RSDC'09: Tag Recommendation Using Keywords and Association Rules

Jian Wang, Liangjie Hong and Brian D. Davison

Department of Computer Science and Engineering
Lehigh University, Bethlehem, PA 18015 USA
{jiw307, lih307, davison}@cse.lehigh.edu

Abstract. While a webpage usually contains hundreds of words, there are only two to three tags that would typically be assigned to this page. Most tags could be found in related aspects of the page, such as the page own content, the anchor texts around the page, and the user's own opinion about the page. Thus it is not an easy job to extract the most appropriate two to three tags to recommend for a target user. In addition, the recommendations should be unique for every user, since everyone's perspective for the page is different. In this paper, we treat the task of recommending tags as to find the most likely tags that would be chosen by the user. We first applied the TF-IDF algorithm on the limited description of the page content, in order to extract the keywords for the page. Based on these top keywords, association rules from history records are utilized to find the most probable tags to recommend. In addition, if the page has been tagged before by other users or the user has tagged other resources before, that history information is also exploited to find the most appropriate recommendations.

1 Introduction

Social bookmarking services allow users to share and store references to various types of World Wide Web (WWW) resources. Users can assign tags to these resources, several words best describing the resource content and his or her opinion. To assist the process of assigning tags, some services would provide recommendations to users as references. In Tatu et al. [5] work, they mentioned that the average number of tags in RSDC'08 bookmarking data is two to three. Thus, it is not an easy task to provide reasonable tag recommendations for the resource with only two to three related tags on average. Tag recommendation is a challenge task in ECML PKDD 2009 where participants should provide either content-based or graph-based methods to help users to assign tags. This work shows some results that aim to this challenge.

The challenge provides description of the resources and posts of the tag. Description contains some basic information about the resources and post is the tuple of user, tag and resource. In the challenge, there are two types of resources, normal web pages, named as *bookmark*, and research publications, named as *bibtex*, with different schemas of descriptions. A post records the resource and the

tags assigned to it by a particular user. The task is to provide ***new*** tags to ***new*** resources with high F-Measure performance on the top five recommendations. The difficulties of this challenge fall in:

- How to take advantage of the record content itself, while the description is very limited? For example, *bookmark* is only described with the title of the web page and a short summary while *bibtex* is usually described with title, publication name, and authors of the paper.
- How to utilize history information to recommend tags which do not appear in the page content? Though we can use keywords to help find possible tags, tags are not just keywords. Tags could be user's opinion about the page, the category of the page, so on and so forth. This kind of tag might be tracked by using history information.
- How to choose the most appropriate two to three tags among the potential pool? By analyzing the page content and history information, we might have a pool which contains the reasonable tag recommendations. Yet we cannot recommend all those to the user. Instead of that, only two to three tags need to be extracted from that pool.

In order to solve the above problems, we propose tag recommendation using both keywords in the content and association rules from history records. After we end with a pool which contains potential appropriate tags, we introduce a method, named *common and combine*, to extract the most probable ones to recommend. Our evaluation showed that integrating association rules can give better F-Measure performance than simply using keywords.

Besides using association rules, some history information will be used more directly, if the resource has been tagged before or the target user tagged other documents before. These history records would greatly improve recommendation performance.

In this paper, we tuned some parameters in our recommendation system to generate the best F-Measure performance while recommending at most five tags.

2 Related Work

Lipczak [3] proposed a recommendation system mainly based on individual posts and the title of the resource. The key conclusion of their experiments is that, they should not only rely on tags previously attached when making recommendations. Sparsity of data and individuality of users greatly reduce the usefulness of previous tuple data. Looking for potential tags they should focus on the direct surrounding of the post, suggesting a graph-based method. Tatu et al. [5] proposed a recommendation system that takes advantage of textual content and semantic features to generate tag recommendations. Their system outperformed other systems in last year's challenge. Katakis et al. [2] proposed a multilabel text classification recommendation system that used titles, abstracts and existing users to train a tag classifier.

