of items, so 10 training items may not be sufficient. In addition, better objective function and optimization methods are necessary for further improvement on both F-measure and running time.

6.4 Parameter Analysis

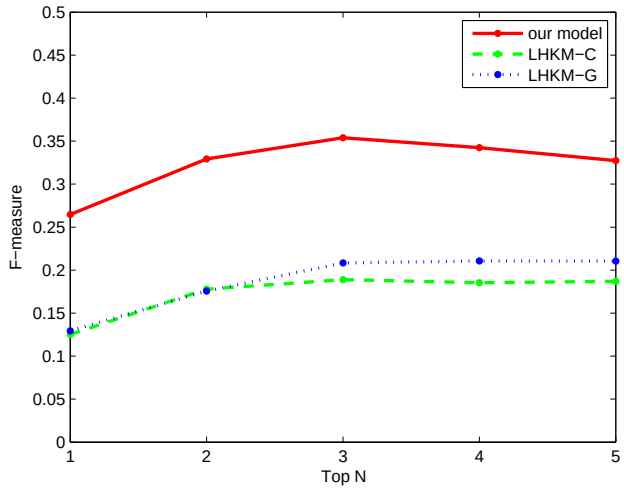
Compared to individual neighbors, the user's ego weight α should be the most important part. It decides the ego weight and relative impact of environmental effects. We consider that usually user's ego weight should be very high. Also the number of neighbors may affect the results of our model.

We find that the optimization process always generates higher α. In this experiment, we fix the number of neighbors to 100 and tune the ego weight alpha, from 0 to 1. The weights of neighbors will be normalized as follow.

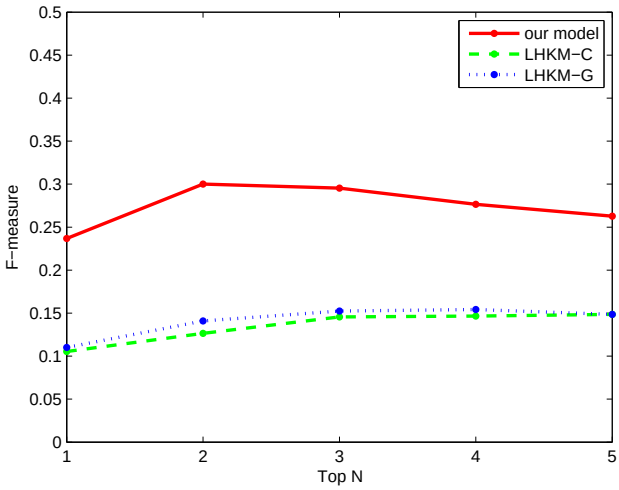
$$p_{u,u_j} \leftarrow (1-\alpha) \cdot \frac{p_{u,u_j}}{\sum_{u_j \neq u} p_{u,u_j}} \tag{17}$$

6.4.1 *Ego-centric effect analysis*

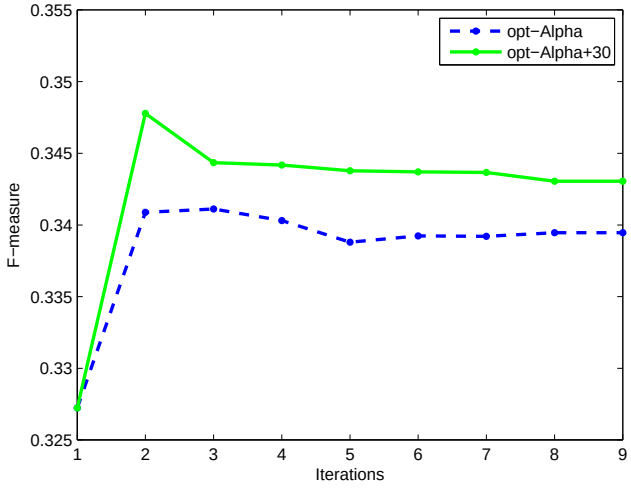
For $P(u_k|u)$, we use user-tag-user similarity for existing users and user-lan-user similarity for new users. We use the most similar 100 users as environmental effects. Figure 3(d) shows the results. In this figure, the straight lines are from LHKM-C and LHKM-G for comparison. (In the online test, the F-measure of LHKM-C and LHKM-G is quite similar, so they only show a single line in the figure.) Our results on the offline tests and on the online tests are highly consistent. When $\alpha = 0$, that means, all information is from the most 100 similar neighbors, the F-measure is still slightly better the LHKM-C and LHKM-G on online test, but slightly worse on offline test. When α is set to 0.05, F-measure dramatically increases, and become higher than that of LHKM-C and LHKM-G in offline test. As α increases, the F-measure increases and achieves the best result when the α is set to 0.7. In offline test, it is around 37.8% (16% higher than LHKM-G) and in online test, it is 27.3% (12% higher than LHKM-G). Another interesting point is that even if α is set to 1, the performance of our model is still much better than LHKM-C and LHKM-G. In online test, regardless of how α is set, our model always outperforms Liczak's methods. These results verify our conjecture that the users' ego weight should be very important in tag prediction.



