Keywords: Social bookmarking, Recommendation system, Collaborative filtering.

Abstract: Web-based bookmark management services called social bookmarking has been in the spotlight recently. Social bookmarking allows users to add several keywords called tags to items they bookmarked. Many previous works on social bookmarking using actual words for tags, called folksonomy, have come out. However, essential information of tags is not represented in their tag names, but in the classification of items by tags. Based on this assumption, we propose an anti-folksonomical recommendation system for calculating similarities between groups of items classified according to tags. In addition, we use hypothesis testing to improve these similarities based on statistical reliability. The experimental results show that our proposed system provides an appropriate recommendation result even if users tagged with different keywords.

1 INTRODUCTION

Recommendation systems have been widely researched (Li and Zaiane, 2004) (Kazienko and Kiewra, 2004) (Ishikawa et al., 2002) (Gunduz and Ozsu, 2003). Most of them are based on ‘collaborative filtering’, which is a method for predicting a specific user’s preferences for new items using preferences obtained from many other users (Goldberg et al., 2003) (Resnick et al., 1994) (Sarwar et al., 2001).

Generally speaking, collaborative filtering is defined as a method for estimating preferences for items that users have not yet found by comparing preferences of items that they have already browsed, not by using item context. We use the term ‘collaborative filtering’ in a narrow sense as a method for calculating preferences as the rating of each unknown item against preferences of previously viewed items. The algorithm is as follows:

1. Collect preferences of viewed items.
2. Calculate similarities between the focus user and others based on their preferences of commonly viewed items.
3. Calculate preference of each unknown item based on similarities between the focus user and others who have already viewed it.

As Sarwar et al. acutely pointed out, more items than users leads to poor recommendation results because of the sparsity of preference data (Sarwar et al., 2001).

As other recommendation systems, novel web services called social bookmarking (SBM) have appeared in recent years. SBM allows users to annotate
each item with one or more keywords called ‘tags’.

Niwa et al. investigated the recommendation system by using tags created by SBM users (Niwa et al., 2006). They aggregated similar tags to reduce word redundancy and made tag clusters with keywords having the same meaning.

However, a keyword may have various meanings depending on the context. Golder et al. pointed out these types of tags as being polysemous (Golder and Huberman, 2005). For example, ‘apple’ has multiple meanings: a sweet red fruit or the consumer electronics company. If the recommendation system includes these types of tags in only one tag cluster, it cannot recommend items to users who use the tag in other contexts. Due to this problem, a vocabulary-based recommendation system may lead to inappropriate results.

Therefore, we did not focus on the vocabulary of tags and instead propose tag-based collaborative filtering. In addition, we define the novel similarity between item clusters based on hypothesis testing.

The rest of this paper is organized as follows. In section 2, we introduce a related study of ours. Next, we explain our recommendation system algorithm in section 3. In section 4, we evaluate and discuss our recommendation system, and we conclude in section 5.

2 RELATED STUDY

In this section, we introduce a related study of conventional recommendation systems using an SBM service and based on the co-occurrence of items.

2.1 Social Bookmarking (SBM)

2.1.1 What is SBM?

SBM services enable users to store, organize, search, manage, and share bookmarks of web pages on the Internet. SBM has a specific feature called ‘tags’. Tags are keywords created by each SBM user for categorization of web pages. In SBM services, users can share their bookmarks with other users and also browse other users’ bookmarks. For example, users can browse a list of SBM users who bookmarked the same item as they did and also browse a list of items that are tagged with the same keyword.

2.1.2 Folksonomy

‘Folksonomy’, from ‘folks’ and ‘taxonomy’, is a method of collaboratively managing tags to catego-
important point to note is that the latter situation causes word redundancy.

Let us look more closely at the bottom-ranked words. We can find unusual tags like ‘toread’ (sticky note for future reference. Golder et al. also pointed it out as ‘Task Organizing’ (Golder and Huberman, 2005)), ‘web/app’ (users’ own directory hierarchy), and so on. That is, in an actual SBM service, most users’ tags are not for others’ convenience but for themselves to manage their own bookmarks for their future use.