In addition, Heymann et al. [1] demonstrated that "Page text was strictly more informative than anchor text which was strictly more informative than surrounding hosts", which suggests that we do not have to crawl other information besides page content. They also showed that the use of association rules can help to find recommendations with high precision.

3 Dataset Analysis and Processing

3.1 Dataset from the Contest

Three table files were provided by the contest, including *bookmark*, *bibtex* and *tas*. The *bookmark* file contains information for bookmark data such as contentID, url, url-hash, description and creation date. The *bibtex* file contains information for bibtex data such as contentID, and all other related publication information. The *tas* file contains information for (user, tag, resource) tuple, as well as the creation date. The detailed characteristics for these files could be found in Table 1. In this work, all contents were transformed into lower case since the evaluation process of this contest ignores case. In the mean time, we filtered the latex format when we exported bibtex data from the database.

Table 1. Detail information about training dataset, provided by the contest

file	# of lines	information
bookmark	263,004	content_id (matches tas.content_id), url_hash (the URL as md5 hash), url, description, extended description, date
bibtex	158,924	content_id (matches tas.content_id), journal, volume chapter, edition, month, day, booktitle, editor, year howPublished, institution, organization, publisher address, school, series, bibtexKey, url, type, description annote, note, pages, bKey, number, crossref, bibtexAbstract simhash0, simhash1, simhash2, entrytype, title, author, misc
tas	1,401,104	userID, tag, content_id (matches bookmark.content_id or bibtex.content_id) content_type (1 = bookmark, 2 = bibtex), date

3.2 Building Experiment Collection

We considered and tried merging duplicate records together in training process yet found it did not help much. Thus we kept the duplicate records when building our experiment collections. Since our proposed tag recommendation approach does not involve a training process, we did not separate the dataset into training one and testing one at first. We evaluated our recommendation system on all

documents in the given dataset. Based on the type of documents, there are three different collections in our dataset:

bookmark collection from dataset provided We created a collection *bookmark_more* to contain all bookmark information which were provided by the contest training dataset. Every document in the collection corresponds to a unique contentID in *bookmark* file. It contains all information for that record, including *description* and *extended description*. There are 263,004 documents in this collection.

During the experiment, we crawled the external webpage for every contentID. Yet the performance showed that the external webpage are not as useful as the simple description provided by the contest. Regardless of performance, it also cost too much time, which is not realistic for online tag recommending. In addition, an external webpage usually contains too many terms, which makes it even harder to extract two to three appropriate terms to recommend as tags.

bibtex collection from dataset provided We created a collection *bibtex_original* to contain all bibtex information which were provided by the original dataset. Every document in the collection corresponds to a unique contentID in *bibtex* file. It contains all information for that record, including all attributes in Table 1 except simhash0, simhash1 and simhash2. There are 158,924 documents docs in this collection.

bibtex collection from external resources If the url of a bibtex record points to some external websites such as portal.acm.org and citeseer, we crawled that webpage and extracted useful information for this record. All these documents are stored in another collection. Similarly, every document in the collection corresponds to a unique contentID in *bibtex* file. There are 3,011 documents in this collection *bibtex_parsed*.

4 Keyword-AssocRule Recommendation

We consider the tag recommendation problem as to find the most probable terms that would be chosen by users. In this paper, $P(X)$ indicates the probability of term X to be assigned to the document as tag. For every document, the term with high $P(X)$ has the priority to be recommended.

4.1 Keyword Extraction

In this step, our assumption is that the more important this term in the document, the more probable for this term to be chosen as tag.

We used two term weighting functions, TF-IDF and Okapi BM25 [4] to extract "keywords" from resources. In a single collection, we calculated TF-IDF and BM25 value for every term in every document.

For TF-IDF, the weighting function is defined as follows:

$$TF - IDF = TF_{t,d} \times IDF_t \tag{1}$$

where $TF_{t,d}$ is the term frequency that equal to the number of occurrences of term t in document d. IDF_t is *inverse document frequency* that is defined as:

$$IDF_t = log\frac{N}{df_t} \tag{2}$$

where df_t is the number of documents in the collection that contain a term t and N is the total number of documents in the corpus.

