A social inverted index for social-tagging-based information retrieval

Speaker: Po-Hsien Shih
AdVisor: Jia-Ling Koh
Source: Journal of Information Science 2012
Date: 25, February, 2013
Outline

• Introduction
• Social inverted index
• Evaluation
• Conclusion
Introduction

- What’s “**keyword**”?
  - A set of keyword terms => the information we want.
  - The searchers’ minds <=> The information in the collection... So we must have a good query!

- Why we use “**tag**” recently?
  - Tags freely assigned by users to web resources on many social websites.
  - Tags can represent not only keywords but also personal ratings or other forms of comments or metadata.
Introduction

• **Personomy**
  • Each user organized the web resources with his or her own vocabulary, or a set of tags.

• **Folksonomy**
  • **Users** may also assign tags to **other users’** resources. This type of tagging is called **social tagging** or **collaborative tagging**.
  • If users can see other users’ tags, they are highly likely to be socially influenced by one another when they choose their own tags.
  • Both of them have been contributory in improving web search, especially in terms of **indexing**.
Introduction

• **Traditional web search**
  - Index terms are automatically extracted from the text in a document by a search engine, and these terms are then used for matching with query terms. (ex: Google, Bing, etc)

• **Tag-based web search**
  - Tags are chosen directly by humans and can be used as a good substitute for or as a supplement to the index terms in a document. (ex: Flicker, Delicious, etc)
  - The tag-based web search is a new form of web search that exploits tag data for retrieving and ranking web resources.
Introduction

• What’s “inverted index”? 
  • An index that maps each index term to a list of documents containing the index term.
  • In a traditional web search, a document consists of terms that form a binary relationship from a document to terms.

<table>
<thead>
<tr>
<th>Term</th>
<th>Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat</td>
<td>1,2,3,4,5</td>
</tr>
<tr>
<td>dog</td>
<td>2,4,6,7,8</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Introduction

• Why we have to build a new index structure?

• A resource (document) is annotated with tags (terms) by a user, creating a ternary relationship among resource, tag, and user.

• In some situations, the ternary relationship should be preserved to generate a more meaningful value, rather than just a user count.

• To preserve the ternary relationship, a new type of inverted index needs to be designed, which should be different from the traditional term-to-documents inverted index and reflect the user aspect of tagging.
Outline

• Introduction

• **Social inverted index**
  • Basic social inverted index
  • index construction and maintenance
  • Considering the semantics of tag

• Evaluation

• Conclusion
Social inverted index

**Notation:**
R - Resource collection / r - one resource  
T - Tag Index / t - one tag  
U - User List / u - one user  
STS - Social Tagging System

---

R = \{ r_1, r_2, r_3 \}  
T = \{ apple, iphone, mobile \}  
U = \{ Alice, Bob, Tom \}  
STS = \{ (r_1, apple, Alice), (r_1, apple, Bob), (r_1, mobile, Alice),  
       (r_1, mobile, Tom), (r_2, apple, Alice), (r_2, apple, Bob),  
       (r_2, apple, Tom), (r_2, iphone, Alice), (r_2, iphone, Tom),  
       (r_3, apple, Bob), (r_3, apple, Tom), (r_3, iphone, Alice),  
       (r_3, iphone, Tom), (r_3, mobile, Bob) \}
If we require a tag-to-resources inverted index while maintaining user information, two possible combinations of two inverted lists may be available:

Figure 1. Cross-references.

Figure 2. Sublists.
Basic Social inverted index

• Why we choose latter one?
  • It’s easy to implement.
  • Each entry for user sublists may have its own weight.
  • The sublists may be sorted by weight.

• Disadvantage
  • The redundancy of users in the sublists may cause more space cost, but it’s acceptable for the efficiency gains from maintaining separate user sublists.
Basic social inverted index

The spaces for the $w_{t,r}$ values may remain empty and be determined on the fly at the time of query evaluation, depending on the query types. The question mark in each entry means that its value may not be determined, while the exclamation mark means that its value is determined.

$W_{t,R} = f_{t,R}$  -> The total frequency of tag $t$ in the collection
$W_{t,r} = f_{t,r}$   -> The frequency of tag $t$ in resource $r$
$w_{t,r,u} = f_{t,r,u} (=1)$  -> The frequency of tag $t$ in resource $r$ by user $u$
• Extension

• With weighted triplets

• With user label

• With user timestamps

• With various user weight
• **Tag frequency**

• Reflecting the popularity of the tag or the degree of consensus among users about describing or categorizing the resource that is annotated with the tag.