Let us now return to Niwa et al.’s study. The recommendation system of using tag names is implicitly based on the collaboration of SBM users. The system then discards the bottom-ranked words, but retains a large percentage of the total tags in the practical SBM service.

We will discuss SBM tagging in detail. An example of the relationship between items and keywords is shown in Fig. 2. The word redundancy is clear. Users use the words ‘robots’ and ‘AI’ for tagging the same concept of ‘machines that can think’. In addition, there is another word redundancy. Users use the same word, ‘robot’, for tagging two different concepts, ‘machines that can think’ and ‘automated systems’. A recommendation system using only the co-occurrence of words would never recommend items in these cases.

In contrast, our proposed system works even if users tag with different keywords because it does not pay attention to the vocabulary of the tags. Instead, it is rather similar to systems based on co-occurrence of items.

2.3 Conventional System based on Co-occurrence of Items

Rucker and Polanco developed a system for recommending items by calculating similarities between folders (categories by user) of bookmarks (Rucker and Polanco, 1997). Their system is similar to our proposed system from the viewpoint of only using sets of items. However, their system does not rank each recommended item.

A comparison between our proposed system and conventional systems discussed previously is shown in Table 1. As can be seen, our system calculates similarities between item clusters by using hypothesis testing for finding similar ones. Furthermore, our system calculates the recommendation rate; therefore, it can rank each item.

3 PROPOSED SYSTEM

3.1 Recommendation based on SBM by using “Item Cluster”

In this paper, we focus on ‘item clusters’, which are sets of items classified by the tags used by each user. Each user has the same number of item clusters as the number of tags he/she uses in the SBM service. When a user issues a query by selecting a tag from his/her tag records, the system searches for items to recommend by focusing on the similarities between ‘query item cluster’ corresponding to the query and ‘recommender item clusters’ corresponding to other tags in the scope of commonly bookmarked items.

3.2 Model of Item Cluster

We focused on a particular tag $t_{query}$ that is tagged by user $u_{focused}$. $B_i$ refers to all the items bookmarked by $u_{focused}$, and $T_i$ refers to all the items tagged $t_{query}$ by $u_{focused}$. All of the items $A$ (bookmarked by all users in the SBM service) can be classified into three sets exclusively, as shown in Fig. 3.

1. Bookmarked by $u_{focused}$, and tagged with $t_{query}$ ($T_{r}$)
2. Bookmarked by $u_{focused}$, but tagged without $t_{query}$ ($B_i \cap \overline{T_r}$)
3. Not bookmarked by $u_{focused}$ ($\overline{T_r}$)

We define a set of items tagged with a certain tag name, like $T_k$ as an ‘item cluster’. Let us consider two item clusters, ‘query item cluster $T_{r}$’ and ‘recommender item cluster $T_k$’. $T_r$ is an item cluster tagged with $t_{query}$ by $u_{focused}$, and $T_k$ is tagged with $t_l$ by $u_l$ (note that $u_l$ is not $u_{focused}$). We studied the conceptual similarity between $T_r$ and $T_k$. Here, $k$ is the number of items that are included in both $T_r$ and $T_k$. $m_k$ is the number of items in $T_k$ that $u_l$ tags with a different
Table 1: Comparison between proposed method and conventional methods.

<table>
<thead>
<tr>
<th></th>
<th>Collaborative Filtering</th>
<th>Conventional (Niwa et al., 2006)</th>
<th>Conventional (Rucker and Polanco, 1997)</th>
<th>Proposed Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Focused on</td>
<td>user and category</td>
<td>tag (records)</td>
<td>user and tag item</td>
<td>user and tag item</td>
</tr>
<tr>
<td>Co-occurrence</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Rating</td>
<td>co-occurrence ratio</td>
<td>co-occurrence ratio</td>
<td>not defined</td>
<td>likelihood ratio</td>
</tr>
</tbody>
</table>

Figure 3: SBM modeling regarding relationship among items, users, and tags.