For Okapi BM25, the weighting function is defined as follows:

$$BM25 = \sum_{i=1}^{n} IDF(q_i)\frac{TF_{t,d}(1+k_1)}{TF_{t,d} + k_1(1 - b + b \times \frac{L_d}{L_{ave}})} \tag{3}$$

where $TF_{t,d}$ is the frequency of term t in document d and L_d and L_{ave} are the length of document d and the average document length for the whole collection. $IDF(q_i)$ here is defined as

$$IDF(q_i) = log\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} \tag{4}$$

The terms in the single document are ranked according to its TF-IDF or BM25 value in decreasing order. A term with high value or high rank is considered to be more important in the document. Thus $P_k(X)$ can be calculated by Algorithm 1.

Algorithm 1 To calculate $P_k(X)$, by using results from keyword extraction method

```
for all documents in the collection do
  rank all terms according to TF-IDF or BM25 value in decreasing order
  for all term X in the document do
    P_k(X) = 100 − rank(X);
    {//rank(X) = 1 indicated the top position, 2 indicated the second position,
    etc. }
  end for
end for
```

As shown in Table 2, TF-IDF performed better than BM25 in tag recommendation process. The following processes in this work were all performed based on results of TF-IDF method.

4.2 Using Association Rules

Recent work by Heymann et al. [1] showed that using association rules could help to find tag recommendation with high precision. They expanded their recommendation pool in decreasing order of confidence. In this paper, we used

Table 2. Performance of key word extraction method in every collection, while recommending at most 5 tags.

collection	BM25			TF-IDF		
	recall	precision	f-measure	recall	precision	f-measure
bibtex_original	0.0951	0.0561	0.0706	0.0989	0.0592	0.0741
bibtex_parsed	0.1663	0.1059	0.1294	0.1800	0.1158	0.1409
bookmark_more	0.1186	0.0940	0.1049	0.1189	0.0943	0.1052

alternative approaches to deeply analyze association rules, which are found in history information. It does help to extract tags which are more likely to be used by users.

Finding association rules in history records We used three key factors in association rules, including *support*, *confidence* and *interest*. Every unique record is treated as the basket and the tags (X, Y, etc.) associated with every record are treated as the items in the basket. For every rule $X \rightarrow Y$, *support* is the number of records that contain both X and Y. *Confidence* indicates the probability of Y in this record if X already associates with the record, i.e., $P(Y|X)$. *Interest* is $P(Y|X) - P(Y)$, showing how much more possible that X and Y associating with the record together.

Table 3. Sample Association rules found in training dataset

bookmark				bibtex			
$X \rightarrow Y$	confidence	support	interest	$X \rightarrow Y$	confidence	support	interest
$blog \rightarrow software$	0.0541	291	0.0454	$systems \rightarrow algorithms$	0.2886	295	0.2757
$blogs \rightarrow blogging$	0.1345	291	0.1333	$algorithms \rightarrow systems$	0.0492	295	0.0470
$blogging \rightarrow blogs$	0.2910	291	0.2885	$systems \rightarrow genetic$	0.2847	291	0.2721
$artery \rightarrow cardiology$	0.9510	291	0.9506	$genetic \rightarrow systems$	0.0497	291	0.0475
$photos \rightarrow photography$	0.3149	290	0.3138	$tagging \rightarrow folksonomy$	0.5097	288	0.5085
$photography \rightarrow photos$	0.3142	290	0.3131	$folksonomy \rightarrow tagging$	0.5115	288	0.5103
$learning \rightarrow foodcooking$	0.1004	290	0.1000	$genetic \rightarrow and$	0.0466	273	0.0441

The rules $X \rightarrow Y$ we constructed all have support > 10, thus at least 10 resources in our training dataset contain both X and Y as tags. As we mentioned before, we did not separate the dataset into training and testing sets. During evaluation, some records might benefit from the rule it contributed at first, yet at least 9 more resources also contributed to the rule. The support limit here is chosen arbitrarily. Two sets of rules are constructed independently, one for bookmark dataset and another one for bibtex dataset. Some sample rules are showed in Table 3.