• \( f_{t,r} = |U_{t,r}| \)

• \( f_{t,R} = |R_t| \)

• \( F_{t,R} = \sum_{r \in R_t} f_{t,r} \)
• Tag co-occurrence frequency

  • Reflecting the degree of semantic similarity between tags, and is said to be meaningful because it is human-generated semantics.

  \[
  \text{macrocof}(t_1, t_2) = |R_{t_1} \cap R_{t_2}| = |\{r_2, r_3\}| = 2
  \]

  \[
  \text{microcof}(t_1, t_2) = \sum_{r \in R_{t_1} \cap R_{t_2}} |U_{t_1,r} \cap U_{t_2,r}| = |\{Alice, Tom\}| + |\{Tom\}| = 3
  \]

  • Tag co-occurrences can be also applied to other applications, such as tag recommendations or query expansion.
The main advantages of social inverted index:

- They preserve the resource–tag–user ternary relationship of tagging.
- They can support a wide variety of existing triplet or user-weighting schemes.
- They are flexible enough to support conditional queries that use varying weight values in the lists, depending on the query.
- They facilitate several critical computations, such as calculations of tag frequencies and tag co-occurrence frequencies.
Index Construction
r1 : (r1, apple, Alice), (r1, apple, Bob), (r1, mobile, Alice), (r1, mobile, Tom)
r2 : (r2, apple, Alice), (r2, apple, Bob), (r2, apple, Tom), (r2, iphone, Alice), (r2, iphone, Tom)
r3 : (r3, apple, Bob), (r3, apple, Tom), (r3, iphone, Alice), (r3, iphone, Tom), (r3, mobile, Bob)
Algorithm: BuildBasicSocialInvertedIndex

Input: A set of resources, each with \((r, t, u)\) triplets in a social-tagging system

Output: A set BasicSocialInvertedIndex

< Notation >
Let \(R\), \(T\), and \(U\) be the set of resources, tags, and users respectively.
Let \(TagIndex\) be a set of index tags.
Let \(R_t\) be a set of resources annotated with \(t\), and \(U_{t,r}\) be a set of users who annotate \(r\) with \(t\).
Let \(f_{t,r}\) be a frequency of \(t\) on \(r\), \(f_{t,R}\) be a resource frequency of \(t\), and \(f_{t,r,u}\) be a frequency of \(t\) in \(r\) by \(u\).
Let ResourceList\(_t\) be a list of resources that are annotated with \(t\), each of which is a triplet of \(<\) resource, frequency, address of user sublist \(>\) and UserList\(_{t,r}\) be a list of users who annotate \(r\) with \(t\), each of which is a pair of \(<\) user, frequency \(>\).

< Index Building >
01 – Initialize TagIndex \(\leftarrow \emptyset\).
02 – For each resource \(r^* \in R\),
03 — Read the \((r^*, t, u)\) triplets.
04 — Sort and group the \((r^*, t, u)\) triplets by \(t\).
05 — For each \(t^*\) of \((r^*, t^*, u)\) triplets,
06 —— If \(t^* \not\in TagIndex\), then
07 —— Set TagIndex \(\leftarrow TagIndex \cup \{t^*\}\).
08 ——— Initialize \(R_{t^*} \leftarrow \emptyset\), \(f_{t^*, R} \leftarrow 0\), ResourceList\(_{t^*}\) \(\leftarrow \emptyset\).
09 —— Else, then read ResourceList\(_{t^*}\).
11 —— If \(r^* \not\in R_{t^*}\), then
12 —— Set \(R_{t^*} \leftarrow R_{t^*} \cup \{r^*\}\), \(f_{t^*, R} += 1\).
13 —— Initialize \(U_{t^*, r^*} \leftarrow \emptyset\), \(f_{t^*, r^*} \leftarrow 0\), \(f_{t^*, r^*, u} \leftarrow 0\), UserList\(_{t^*, r^*}\) \(\leftarrow \emptyset\).
14 —— Execute \(Add(\text{ResourceList}_{t^*}, <r^*, f_{t^*, r^*}, \text{Address(UserList}_{t^*, r^*})>)\).
15 —— For each \(u^*\) of \((r^*, t^*, u^*)\) triplets,
16 ——— Set \(U_{t^*, r^*} \leftarrow U_{t^*, r^*} \cup \{u^*\}\), \(f_{t^*, r^*} += 1\), \(f_{t^*, r^*, u^*} += 1\).
17 ——— Execute \(Update(\text{ResourceList}_{t^*}, f_{t^*, r^*}), Add(\text{UserList}_{t^*, r^*}, <u^*, f_{t^*, r^*, u^*}>)\).
18 — Return BasicSocialInvertedIndex = \{ \(<t, f_{t,R}, \text{Address(ResourceList})> | t \in \text{TagIndex}\}\).
Index maintenance

