A Tag Recommendation System based on contents

Ning Zhang, Yuan Zhang, and Jie Tang

Knowledge Engineering Group
Department of Computer Science and Technology, Tsinghua University, Beijing, China
zntsinghua1117@gmail.com, fancyzy0526@gmail.com and jietang@tsinghua.edu.cn

Abstract. Social bookmarking tools become more and more popular nowadays and tagging is used to organize information and allow users to recall or search the resources. Users need to type the tags whenever they post a resource, so that a good tag recommendation system can ease the process of finding some useful and relevant keywords for users. Researchers have made lots of relevant work for recommendation system, but those traditional collaborative systems do not fit to our tag recommendation. In this paper, we present two different methods: a simple language model and an adaption of topic model. We evaluate and compare these two approaches and show that a combination of these two methods will perform better results for the task one of PKDD Challenge 2009.

1 Introduction

With the event of social resource sharing tools, such as BibSonomy[1], Flickr[2], del.icio.us[3], tagging has become a popular way to organize the information and help users to find other users with similar interests and useful information within a given category. Tags posted by a user are not only relevant to the content of the bookmark but also to the certain user. According to [3], the collection of a user's tag assignments is his/her *personomy*, and *folksonomy* consists of collections of personomies. From the available training data, we can find that some tags might just be words extracted from the title, some tags might be the concept or main topic of the resource, and other might be very specific to a user.(see Table 1). The last three lines of the table show that the user 293 post tags like swss0603, swss0609, swss0602, which are very specific to the user. Since the test data for task one contains posts whose user, resource or tags are not in the training data, some traditional collaborative recommendation systems might not perform well. It is because most of the collaborative recommendation systems cannot recommend tags which are not in the tag set of the training data. This paper presents our tag recommendation system,

[1] http://www.bibsonomy.org
[2] http://www.flick.com
[3] http://del.icio.us

which is a combination of two methods: simple Language model and an adaption of topic model according to [7].

USER	TITLE	TAGS	SOURCE OF TAGS
37	SVG: Adobe	adobe, svg	title
787	SourceForge.net: delicious-java	api java, delicious	title
173	Reassessing Working Memory: Comment on Just and Carpenter (1992) and Waters and Caplan (1996)	psycholinguistics, review, workingmemory	concept or topic
293	A Semantic Web Primer	swss0603, ontolex2006, semwebss06, swss0602	specific to the user
293	The ABCDE Format Enabling Semantic Conference Proceedings	semwiki2006,swikig,wiki, eswc2006, semantic	specific to the user
293	Learning of Ontologies for the Web: the Analysis of Existent Approaches	ontologylearning, semanticweb,semwebss06, sw0809, sw080912, swss0609	specific to the user

Table 1: Examples of tag sources

The first method can extract some keywords from the description and other information of the post and they constitute a candidate set. Then we use the relevance of a word and a document to score the words in the candidate set and recommend the words with highest scores. The second method uses an ACT model [7], and it can get some conceptual or topic knowledge of the post. Given a test post, the model can score all the tags which have been posted previously and recommend the tags with highest scores.

These two methods focus on two different aspects. The first method will probably recommend some keywords extracted from the title while the second method uses a probabilistic latent semantic method to recommend tags which are similar to the post in terms of conceptual knowledge. Comparing these two methods, we can find that the tags recommended are always different. Consequently, the combination is a intuitive better way.

This paper is organized as follows: Section 2 reviews recent development in the area of social bookmark tag recommendation systems. Section 3 describes our proposed system and the combination method in details. In section 4, we present and evaluate our experimental results on the test data of ECML PKDD challenge and conclude the results in section 5.

2 Related work

The recent rise of Web 2.0 technologies has aroused the interest of many researchers to the tag recommendation system. Some approaches are based on collaborative information. For example, AutoTag[9] and TagAssist[8] use some information retrieval skills to recommend tags for weblog posts. They recommend tags based on the tags posted to the similar weblogs and they cannot recommend new tags which are not in the training file. FolkRank[4,5], which is an adaption of the famous PageRank algorithm, is a graph-based recommendation system. Also, it cannot recommend new tags not in the training file. The experimental results of FolkRank in [5] reveal that the FolkRank can outperform other collaborative methods. But to some extents FolkRank relies on a dense core of training file and it might not be fit to our task.