Choosing appropriate recommendations by using association rules Here the problem becomes to be:

If $X \rightarrow Y$ exists in the association rules, how possible that term Y should be recommended when X is likely to be recommended?

Given $P(X)$ and the confidence value $P(Y|X)$, $P(Y)$ could be calculated according to law of total probability, which is sometimes called as law of alternatives:

$$P(Y) = \sum_n P(Y \cap X_n) \tag{5}$$

or

$$P(Y) = \sum_n P(Y \mid X_n)P(X_n) \tag{6}$$

since $P(Y \cap X_n) = P(Y \mid X_n)P(X_n)$. According to the above equations, the algorithm to calculate $P_a(Y)$, which is called $Assoc(Y)$ in this paper, is shown in Algorithm 2.

Algorithm 2 To calculate $P_a(X)$, by using association rules

```
for all documents in the collection do
  for all term X in the document do
    for all association rule X → Y do
      P_a(Y)+ = (confidence of X → Y) * P_k(X);
      {//P_k(X) is calculated by Algorithm 1}
    end for
  end for
end for
```

4.3 Combining Keyword Extraction with Association Rules Results

After $P_k(X)$ and $P_a(X)$ are calculated for every term in the document, one method, in Algorithm 3, is to linearly combine the two values to calculate the final probability $P_c(X)$ for recommending a term X.

Similarly, term with higher $P_c(X)$, i.e., higher rank in Combined results has the priority to be recommended.

The experiments showed that *weight* could affect the F-Measure performance and the optimal *weight* to combine is different for every collection. Figure 1 shows the effect of *weight* in *bibtex_parsed* collection, where F-Measure reaches the peak during increase of *weight* from 0.1 to 0.9. This trend is similar in other two collections. Our experiments indicated that the optimal *weight* to achieve best F-Measure for *bibtex_parsed*, *bibtex_original*, *bookmark_more* is 0.7, 0.5 and 0.5, respectively. The evaluation results with optimal *weight* for every collection, in this step, is shown in the second column of Table 4. Compared to the TF-IDF results in the first column, it is obvious that the association rules can greatly help to improve the F-Measure performance.

Another method we found that worked well is *common and combine*. In *common* step, if the term in top rank of keyword extraction results do have $Assoc(X) > 0$, then recommend this term. In *combine* step, extract terms with

Algorithm 3 To calculate $P_c(X)$, by linearly combining results from TF-IDF & association rules

for all documents in the collection **do**
  $TF-IDFmax$ = maximum $TF-IDF(X)$ for all terms in this document
  $Assocmax$ = maximum $Assoc(X)$ for all terms in this document
  for all term X in the document **do**
    $TF-IDF(X) = TF-IDF(X)/TF-IDFmax$;
    {// normalize $TF-IDF(X)$ value}
    $Assoc(X) = Assoc(X)/Assocmax$;
    {// normalize $Assoc(X)$ value, $Assoc(X) = P_a(X)$ in Algorithm 2.}
    $Combined(X) = TF-IDF(X) * weight + Assoc(X) * (1 - weight)$;
    {//linearly combine the two values}
    rank terms according to decreasing order of Combined(X);
    $P_c(X) = 100 - rank\ of\ Combined(X)$;
    {//rank = 1 indicated the top position, 2 indicated the second position, etc.}
  end for
end for