Figure 4: Item recommendation by comparing item clusters.

tag name from $t_j$: $m_o$ is the number of items in $T_o$ that $u_{focused}$ tags with a different tag name from $t_query$. Let $m = m_s + m_o$ and $n = m + k$. The relationship between $k$ and $n$ shows a conceptual similarity of the two item clusters. $n$ and $k$ are shown in Fig. 4 and described as

$$n = |B_s \cap B_o \cap (T_s \cup T_o)|$$  \hspace{1cm} (1)

$$k = |T_s \cap T_o|.$$  \hspace{1cm} (2)

Next, we look at the expected similarity of the two item clusters. Here, we assume that there are only two relationships between item clusters — similar viewpoint and different viewpoint. If two users tag items from similar viewpoints, the expected probability that both users tag the same item is assumed to be $p_1$. Otherwise, the expected probability is assumed to be $p_0$. Here, $p_1 > p_0$. $p_1$ and $p_0$ can be estimated by observing all item clusters to separate them into similar and different viewpoints. Desirable items should be recommended from similar item clusters. The conceptual similarity between $T_s$ and $T_o$ is defined by

$$\text{sim}(T_s, T_o) = \log \frac{L(n, k, p_1)}{L(n, k, p_0)} = k \log \frac{p_1}{p_0} + (n - k) \log \frac{1 - p_1}{1 - p_0},$$  \hspace{1cm} (3)

where

$$L(n, k, p) = n C_k p^k (1 - p)^{n - k}.$$  \hspace{1cm} (4)

The log likelihood ratio of whether the similarity of two clusters is likely to be $p_1$ or $p_0$ is shown in Eq. 3. Here we assume the relationship between $k$ and $n$ follows a binomial distribution (Eq. 4) with parameters $p_1$ and $p_0$ for similar and different viewpoints, respectively.

Finally, we define the recommendation rate of each item by using similarities between the item clusters. The system selects one item as the candidate to be recommended from $B_s$ ($i \in B_s$). We define $i$’s recommendation rate by calculating the sum of similarities between the query item cluster and each recommender item cluster that contains $i$.

$$R(T_s, i) = \sum_{T_o \in T_{all}} \chi(T_o, i) \text{sim}(T_s, T_o)$$  \hspace{1cm} (5)

$$\chi(T_o, i) = \begin{cases} 1 & \text{if } i \in T_o \cap \text{sim}(T_s, T_o) \geq 0 \\ 0 & \text{otherwise} \end{cases}$$  \hspace{1cm} (6)

where $T_{all}$ is the set of all item clusters.

3.3 Procedure of the Proposed System

The recommendation algorithm is as follows: $u_{focused}$ issues a query by selecting tag name ‘$t_query$’ from his/her tag records. We define it as $T_s$.

1. Calculate each $T_o \cap \text{sim}(T_s, T_o)$ as Eq. 3.
2. Calculate each $\chi(T_o, i)$ as Eq. 6.
3. Calculate each $i$ of $R(T_s, i)$ as Eq. 5.
4. Sort items according to recommendation rate and recommend $i$ whose $R(T_s, i)$ is top $th_{rec}$.

Here, $th_{rec}$ is the number of items to be recommended. Fig. 5 is an example of the procedure. $T_s$, $B_s$ and $T_o$ are item clusters. Now we consider a recommendation for $T_o$. That is, we define $T_o$ as query item cluster.
(T_i) and the others as recommender item clusters (T_o). i_1 and i_2 are the items to be recommended. That is, the user who makes T_o has not bookmark these items.

First, the system calculate similarity between query item cluster T_q and each recommender item clusters T_o, T_r. Next, it checks every χ. For example, i_1 is included in T_o so χ(T_o, i_1) = 1. On the other hand, T_r does not include i_1 so χ(T_r, i_1) = 0. Finally, it calculates each recommendation rate R(T_o, i_1), R(T_o, i_2) based on sum of products which are computed by multiplying similarity by χ. Thus, 

\[ R(T_o, i_1) = \text{sim}(T_o, T_b)\chi(T_o, i_1) + \text{sim}(T_o, T_c)\chi(T_o, i_1). \]

4 EXPERIMENTS

We performed three experiments using live data obtained from del.icio.us, which is one of the most famous SBM service sites. In these experiments, we set p_0 = 0.1, p_1 = 0.6.