All the methods mentioned above are based on collaborative information and similarity between users and resources. However, in the cases when there are many new users and resources in the test data (our task one), those methods cannot perform well. In the RSDC '08 challenge, the participants[1,2] who use methods based on words extracted from the title or semantic knowledge and user's personomy can outperform other methods. Consequently, we propose our tag recommendation system mainly based on the contents.

3 Our Tag Recommendation System

3.1 Notations

First, we define notations used in this paper. We group the data in bookmark by its url_hash and data in bibtex by its simhash1. If some posts in bookmark or bibtex file have the same url_hash or simhash1, they are mapped to one resource r. In bookmark, we extract description, extended description while in bibtex, we extract journal, booktitle, description, bibtexAbstract, title and author. We define these information as the description of resource r. For each resource r and each user u who has posted tags to resource r, assuming that its description contains a vector $\mathbf{w}_d$ of N_d words; a vector $\mathbf{t}_d$ of T_d tags posted to this resource r by the user u_d.Then the training dataset can be represented as

$$D = \{(w_1, t_1, u_1), ..., (w_D, t_D, u_D)\}$$

Table 2 summarizes the notations.

3.2 Language Model

Language model is widely used in natural language processing applications such as speech recognition, machine translation and information retrieval. In our model, first we pick some words to form a candidate set of recommended tags and then score all the words in the candidate set and recommend words with highest scores for our tag

Symbol	**Description**
T	the collection of tags posted in the training data
R	the collection of resources posted in the training data (grouped by the url_hash or simhash1)
U	the collection of users who posted tags in the training data
D	training data set containing tagged resources. D={(**w**$_i$, **t**$_i$,u$_i$,)}, which represents a set of pairs of resources and users, with the assigned tags by the corresponding users.
D'	The test data set containing resources and users. D'={(r$_j$, u$_i$)}$_{\{i,j\}}$. Note that: 1) either the user u$_i$ or the resource r$_j$ may not appear in the training data set.
N_d	number of word tokens in the d ∈ D
T_d	number of tags posted by user u to resource r in d ∈ D
$\mathbf{w}_d$	vector form of word tokens in d ∈ D
$\mathbf{t}_d$	vector form of tags in d ∈ D
u_d	the user in d ∈ D
$\tilde{T}(u,r)$	the set of tags that will be recommended for a given user u and a given resource r
z	hidden topic layer in ACT model

Table 2: Notations.

recommendation system. The candidate set C is composed with two subset, C_1 and C_2, i.e. $C = C_1 \cup C_2$.

We extract useful words from the description of the active resource r* in the test data. Then we remove all the characters which are neither numbers nor letters and get rid of the stop words in the English dictionary. The rest words form the part of the candidate set C_1. For each $t_1 \in C_1$, we have the following generative probability:

$$P_1(t_1 \mid r^*) = \frac{N_d}{N_d + \lambda} \cdot \frac{tf(t_1, d)}{N_d} + (1 - \frac{N_d}{N_d + \lambda}) \cdot \frac{tf(t_1, D')}{N_{D'}} \quad (1)$$

where N_d is the number of word tokens in the description d of r^*, tf(t_1,d) is the word frequency(i.e., occurring number) of word t_1 in the description of d of r^*, $N_{D'}$ is the number of word tokens in the entire test dataset, and tf(t_1,D') is the word frequency of word t_1 in the collection D'. λ is the Dirichlet smoothing factor and is commonly set according to the average document length, i.e. $N_{D'}/|D'|$ in our cases.

As for C_2, in order to get more information about the new resource, we take the similarity between resources into consideration and add tags previously posted to the similar resource into C_2. The similarity of resource is determined by the url of the resource. Each url can be split into several sections, for example, 'http://www.kde.cs.uni-kassel.de/ws/dc09' will be split into three sections: 'www.kde.cs.uni-kassel.de', 'ws' and 'dc09'. The similarity between r_1 and r_2 is defined as follows, sim(r_1,r_2) = 2^(number of the identical sections of url$_1$ and url$_2$ – maximum number of sections of url$_1$ and url$_2$). For each resource r, we will choose three most similar urls to the url of r and their corresponding resources form the neighbor of the resource r, noted as neigh(r). C_2 is compose of the tags previous posted to the neigh(r^*). For each $t_2 \in C_2$, we have the following generative probability:

$$P_2(t_2 \mid r^*) = \sum\nolimits_{r \in neigh(r^*)} \left[\left(\frac{T_r}{T_r + \lambda} \cdot \frac{n(t_2, r)}{T_r} + (1 - \frac{T_r}{T_r + \lambda}) \cdot \frac{n(t_2, R)}{|T|} \right) \cdot sim(r, r^*) \right] \quad (2)$$

where $n(t_2,r)$ is the number of times that tag t_2 has been posted to the resource r and $n(t_2,R)$ is the number of times that tag t_2 has been posted in the training data. T_r is the number of tags posted to the resource r and λ is the Dirichlet smoothing factor and is commonly set according to the average document length, i.e. $|T|/|R|$

Now, we have the definition of the candidate set $C = C_1 \cup C_2$ and for an active resource r^*, we have $P_1(t_1|r^*)$ for $t_1 \in C_1$ and $P_2(t_2|r^*)$ for $t_2 \in C_2$. Then the set of recommended tags will be: $\widetilde{T}(u,r) \coloneqq \mathrm{argmax}^n_{t \in C} (P_1(t|r^*) + P_2(t|r^*))$ where n is the number of recommended tags.

3.3 ACT model

The model we use is called Author-Conference-Topic (ACT) model[7], which is an adaptation of topic model. The initial model was used to simultaneously modeling papers, authors, and publication venues within a unified probabilistic topic model. The model utilizes the topic distribution to represent the inter-dependencies among authors, papers and publication venues. In our task, for each $(\mathbf{w}_d, \mathbf{t}_d, u_d) \in D$, we map the $\mathbf{w}_d$ to conference information, map $\mathbf{t}_d$ to author of the paper, map u_d to the publication venue. Consequently, after modeling, we can get probabilistic relations among description, tags and user given a post. In details, we can get these inter-dependencies: P(z|t), P(w|z) and P(u|z), where z is a hidden topic layer in the topic model. From this model, we can utilize the hidden topic layer to learn some conceptual knowledge of the post. The number of topic z can be set manually.
After obtaining the probability of P(z|t), P(w|z) and P(u|z) from the model, we can derive that for each word $w \in \mathbf{w}_d$, each tag $t \in \mathbf{t}_d$, we have:

$$P(w \mid t) = \sum\nolimits_z P(w \mid z) P(z \mid t)$$
$$P(u \mid t) = \sum\nolimits_z P(u \mid z) P(z \mid t)$$

To recommend tags for a given user u' and a given resource r', we will score all the tags $t \in T$ using probabilistic methods as follows:

$$P(t \mid u', r') \propto P(t, u' r') = P(t) P(u' \mid t) P(r' \mid t) \approx P(t) P(u' \mid t) \prod\nolimits_{w \in r'} P(w \mid t) \quad (3)$$

There are two things that needed to be mentioned here. First, if the given user $u' \notin U$, which means the given user is a new user. Then we cannot obtain P(u'|t) in the equation (3), in that case we will set the value to 1 and the equation (3) will become:

$$P(t) \prod\nolimits_{w \in r'} P(w \mid t) \approx P(t) P(r' \mid t) \propto P(t \mid r') \quad (4)$$

Because the user is a new one, so that we have nothing about his/her history of posting tags in our training data, in that case equation (4) is reasonable. Secondly, not all the words in the given resource are in our training data, so when calculating equation (3), we will ignore the word which has not appeared in the training data.

The set of recommended tags for a given user u' and a given resource r' will be: $\widetilde{T}(u', r') \coloneqq \mathrm{argmax}^{n}_{t \in T} P(t|u', r')$ where n is the number of recommended tags, T is the collection of tags posted in the training file.