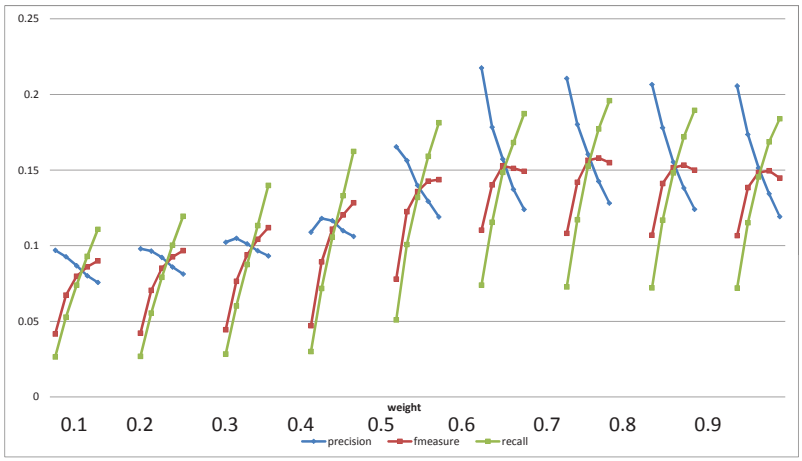

Fig. 1. Performance for different weight in bibtex_parsed. For every weight, no. of tags changes from 1 to 5.

Table 4. Performance for only using TF-IDF results, linearly combining results of TF-IDF & association rules, and common & combine the two results, for the top N tag recommendations

TF-IDF				linearly combining results				common & combine			
bibtex_original											
Top N	recall	precision	f-measure	Top N	recall	precision	f-measure	Top N	recall	precision	f-measure
1	0.0199	0.0636	0.0304	1	0.0339	0.1105	0.0519	1	0.0344	0.1123	0.0527
2	0.0378	0.0610	0.0467	2	0.0593	0.0979	0.0739	2	0.0619	0.1018	0.0770
3	0.0579	0.0603	0.0591	3	0.0824	0.0900	0.0860	3	0.0848	0.0927	0.0886
4	0.0787	0.0598	0.0680	4	0.1046	0.0849	0.0937	4	0.1065	0.0867	0.0956
5	0.0989	0.0592	0.0741	5	0.1244	0.0802	0.0975	5	0.1264	0.0816	0.0992
bibtex_parsed											
Top N	recall	precision	f-measure	Top N	recall	precision	f-measure	Top N	recall	precision	f-measure
1	0.0708	0.2033	0.1050	1	0.0728	0.2106	0.1081	1	0.0723	0.2155	0.1083
2	0.1138	0.1710	0.1367	2	0.1171	0.1802	0.1419	2	0.1212	0.1871	0.1471
3	0.1438	0.1487	0.1462	3	0.1527	0.1605	0.1565	3	0.1549	0.1635	0.1591
4	0.1665	0.1316	0.1470	4	0.1771	0.1425	0.1580	4	0.1778	0.1432	0.1586
5	0.1800	0.1158	0.1409	5	0.1959	0.1281	0.1549	5	0.1968	0.1291	0.1559
bookmark_more											
Top N	recall	precision	f-measure	Top N	recall	precision	f-measure	Top N	recall	precision	f-measure
1	0.0388	0.1285	0.0596	1	0.0449	0.1547	0.0696	1	0.0460	0.1599	0.0715
2	0.0693	0.1172	0.0871	2	0.0846	0.1400	0.1055	2	0.0872	0.1487	0.1099
3	0.0919	0.1080	0.0993	3	0.1133	0.1286	0.1205	3	0.1165	0.1358	0.1254
4	0.1077	0.1001	0.1038	4	0.1375	0.1202	0.1283	4	0.1415	0.1255	0.1330
5	0.1189	0.0943	0.1052	5	0.1581	0.1132	0.1319	5	0.1623	0.1172	0.1361

high $P_c(X)$ for recommendation. The total number of tags to recommend is controlled by k, the number of tags to check in the *common* step is *common-no, and the number of tags to extract in the combine step is* combine-no. Detailed steps are shown in Algorithm 4.

Since the evaluation of this contest only cares for the first 5 tags to recommend, we set $k = 5$. If *common-no = 10 and* combine-no = 5, the results for all three collections are shown in third column of Table 4.

Generally speaking, F-Measure increases with the increase of *common-no and reaches the peak near* common-no = 20. At the same time, it reaches its highest point as *combine-no increases, and remains the same level with the further increase of* combine-no. Since the total number of tags to recommend is fixed to be 5, the *combine* step will stop before it reaches the limit of how many tags to check, i.e., *combine-no. Thus if* combine-no is greater than a certain number, it won't affect the f-measure performance anymore.