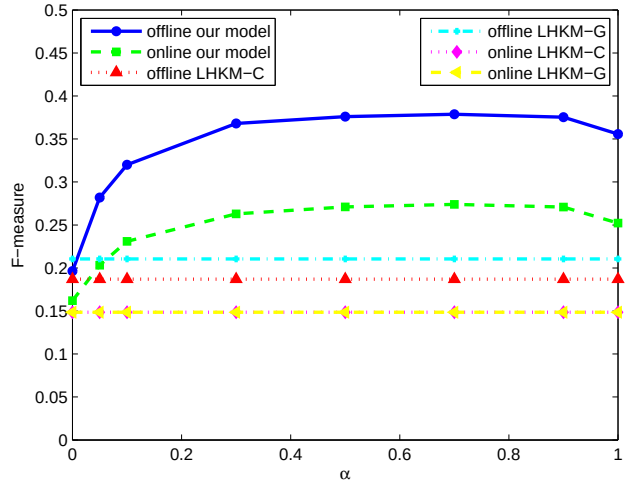
(a) Comparison with offline



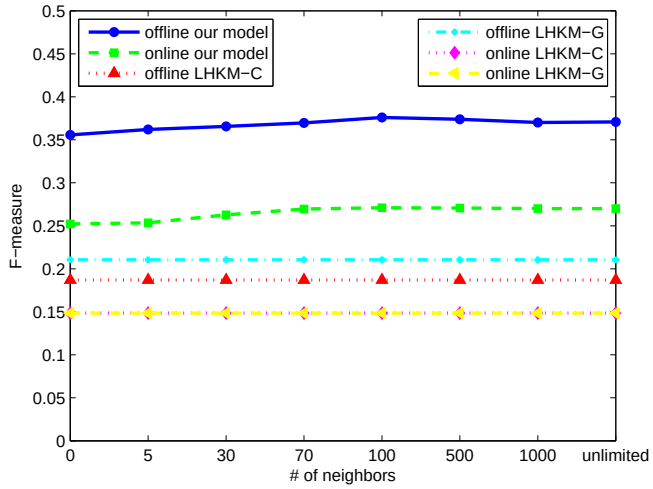
(b) Comparison with online



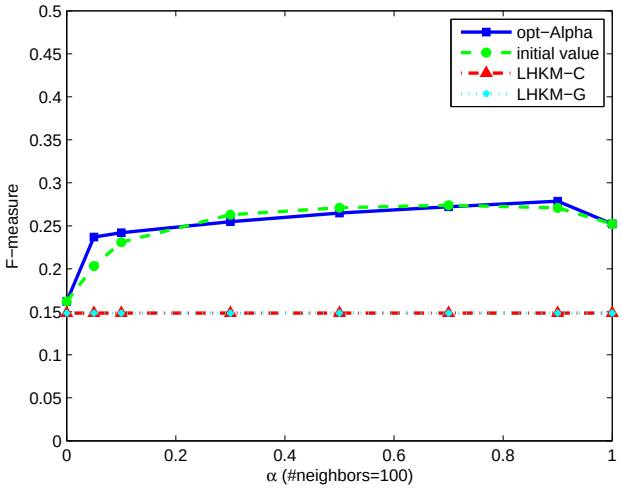
(c) Results on iterative optimization



(d) Ego-centric effect analysis



(e) Environmental effects analysis



(f) Online Experiments

Figure 3: Experimental Results

6.4.2 *Environmental effects analysis*

Then, we fix $\alpha = 0.5$, and tune the number of effective neighbors from 0 to unlimited—that is, we use all possible users and in our model, for existing users, as long as user-tag-user similarity is non-zero, then this user will be treated as an effective neighbor. The results are showed in Figure 3(e). The straight lines are also from LHKM-C and LHKM-G. From Figure 3(e), in the beginning, as the number of neighbors increase, the F-measure increase. When the number of neighbors is set to 100, our model achieves the best F-measure on both offline test and online test, which are 37.5% and 27.1% respectively and also much better than LHKM-C and LHKM-G. We also notice that compared to α, the number of neighbors affects the results less. Thus, the number of neighbors is less important than the ego weight α and it can be simply set to 100 to get the best performance.

6.4.3 *Online experiment*

Based on the manually tuned α, we also try to optimize the ego weight to get the highest F-measure on the online test for real world performance. In this case, the manually tuned α and $P(u_k|u)$ will be used as initial values for the learning algorithm. For each test user, we still use 10 training items to optimize the ego weight. The learning rate is 1. The results are showed in Figure 3(f) opt-Alpha is the optimization version while the "initial value" is the same as the one in 6.4.1. The straight lines are also from LHKM-C and LHKM-G. From Figure 3(f) we can see that at some points, e.g., $\alpha = 0.05$, 0.1 and 0.9, opt-Alpha improves the F-measure and there are also some points where the performance of opt-Alpha and initial value are similar. Here, we also get the highest F-measure 27.9% on the online test which is 13% higher than LHKM-C (an improvement of more than 85%). Comparing to the results of learning algorithm, the results of manually tuned α are good enough and it runs much faster. At this moment, we suggest to manually tune α.

6.5 Five-Fold Cross Validation