3.4 Combination

We have proposed two different methods to recommend tags, model one focuses on the useful words extracted from the description or title of the resource while the second model focuses on the conceptual knowledge and probabilistic relations among tags, resource and users. We are interested in the following problem: **Can we combine these two models to perform a better result for tag recommendation?**

```
Input: a given resource r and a given user u and the result of ATC model $P(t)$,$P(w|t)$
    and $P(u|t)$ for all tags, users, words in the training file.
Output: $\widetilde{T}(u, r)$the set of recommended tags
begin
    //Model one
    $\mathbf{w_1} \leftarrow$ words in the candidate set C
    foreach $w \in \mathbf{w_1}$ do
        $score1[w] = P_1(w|r) + P_2(w|r)$
    end
    $max\_score1 \leftarrow \max_{w \in w_1} score1[w]$
    //Model two
    foreach $t \in T$ do
        $score2[t] = P(t)P(u|t) \prod_{w \in r} P(w|t)$
    end
    $max\_score2 \leftarrow \max_{t \in T} score2[t]$
    //Combination
    foreach $t \in \mathbf{w_1} \cup T$ do
        $score[t] = score1[t] + score2[t] * max\_score1 / max\_score2$
    end
    $\widetilde{T}(u, r) \coloneqq \mathrm{argmax}^{n}_{t \in T} score[t]$
end
```

Algorithm 1: The combined tag recommendation system

We have tried some different approaches to combine these two models. A simple method is to combine the scores of these two models and recommend tags with highest scores after combination (Algorithm 1). We can make use of the two scores calculated in the two approaches and there are two things worthy to be noted here: 1) model one only calculates the scores of tags in the candidate set but model two calculates all the tags $t \in T$. 2) due to the different distribution of scores, we need to normalize the two scores before combination. In order to solve the first problem, we consider all the tags $t \in C \cup T$ where C is the candidate used in the model one. In terms of normalization, we make the $\|score1\|_\infty = \|score2\|_\infty$ and then add these two score, if a tag t is in the candidate set C but not in the T, the score2[t] = 0 and if a tag t is in T but not in C, then the score1[t] =0.

4 Experimental Results

4.1 Dataset

We evaluate our experimental results using the evaluation methods provided by the organizers of ECML PKDD discovery challenge 2009. The training set and the test set are strictly divided and we use the cleaned dump as our training set for our tag recommendation system.

Here are some statistical information about training data and test data: There are 1,401,104 tag assignments. 263,004 bookmarks are posted and among which there are 235,328 different url resources while 158,924 bibtex are posted and among which there are 143,050 different publications. From this, we can see that many resources appear just once in the training file. There are 3,617 users and 93,756 tags in all in the training file. The average number of tags posted to bookmark is 3.48 and the average number of tags posted to bibtex is 3.05.

In the test data, there are 43,002 tag assignments, 16,898 posted bookmarks and 26,104 posted bibtex. Among all the posts, there are only 1,693 bookmark resources and 2,239 bibtex resources which are in the training file. The average number of tags posted to bookmark is 3.81 and the average number of tags posted to bibtex is 3.82.

4.2 Data Preprocessing

The training data is provided by the organizers of the ECML PKDD, we establish three tables, bookmark, bibtex and tas in our MySQL database. In order to get the similarity between resources, we need to preprocess the url field. For each url in the bookmark, we eliminate the prefix such as ‘http://’, ‘https://’ and ‘ftp://’. Then we split the url by the character ‘/’. For example, a url ‘http://www.kde.cs.uni-kassel.de/ws/dc09’ will be split into ‘www.kde.cs.uni-kassel.de’, ‘ws’ and ‘dc09’. As we mentioned above, we define some information extracted from the table as the description of a resource r. We eliminate the stop words in the English dictionary and stem the words, for example, ‘knowledge’ will be stemmed to ‘knowledg’ and both ‘biology’ and ‘biologist’ will be stemmed to ‘biologi’.

4.3 Results and analysis

As performance measures we use precision, recall and f-measure. For a given user u and a given resource r, the true tags are defined as TAG(u,r), then the precision, recall and f-measure of the recommended tags $\widetilde{T}(u,r)$ are defined as follows:

$$\mathrm{recall}\left(\widetilde{T}(u,r)\right) = \frac{1}{|U|}\sum_{u\in U}\frac{|TAG(u,r) \cap \widetilde{T}(u,r)|}{|TAG(u,r)|}$$

$$\mathrm{precision}\left(\widetilde{T}(u,r)\right) = \frac{1}{|U|}\sum_{u\in U}\frac{|TAG(u,r) \cap \widetilde{T}(u,r)|}{|\widetilde{T}(u,r)|}$$

$$f-\mathrm{measure}\left(\widetilde{T}(u,r)\right) = \frac{2 \times \mathrm{recall} \times \mathrm{precision}}{\mathrm{recall} + \mathrm{precision}}$$