Since the recommendations would be further modified by history results, we set $k = 80$, *common-no=10, and* combine-no=80 here.

If only recommending at most 5 tags, the F-Measure performance of all above methods, including only using TF-IDF, linearly combining results of TF-IDF & association rules, and *common & combine* the two, are shown in Figure 2. It is obvious that using association rules can greatly enhance the TF-IDF performance, either by linear combination or *common & combine*. *Common & combine* method is slightly better than linearly combining the two.

4.4 Checking Resource or User Match with History Records

In this section, historical information is used more directly. We performed 10-fold cross validation to report the performance in this section.

Algorithm 4 Common and Combine, to integrate results from TF-IDF & association rules

```
for all documents in the collection do
  count = 0;
  {// common step}
  for i = 1 to common-no do
    {// common-no is the parameter to tune. It controls how many terms to check
    in TF-IDF results.}
    extract term X with TF-IDF rank = i;
    if Assoc(X) > 0 then
      recommend this term X;
      count++;
      {//count is the number of tags that have been recommended in common
      step.}
    end if
    i++;
  end for
  {// combine step}
  rank all terms by P_c(X) in decreasing order.
  {// P_c(X) is the combination value of keyword extraction and association rules
  results, calculated by Algorithm 3.}
  for j = 1 to (k − count) do
    {// k is the total number of tags to recommend, k − count is the number of tags
    to recommend in combine step}
    extract the term Y with (rank in P_c(X) results) = j;
    if Y is not in the recommendation list then
      recommend this term Y;
    end if
    j++;
    if j > combine-no then
      {// combine-no is the parameter to tune. It controls how many terms to check
      in combined results of TF-IDF & association rules.}
      exit the combine step;
    end if
  end for
end for
```

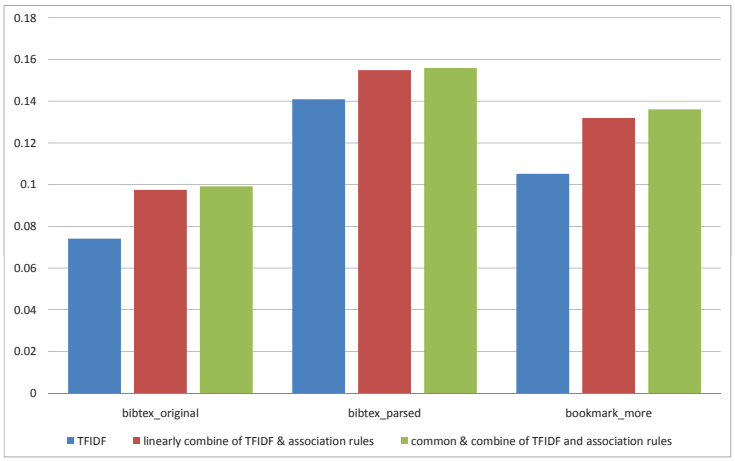

Fig. 2. F-Measure performance for different methods in all collections. For every method, at most 5 tags are recommended. The three methods to compare are only using TF-IDF, linearly combining results of TF-IDF & association rules, and common & combine

Resource match If the bookmark or bibtex in the testing dataset already appeared before in training dataset, regardless of which user assigned the tags, the tags that were assigned before would be directly inserted into our recommendation list for this document. These tags from historical information have higher priority than the tags that were recommended in previous steps.

User match Suppose the tags that are assigned by users previously in the training dataset, regardless of to which documents, make up the user's *tagging vocabulary*. Our assumption here is that every user prefers to use tags in his/her own tagging vocabulary, as long as the tags are relevant to the document. Thus the tags in the user's tagging vocabulary would be given higher priority. The *common and combine* algorithm is again applied here. In *common* step, if the terms with high rank in previous steps do appear in user's tagging vocabulary, then recommend this term. In *combine* step, extract terms with high ranks in previous steps to recommend. The number of tags to check in the *common* step is *common-no, and the number of tags to extract in the combine step is* combine-no.