4.1 Performance Evaluation

We randomly collected data of 1,000 SBM users in August, 2006. They had bookmarked about 310,000 unique items (URLs) and had tagged items with about 260 keywords on average. Therefore, we collected about 260,000 sets of item clusters. We used all of these item clusters for the experiments.

4.1.1 Evaluation Method

We used the collected data for calculating similarities and for evaluating our system. We masked tag information, that is, we hid whether all items included in B_i were tagged or not and revealed them after a recommendation was made.

The evaluation method was as follows.

1. Select T_q from the collected data. We defined the items included in T_q as correct class X.

2. Calculate recommendation rate of each item corresponding to B_i, and recommend items from top to t_{th}rec-th. We defined these items as recommended class R.

3. Count the number of items X, R, and R \cap X, then calculate the recall and precision.

Recall and precision were defined as follows.

\[ \text{recall} = \frac{R \cap X}{X} \quad \text{(7)} \]
\[ \text{precision} = \frac{R \cap X}{R} \quad \text{(8)} \]

In addition, we used F-measure, defined as follows.

\[ \text{F-measure} = \frac{2 \cdot \text{recall} \cdot \text{precision}}{\text{recall} + \text{precision}} \quad \text{(9)} \]

In this evaluation, we focused on the top 100 query item clusters, which were ranked by the number of items in T_q. The biggest number of B_i was 17,960, and the smallest was 865. The average was 6,758.4. The biggest number of T_q in the query item clusters was 1,991, and the smallest was 477. The average was 729.28.

Note that we omitted isolated items that had been tagged only by T_q to evaluate net performance.

4.1.2 Experimental Results

The results of the evaluation are shown in Figs. 6 and 7. The averages of recall, precision, and F-measure for each query cluster are shown in Fig. 6, and the relationship between recall and precision for item clusters 1 to 8 (Table 2) is shown in Fig. 7.
4.1.3 Discussion

In Fig. 6, we can see the precision was 0.78 when our proposed system recommended the top 100 items ranked by their similarities. The precision decreased to 0.67 for 200 items. The F-measure, which is an important measure for recommendation, peaked at about the top 400. Judging from the above, we may say that our proposed system is useful for at least the top 400 items, which is enough for a recommendation system.

The coefficient of correlation between the number of items in the item clusters and the maximum of the F-measure is -0.34. This fact shows the robustness of our proposed system. Its robustness is also evident in Fig. 7, especially in clusters 1 to 3 and clusters 6 to 8. These clusters gave good results in spite of the various numbers of items.

In some cases, we found cases of low precision, such as cluster 4, and cases of low recall, such as cluster 5. We can say with fair certainty that one of the reasons was a lack of data. We could only gather data from 1,000 people, which is less than 1% of SBM users.

We found another reason by looking at the data. The name of the tag in cluster 4 was 'javascript', however, most of the recommender item clusters were ‘programming’. The scope of the recommender item clusters seemed to be broader than that of cluster 4. However, lack of data makes the similarities between cluster 4 and these recommender item clusters relatively high. The precision decreased because cluster 4 frequently recommended items out of the focus. The name of the tag in cluster 5 was ‘art’, and most of the recommender item cluster was ‘webdesign’. The scope of the recommender item cluster seemed to be narrower than that of cluster 5. Therefore, cluster 5 was recommended in only a part of items recommended and recall was decreased.

These problems were caused by the lack of data, but there is further room for investigation. For example, we will combine item clusters to create more suitable recommender item clusters. However, a more comprehensive study on creating data lies outside the scope of this paper.