4.3.1 Performance of Language model

In table 3, we show the performance of Language model on the test data provided by the organizers of ECML PKDD challenge 2009. We show the performance of bookmark, bibtex and the whole data respectively. From the table, we can find that the result of bookmark is better than that of bibtex and we can have a highest f-measure of 13.949% when the number of recommended tags is 10.

	bookmark(%)	bibtex(%)	overall(%)
1	5.256/18.287/8.165	3.535/13.714/5.620	4.207/15.509/6.619
2	9.298/17.417/12.124	6.399/12.571/8.481	7.534/14.474/9.910
3	12.467/16.789/14.308	9.030/11.941/10.284	10.377/13.845/11.862
4	14.755/16.114/15.405	11.242/11.399/11.320	12.619/13.251/12.927
5	16.314/15.634/15.967	12.889/10.879/11.799	14.231/12.747/13.448
6	17.288/15.195/16.174	14.361/10.460/12.104	15.507/12.320/13.731
7	17.964/14.874/16.273	15.564/10.103/12.253	16.503/11.977/13.880
8	18.459/14.610/16.311	16.531/9.779/12.289	17.285/11.677/13.938
9	18.827/14.430/16.338	17.280/9.496/12.256	17.884/11.434/13.949
10	19.127/14.270/16.346	17.894/9.243/12.190	18.374/11.218/13.931

Table 3: performance of language model on the test data, the numbers are shown in the following format: recall/precision/f-measure

4.3.2 Performance of ACT model

In table 4, we show the performance of Language model on the test data provided by the organizers of ECML PKDD challenge 2009. We show the performance of bookmark, bibtex and the whole data respectively. From the table, we can see that the performance of ACT model is worse than the language model. We have the highest f-measure of 3.077% when the number of recommended tags is set to five. Also, the performance of bookmark is a little bit better than the bibtex and the reason might be that description in bibtex has some information which is irrelevant to the main topic of the publication.

	bookmark(%)	bibtex(%)	overall(%)
1	2.142/6.800/3.258	0.944/2.758/1.406	1.415/4.346/2.135
2	3.523/5.628/4.333	1.431/2.320/1.770	2.253/3.620/2.778
3	4.519/4.825/4.667	1.885/2.076/1.976	2.920/3.156/3.034
4	5.179/4.196/4.636	2.167/1.868/2.007	3.351/2.783/3.041
5	5.829/3.815/4.612	2.466/1.721/2.027	3.788/2.544/3.044
6	6.418/3.536/4.560	2.724/1.582/2.002	4.175/2.350/3.077
7	6.870/3.257/4.419	2.977/1.483/1.980	4.507/2.180/2.939
8	7.377/3.059/4.324	3.205/1.393/1.942	4.844/2.048/2.878
9	7.849/2.891/4.225	3.557/1.346/1.953	5.244/1.953/2.846
10	8.289/2.746/4.126	3.721/1.276/1.900	5.516/1.854/2.775

Table 4: performance of ACT model on the test data, the numbers are shown in the following format: recall/precision/f-measure

4.3.3 Performance after combination

In table 5, we show the performance after the combination of these two models. From the table, we can see that after combination, our recommendation system works a little better than the Language model and has a highest f-measure of 14.398% when recommending five tags.

	final result(%)
1	4.624/15.271/7.099
2	7.753/14.550/10.116
3	10.626/14.900/12.405
4	12.738/14.944/13.753
5	13.916/14.915/14.398

Table 4: performance of ACT model on the test data, the numbers are shown in the following format: recall/precision/f-measure

The result after combination is shown in Fig. 1, together with the results of the previous two methods.

5 Conclusions

In this paper, we describe our tag recommendation system for the first task in the ECML PKDD Challenge 2009. We exploit two different models to recommend tags. The experimental results show that the Language model works much better than the ACT model and the combination of these two methods can improve the results.