The two parameters, *common-no and* combine-no, are tuned to achieve the best F-Measure performance when recommending at most 5 tags. *common-no is fixed to be 53 in Figure 3, while* combine-no increases from 1 to 5. In that figure, it shows that F-Measure increases and reaches the peak point at *combine-no = 1. In Figure 4,* combine-no is fixed to be 1 and *common-no increases from 1 to 80. F-Measure increases with the initial increase of* common-no and reaches the peak point in the middle. In this work, we set *common-no = 53, and* combine-no = 1.

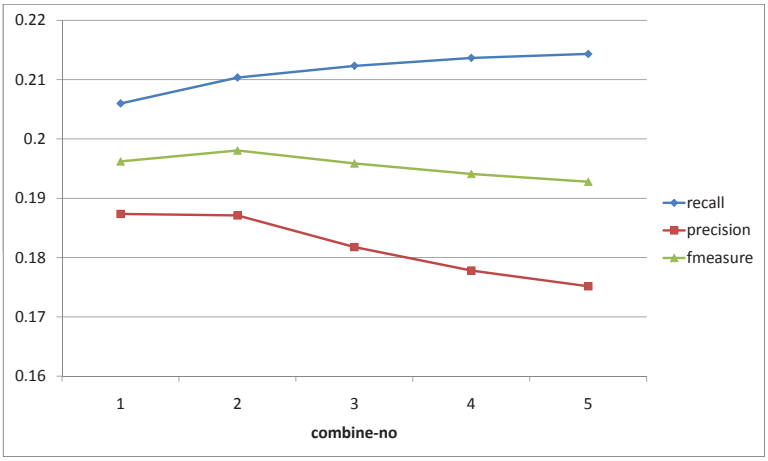

Fig. 3. In *common and combine* method for checking user match, performance for different *combine-no and fixed* common-no = 53. At most 5 tags are recommended.

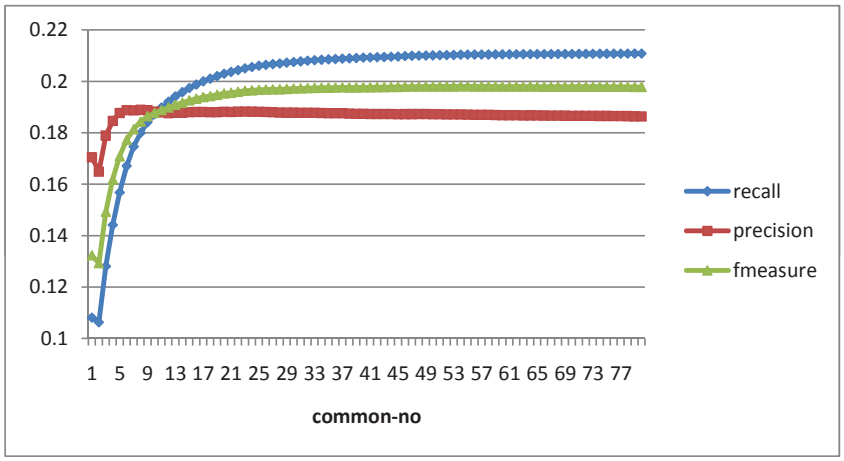

Fig. 4. In *common and combine* method for checking user match, performance for different *common-no and fixed* combine-no = 1. At most 5 tags are recommended.

Exact match with same user and same resource In this step, if user has tagged the same document in the training dataset, then the tags he used before for this document would be directly recommended again.

4.5 Combining Results in all Collections

According to the performance of each collection, our priority to combine the results is shown in Table 5.

Table 5. Priority to combine results

Priority	Method
Higher to lower	Tags from records that has exact match with same user and same bookmark/bibtex Tags from records that has match with same user Tags from records that has match with same resource (bookmark url or bibtex publication) common & combine results of *bibtex_parsed* common & combine results of *bibtex_original* common & combine results of *bookmark_more*

For example, if a record both exists in *bibtex_parsed* and *bibtex_original*, the results for this record are chosen from *bibtex_parsed* instead of *bibtex_original*, since the former one has higher priority.