Because our test set is relatively small, in order to show the robustness of our model, k-fold cross validation was used to compare the performance of our model vs. LHKM-C and LHKM-G. In k-fold cross-validation, the original sample is randomly partitioned into k subsamples. Of the k subsamples, a single subsample is retained for testing the model and the remaining $k-1$ subsamples are used as training data. The cross-validation process is applied a total of k times (the folds), with each of the k subsamples used exactly once as the test data. In our experiment, $k = 5$ and we do offline testing. The number of neighbors is set to unlimited and alpha is set to 0.5. The parameters of LHKM-G and LHKM-C are the same as previous experiments. For each part of test and training data, the training data contains around 210,000 posts, 2,400 users, 190,000 items and 50,000 tags. and test data contains around 52,000 posts, 1,600 users, 50,000 items and 24,000 tags. Among them there are around 300 new users, 45,000 new items and 63,00 new tags. This is also consistent with our small test set.

In Table 3, we can see that our model outperforms the LHKM-C and LHKM-G by more than 10% on F-measure. The 5 results are quite similar and this also demonstrates that our model can generate better results than LHKM-C and LHKM-G stably.

Table 3: 5-Fold Cross Validation

	LHKM-C	LHKM-G	our model
Test 1	0.193	0.202	0.348
Test 2	0.194	0.213	0.348
Test 3	0.193	0.210	0.347
Test 4	0.194	0.211	0.347
Test 5	0.195	0.211	0.348
mean	0.1938	0.2094	0.3476

7. CONCLUSIONS AND FUTURE WORK

In this paper, we suggest that social tagging by nature is an incremental process, and perform a time-sensitive sampling on an existing public dataset. Our analysis shows that in the real world, the problem of tag prediction is dominated by the need to predict tags for existing users when they tag new items. Most graph-based methods require CORE p, and thus may simply not function in such situations.

We proposed a novel probabilistic model for personalized tag prediction. Our online experiments and 5-fold cross validation experiments indicate that our model achieves over 30% improvement on F-measure compared to a leading method, in the "real-world" test scenario.

Although manually tuned parameters can achieve a high performance, all the users share the same ego weight. We believe that different users should have different user profiles—personalized weights of ego-centric effect and environmental effects. In the future, a probabilistic analysis on the effects of neighboring users may be needed to make further improvements.

Acknowledgments

This work was supported in part by a grant from the National Science Foundation under award IIS-0545875 and an equipment grant from Sun Microsystems. We also appreciate Xiong Xiong and Xiaoguang Qi's help on the model and experiments and Lifeng Shang's helpful discussions on the model.