4.2 Comparison 1: Recommendation based on Folksonomy

In this section, we compare the recommendation systems based on folksonomy with our proposed system from the viewpoint of recall and precision. We show the difference between the two methods in Table 3 (Comparison 1 vs Proposed).

4.2.1 Evaluation

The comparative recommendation system based on folksonomy is as follows.

1. User inputs a tag name into the system as a query.
2. System recommends items with such a tag in descending order.

4.2.2 Experimental Results

We show an example of the comparative results in Fig. 8. This result corresponds to the item cluster which \( t_{\text{query}} \) is ajax, \( B \) is 2118 and \( T \) is 84. Our method clearly outperformed the system based on folksonomy.
4.2.3 Discussion

One can safely state that the recommendation system based on item clusters can produce better results than the recommendation system based on folksonomy. One might also think that folksonomy would lead to better results than our system when the query word is used commonly. However, these results show that this may not be true. Note that 'ajax', which refers to javascript programming techniques, is a well-known word among web programmers.

Let us look closely at the results to find why our system is more appropriate than the comparative one. In a query item cluster, the items tagged with 'ajax' show us high quality interfaces or programming techniques. On the other hand, in a recommender item cluster, the tagged items show us only an implementation of ajax. That is, these item clusters are based on different opinions even though the tags are the same. Moreover, recommendation system based on folksonomy cannot recommend items to users who use singular tag names such as 'java/app' or '***java***'.

Our proposed system, however, is not limited by the tag name.

4.3 Comparison 2: Similarity by Jaccard Coefficient

In this section, we compare the similarity by Jaccard coefficient with that based on hypothesis testing. We show the difference between the two methods in Table 3 (Comparison 2 vs Proposed).

4.3.1 Evaluation

There are conventional systems for comparing the similarity of sample sets, such as the Jaccard and cosine coefficients. The Jaccard coefficient is defined as the two sample sets' intersection divided by their union. We can define the Jaccard coefficient for our situation as follows.

\[
\text{sim}_{\text{Jaccard}}(s, o) = \frac{k}{n},
\]

where \( n \) and \( k \) are the values in Eqs. (1) and (2).

Therefore, we assume a comparative system replacing the Jaccard coefficient with similarities based on hypothesis testing. It can be said that the system is a conventional simple collaborative filtering system based on tags.

4.3.2 Results

We show an example of the comparative results in Fig. 8. This result corresponds to the item cluster previously described in 4.2.2. Our system is clearly more appropriate than the system based on the Jaccard coefficient.

4.3.3 Discussion

We explain why our similarity is better than the conventional similarity based on the Jaccard coefficient.
A comparison between hypothesis testing and the Jaccard coefficient is shown in Fig. 9. Lines (1)-(a) and (b) show the same similarities by the Jaccard coefficient. On the other hand, lines (2)-(a) and (b) show the same similarities by hypothesis testing. Line (1) and (2)-(a) shows the value of 0.6 and (1) and (2)-(b) shows the value of 0.4.

Then, it is open to question to equate the case of \( n = 4, k = 3 \) with the case of \( n = 20, k = 15 \). The former case would arise more often than the latter. Therefore, in the Jaccard coefficient, a small value of \( n \) leads to worse results. In other words, we have to avoid any accidental co-occurrence for a high-precision and high-recall recommendation system.

Hypothesis testing shows a small value when \( n \) or \( k \) is small and a large value when both \( n \) and \( k \) are large. Then, Eq. 3 can calculate similarity except in accidental co-occurrences.

5 CONCLUSIONS

We proposed a novel recommendation system using SBM data. Several conventional systems using folksonomy have focused on actual tag names. However, we focused on item clusters, which are sets of items tagged by each SBM user. We assumed SBM users’ behavior follows binomial distribution and used hypothesis testing to calculate the similarities between two item clusters. In addition, we evaluated our recommendation system. The results showed high recall and precision. We compared our proposed system with the systems using actual tag names and showed that our proposed system was more appropriate. We also compared our proposed similarity calculation based on hypothesis testing with a conventional similarity calculation and verified that our resultant similarities were better than the conventional ones.

REFERENCES