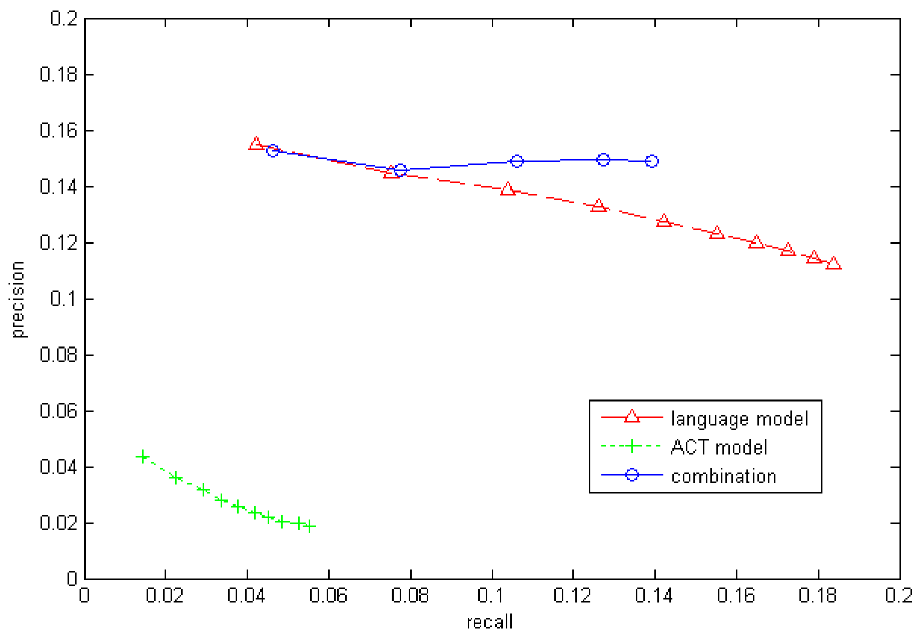

Fig.1 Recall and precision of tag recommendation system

However, we have an unexpected result of the poor ACT model, from the previous test using the separated test data from the training file, the ACT model can have a f-measure over 20%.

We need to further analyze the results to see why ACT has a poor result for the test data. Also, we can try to change the scoring scheme or expand the candidate set in the language model. Future work also includes some new methods of combination.

References

1. Marta Tatu, Munirathnam Srikanth, and Thomas D'Silva. RSDC'08: Tag Recommendations using Bookmark Content. In Proc. of ECML PKDD Discovery Challenge (RSDC 08), pp. 96-107
2. Marek Lipczak. Tag Recommendation for Folksonomies Oriented towards Individual Users. In Proc. of ECML PKDD Discovery Challenge(RSDC 08), pp. 84-95
3. Andreas Hotho, Robert Jaschke, Chiristoph Schmitz, and Gerd Stumme. BibSonomy: A Social Bookmark and Publication Sharing System. In Proc. the First Conceptual Structures Tool Interoperability Workshop at the 14th Int. Conf. on Conceptual Structures, pages 87-102, Aalborg, 2006. Aalborg Universitetsforlag.
4. Andreas Hotho, Robert Jaschke, Christoph Schmitz, and Gerd Stumme. FolkRank: A Ranking Algorithm for Folksonomies. In Proc. of FGIR 2006
5. Robert Jaschke, Leandro Marinho, Andreas Hotho, Lars Schmidt-Thieme, and Gred Stumme. Tag Recommendations in Folksonomies. In Knowledge Discovery in

Database: PKDD 2007, 11th European Conference on Principles and Practice of Knowledge Discovery in Database, Warswa, Poland, September 17-21, 2007, Proceedings, volume 4702 of LNCS, pages 506-514. Springer, 2007.

6. Mark Steyvers, Padhraic Smyth, and Thomas Griffiths. Probabilistic Author-Topic Models for Information Discovery. In Proc. of KDD'04.
7. Jie Tang, Ruoming Jin, and Jing Zhang. A Topic Modeling Approach and its Integration into the Random Walk Framework for Academic Search. In Proceeding of 2008 IEEE international Conference on Data Mining(ICDM'2008). pp. 1055-1060.
8. Sanjay C. Sood, Sara H.Owsley, Kristian J.Hammond, and Larry Birnbaum TagAssist: Automatic tag suggestion for blog posts. In Proc. the International Conference on Weblogs and Social Media(ICWSM 2007),2007.
9. Gilad Mishne. Autotag: a collaborative approach to automated tag assignment for weblog posts. In WWW'06: Proceedings of the 15th international conference on World Wide Web, pages 953-954, New York, NY, USA, 2006. ACM Press.