8. REFERENCES

[1] S. Bao, G. Xue, X. Wu, Y. Yu, B. Fei, and Z. Su. Optimizing web search using social annotations. In *Proceedings of the 16th International Conference on World Wide Web*, pages 501–510. ACM Press, 2007.

[2] C. Biancalana, A. Micarelli, and C. Squarcella. Nereau: a social approach to query expansion. In *Proceedings of the 10th ACM Workshop on Web Information and Data Management (WIDM)*, pages 95–102, 2008.

[3] A. Blum. On-line algorithms in machine learning. In *Developments from a June 1996 seminar on Online algorithms*, pages 306–325, London, UK, 1998. Springer-Verlag.

[4] F. Eisterlehner, A. Hotho, and R. Jäschke, editors. *ECML/PKDD Discovery Challenge Workshop (DC09)*, volume 497 of *CEUR Workshop Proceedings*, Sept. 2009.

[5] N. Garg and I. Weber. Personalized, interactive tag recommendation for flickr. In *Proceedings of the 2nd ACM Conference on Recommender Systems (RecSys)*, pages 67–74. ACM Press, 2008.

[6] Z. Guan, J. Bu, Q. Mei, C. Chen, and C. Wang. Personalized tag recommendation using graph-based ranking on multi-type interrelated objects. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 540–547. ACM Press, 2009.

[7] P. Heymann, D. Ramage, and H. Garcia-Molina. Social tag prediction. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 531–538. ACM Press, 2008.

[8] A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. Information retrieval in folksonomies: Search and ranking. In *The Semantic Web: Research and Applications*, Lecture Notes in Computer Science, chapter 31, pages 411–426. Springer, 2006.

[9] R. Jäschke, L. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme. Tag recommendations in folksonomies. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 506–514, Berlin, Heidelberg, 2007. Springer-Verlag.

[10] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 426–434. ACM Press, 2008.

[11] Y. Koren. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 447–456. ACM Press, 2009.

[12] M. Lipczak, Y. Hu, Y. Kollet, and E. Milios. Tag sources for recommendation in collaborative tagging systems. In *ECML/PKDD Discovery Challenge Workshop (DC09)*, 2009.

[13] B. Marlin. Modeling user rating profiles for collaborative filtering. In *Advances in Neural Information Processing Systems*, volume 16, pages 627–634. MIT Press, 2003.

[14] D. Ramage, P. Heymann, C. D. Manning, and H. Garcia-Molina. Clustering the tagged web. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining (WSDM)*, pages 54–63. ACM Press, 2009.

[15] S. Rendle, L. Balby Marinho, A. Nanopoulos, and L. Schmidt-Thieme. Learning optimal ranking with tensor factorization for tag recommendation. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 727–736. ACM Press, 2009.

[16] S. Rendle and L. Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of the 3rd International Conference on Web Search and Web Data Mining (WSDM)*, pages 81–90. ACM Press, 2010.

[17] K.-U. Schmidt, T. Sarnow, and L. Stojanovic. Socially filtered web search: an approach using social bookmarking tags to personalize web search. In *Proceedings of the 24th ACM Symposium on Applied Computing (SAC)*, pages 670–674. ACM Press, 2009.

[18] Y. Song, L. Zhang, and C. L. Giles. A sparse gaussian processes classification framework for fast tag suggestions. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM)*, pages 93–102. ACM Press, 2008.

[19] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos. Tag recommendations based on tensor dimensionality reduction. In *Proceedings of the 2nd ACM International Conference on Recommender Systems (RecSys)*, pages 43–50. ACM Press, 2008.

[20] University of Kassel, Germany. Bibsonomy. `http://www.bibsonomy.org/`, 2010.

[21] S. Xu, S. Bao, B. Fei, Z. Su, and Y. Yu. Exploring folksonomy for personalized search. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 155–162. ACM Press, 2008.

[22] Yahoo! Delicious home page. `http://delicious.com/`, 2010.

[23] Z. Yin, R. Li, Q. Mei, and J. Han. Exploring social tagging graph for web object classification. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 957–966. ACM Press, 2009.

[24] D. Zhang, R. Mao, and W. Li. The recurrence dynamics of social tagging. In *Proceedings of the 18th International Conference on World Wide Web*, pages 1205–1206. ACM Press, 2009.