If we only consider to combine the *common & combine* results for all three collections, the best performance is shown in column *without checking the history records* of Table 6.

Table 6. Performance for without checking history records, resource match with higher priority and user match with higher priority, for the top N tag recommendations

without checking history records				resource match higher				user match higher			
Top N	recall	precision	f-measure	Top N	recall	precision	f-measure	Top N	recall	precision	f-measure
1	0.0415	0.1395	0.0639	1	0.0835	0.2312	0.1226	1	0.0867	0.2396	0.1273
2	0.0783	0.1305	0.0979	2	0.1344	0.2143	0.1652	2	0.1374	0.2220	0.1698
3	0.1059	0.1204	0.1126	3	0.1667	0.1980	0.1810	3	0.1684	0.2064	0.1855
4	0.1292	0.1115	0.1197	4	0.1915	0.1866	0.1890	4	0.1916	0.1954	0.1935
5	0.1510	0.1046	0.1235	5	0.2118	0.1778	0.1933	5	0.2104	0.1871	0.1981

If step *Tags from records that match with same user* has lower priority than *tags from records that match with same resource*, the best result is shown in column *resource match higher* of Table 6. Otherwise, the best result is shown in column *user match higher* of Table 6. The results indicate that even for those bookmarks that were tagged by other users before, it is still beneficial to consider the target user's own tagging vocabulary.

To sum up, the best performance on training dataset is shown in Table 7, including the detailed results only for bookmark and bibtex.

5 Conclusions and Future Work

In this paper, we proposed a tag recommendation system using keywords in the page content and association rules from history records. If the record resource

Table 7. Best performance on training dataset, only for bookmarks and only for publications, for the top N tag recommendations

all resources				only for bookmark				only for bibtex			
Top N	recall	precision	f-measure	Top N	recall	precision	f-measure	Top N	recall	precision	f-measure
1	0.0867	0.2396	0.1273	1	0.0771	0.2364	0.1163	1	0.1025	0.2448	0.1445
2	0.1374	0.2220	0.1698	2	0.1296	0.2215	0.1635	2	0.1504	0.2228	0.1796
3	0.1684	0.2064	0.1855	3	0.1613	0.2056	0.1808	3	0.1803	0.2076	0.1930
4	0.1916	0.1954	0.1935	4	0.1843	0.1932	0.1887	4	0.2038	0.1990	0.2014
5	0.2104	0.1871	0.1981	5	0.2035	0.1841	0.1933	5	0.2218	0.1921	0.2059

or target user appeared before, the history tags would be used as references to recommend, in a more direct way. Our experiments showed that association rules could greatly improve the performance with only keyword extraction method, while history information could further enhance the F-Measure performance of our recommendation system.

In the future, other keyword extraction method can be implemented to compare with TF-IDF performance. In addition, graph-based methods could be combined with our recommendation approach to generate more appropriate tag recommendations.

Acknowledgments

This work was supported in part by a grant from the National Science Foundation under award IIS-0545875.

References

1. P. Heymann, D. Ramage, and H. Garcia-Molina. Social tag prediction. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 531–538, New York, NY, USA, 2008. ACM.
2. I. Katakis, G. Tsoumakas, and I. Vlahavas. Multilabel text classification for automated tag suggestion. In *Proceedings of the ECML/PKDD 2008 Discovery Challenge Workshop, part of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2008.
3. M. Lipczak. Tag recommendation for folksonomies oriented towards individual users. In *Proceedings of the ECML/PKDD 2008 Discovery Challenge Workshop, part of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2008.
4. S. E. Robertson. Overview of the OKAPI projects. *Journal of Documentation*, 53(1):3–7, 1997.
5. M. Tatu, M. Srikanth, and T. D'Silva. Rsdc'08: Tag recommendations using bookmark content. In *Proceedings of the ECML/PKDD 2008 Discovery Challenge Workshop, part of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 96–107, 2008.