- **Three strategies for maintenance**
  - Rebuild
  - Intermittent merge
  - Incremental update

- **Four important observations of tagged resources**
  - The collection is highly dynamic.
  - Temporal information of tagging can be very useful.
  - Once users annotate a resource with tags, they rarely change their tags.
  - The only update operation for a resource is the additions of new sets of tags assigned by new users.
Index maintenance

- From the above we propose a merge-based index update method:
  - Identify the new triplets that have been added since the last update using the timestamp information of tagging (need to store the tagging history).
  - Build a social inverted index with these new triplets.
  - Merge this (small) social inverted index with the existing social inverted index.
Outline

• Introduction
• Social inverted index
• Evaluation
• Conclusion
Evaluation

- We choose three social-tagging systems for the datasets: Delicious (DL), BibSonomy (BS), and CiteULike (CU).

- The goal of the experiments is to show that the social inverted index is better than (1) a normal inverted index and (2) no inverted index. We compared two types of inverted index: N (normal) and B (basic with timestamp weights). In other words, Type N is a normal inverted index with no user sublists. Type B is the basic social inverted index with timestamp information as user weights.

<table>
<thead>
<tr>
<th>STS</th>
<th>Method</th>
<th>No. of triplets</th>
<th>No. of tags</th>
<th>No. of resources</th>
<th>No. of users</th>
<th>Completeness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delicious (DL)</td>
<td>Crawling</td>
<td>13,510,165</td>
<td>300,901</td>
<td>10,826</td>
<td>637,166</td>
<td>N</td>
</tr>
<tr>
<td>BibSonomy (BS)</td>
<td>Database</td>
<td>2,727,080</td>
<td>222,958</td>
<td>873,467</td>
<td>7,238</td>
<td>Y</td>
</tr>
<tr>
<td>CiteULike (CU)</td>
<td>Database</td>
<td>14,028,761</td>
<td>633,443</td>
<td>3,051,409</td>
<td>89,461</td>
<td>Y</td>
</tr>
</tbody>
</table>
Index Construction

Table 3. Comparison of time costs for the three steps of index construction (seconds)

<table>
<thead>
<tr>
<th></th>
<th>Run building</th>
<th>Run merging</th>
<th>Sorting</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>DL</td>
<td>Type N</td>
<td>109</td>
<td>675</td>
<td>111</td>
</tr>
<tr>
<td></td>
<td>Type B</td>
<td>261 (+139%)</td>
<td>2244 (+232%)</td>
<td>389 (+250%)</td>
</tr>
<tr>
<td>BS</td>
<td>Type N</td>
<td>136</td>
<td>1014</td>
<td>180</td>
</tr>
<tr>
<td></td>
<td>Type B</td>
<td>166 (+22%)</td>
<td>1393 (+37%)</td>
<td>250 (+39%)</td>
</tr>
<tr>
<td>CU</td>
<td>Type N</td>
<td>597</td>
<td>4061</td>
<td>731</td>
</tr>
<tr>
<td></td>
<td>Type B</td>
<td>747 (+25%)</td>
<td>5629 (+39%)</td>
<td>1007 (+38%)</td>
</tr>
</tbody>
</table>
### Table 4. Space costs for a tag index (megabytes)

<table>
<thead>
<tr>
<th></th>
<th>DL</th>
<th>BS</th>
<th>CU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type N and B</td>
<td>13.01</td>
<td>10.00</td>
<td>30.17</td>
</tr>
</tbody>
</table>

### Table 5. The space costs for resource-posting lists plus user sublists (megabytes)

<table>
<thead>
<tr>
<th></th>
<th>DL</th>
<th>BS</th>
<th>CU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type N</td>
<td>10.51</td>
<td>20.80</td>
<td>95.21</td>
</tr>
<tr>
<td>Type B</td>
<td>311.05 ( +2860%)</td>
<td>68.62 ( +230%)</td>
<td>710.64 ( +646%)</td>
</tr>
</tbody>
</table>
They show that 95% of the lengths of the resource-posting lists and user sublists were less than 9 and 26, respectively. This implies that most resources in BibSonomy and CiteULike are annotated by one single user.
Query evaluation

- The second experiment compared the time costs for evaluating queries of Types N and Type B. To emphasize the need to maintain the user sublists in the inverted index.

- Type B can handle temporal queries, but Type N can’t.

- To assist Type N, we acquired the necessary information on the fly from the triplet table stored in an RDB and a NoSQL DB.

- We then calculated the cosine similarity scores between 100 temporal queries (from the start date of 1,1,2010 to the end date of 31,12 2010) and the corresponding resources to get a ranked list of resources by using the normal inverted index and the basic social inverted index.

<table>
<thead>
<tr>
<th>Table 7. Comparison of execution times for query evaluations (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type B</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>DL</td>
</tr>
<tr>
<td>BS</td>
</tr>
<tr>
<td>CU</td>
</tr>
</tbody>
</table>
The third experiment compared the time costs for calculating tag co-occurrence frequencies. In comparison with the social inverted index of Type B, we used an triplet table as a baseline.

We used a set of randomly chosen tags such that the resource frequency of each tag was greater than 1 and smaller than 30,000, and the sample size was 1% of the original size.

We then calculated the macro-level and micro-level co-occurrence frequencies for all pairs of the chosen tags.

<table>
<thead>
<tr>
<th></th>
<th>MacroCoF</th>
<th>MicroCoF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Type B</td>
<td>RDB</td>
</tr>
<tr>
<td>DL</td>
<td>96</td>
<td>13,733</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(×143)</td>
</tr>
<tr>
<td>BS</td>
<td>83</td>
<td>10,317</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(×124)</td>
</tr>
<tr>
<td>CU</td>
<td>848</td>
<td>57,796</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(×68)</td>
</tr>
</tbody>
</table>
Index maintenance

Table 9. Statistics for monthly cumulative versions of DL datasets

<table>
<thead>
<tr>
<th>Version</th>
<th>No. of triplets</th>
<th>No. of tags</th>
<th>No. of resources</th>
<th>No. of users</th>
</tr>
</thead>
<tbody>
<tr>
<td>May 2010</td>
<td>7,923,364</td>
<td>202,764</td>
<td>6,503</td>
<td>508,573</td>
</tr>
<tr>
<td>Jun 2010</td>
<td>8,297,092</td>
<td>209,840</td>
<td>6,671</td>
<td>517,776</td>
</tr>
<tr>
<td>Jul 2010</td>
<td>8,701,895</td>
<td>217,037</td>
<td>6,839</td>
<td>527,886</td>
</tr>
<tr>
<td>Aug 2010</td>
<td>9,115,754</td>
<td>224,143</td>
<td>7,009</td>
<td>537,870</td>
</tr>
<tr>
<td>Sep 2010</td>
<td>9,553,282</td>
<td>232,107</td>
<td>7,200</td>
<td>548,693</td>
</tr>
<tr>
<td>Oct 2010</td>
<td>9,999,912</td>
<td>240,442</td>
<td>7,402</td>
<td>559,802</td>
</tr>
<tr>
<td>Nov 2010</td>
<td>10,483,851</td>
<td>249,265</td>
<td>7,651</td>
<td>570,902</td>
</tr>
<tr>
<td>Dec 2010</td>
<td>10,874,076</td>
<td>256,062</td>
<td>7,858</td>
<td>579,081</td>
</tr>
<tr>
<td>Jan 2011</td>
<td>11,321,369</td>
<td>263,846</td>
<td>8,080</td>
<td>588,459</td>
</tr>
<tr>
<td>Feb 2011</td>
<td>11,773,890</td>
<td>271,840</td>
<td>8,351</td>
<td>598,726</td>
</tr>
<tr>
<td>Mar 2011</td>
<td>12,304,037</td>
<td>280,731</td>
<td>8,720</td>
<td>609,580</td>
</tr>
<tr>
<td>Apr 2011</td>
<td>12,711,291</td>
<td>287,716</td>
<td>9,212</td>
<td>617,405</td>
</tr>
</tbody>
</table>
Outline

• Introduction
• Social inverted index
• Evaluation
• Conclusion
Conclusion

• We present a novel extended inverted index, or social inverted index, for social-tagging-based IR.

• Our social inverted index fully supports the social dimension of social tagging by adding a user sublist to each resource in resource-posting lists.

• Each user in the user sublist has various weights for matching with the user query. It highlights the value of a user who participated in tagging.

• It also allows flexible response to various types of conditional query, an ability that is increasingly required in recent tag-based IR.