Automatic Tag Recommendation for Weblogs

Yicen Liu, Mingrong Liu, Xing Chen, Liang Xiang and Qing Yang
Institute of Automation, Chinese Academy of Sciences,
Beijing 100190, China,
Email: {ycliu,mrliu,xchen,lxiang,qyang}@nlpr.ia.ac.cn

***Abstract*—There have been many researches on how to recommend tags for weblogs. In this paper, we propose a novel automatic tag recommendation algorithm, which can be used in the large-scale and real-time data process effectively and efficiently. Most existing researches on tag suggestion focus on firstly mining the relationship between testing and training data and then assigning the top ranked tags of the most related training data to the testing object. However, they ignore the internal relationship between tags and weblogs. According to our research, more than 43% tags, which have been labeled by weblog users, have actually been used in the body of the text. At the meanwhile, the term frequency distribution, the paragraph frequency distribution and the first occurrence position of tags are very different from the ones of non-tags in the text. In this paper, the tags of a weblog are assigned in two steps. First of all, some probability distributions of the word attributes are trained by the labeled training weblogs, and some keywords of a testing weblog are extracted as one part of the tags based on the probability distributions. Then the other part of the tags are obtained from the first part ones with the help of Latent Semantic Indexing (LSI) model. Experiments on a large-scale tagging dataset of weblogs [1][2] show that the average tagging time for a new weblog is less than 0.02 seconds, and over 74% testing weblogs are correctly labeled with the top 15 tags.**

***Keywords*—recommendation system, data mining**

I. Introduction

With the rapid growth of the World Wide Web, how to use Internet effectively has been concerned for a long time. Tags, which are the annotations of web pages with keywords, could be used to improve the user-experience when surfing on the Internet. Recently, many Web 2.0 applications, such as Delicious[3] and Flickr[4], organize web pages with collaborative tags and have become more and more popular. At the same time, other Web 2.0 applications, like weblog services, have provided a tagging system, in which users can label their contents with some keywords. Tagging is becoming much more important than ever before. Many researches and applications have paid a lot of attention to how to recommend tags for web pages automatically and to improve the quality of other web services with the tags [1] [2].

Some researches implement their tag recommendation systems using collaborative filtering or recommender system [3] [4] [5] [6]. One basic method of these systems is the K-Nearest-Neighbor (KNN), which is based on the text vector similarity comparison. Other approaches, however, use much more complicated technology, such as graph cluster [7].

Another important aspect of tag recommendation is personalized tags recommendation [8]. Retrieving and tracking user's interest plays an important role in the tag suggestion. When a new user is added, the system firstly selects the similar users according to personal information profile. Then the tags are supplied after analyzing the personal annotations.

Almost all of the algorithms, to the best of our knowledge, are based on the core policy, i.e. finding the similar pages from the training dataset. None of the algorithms pays attention to the relationship between the tags and the web page. In this paper, we propose a novel solution for a large-scale and real-time automatic tag recommendation system, and evaluate our approach on a tagging weblog dataset, which contains 85120 weblogs collecting from Sina[1] and Sohu[2], the two biggest weblog services in China.

According to our research on these weblogs, the tags of a weblog can be divided into two parts. More than 43% of the tags appear in the body of a weblog text, and the other 56% do not. In this paper, the two parts of tags are called Tag-In and Tag-Out, respectively. Our solution is inspired from this fact, and the tags are recommended in two steps. In the system, the algorithm firstly analyzes the text of a weblog, and selects some keywords as the tags of Tag-In. At the meanwhile, the co-occurrence information of the tags between Tag-In and Tag-Out is recorded as a co-occurrence adjacency matrix, which will be used to compute the tags of Tag-Out. When the tags of Tag-In are obtained, the Tag-Out tags are suggested using Latent Semantic Indexing (LSI) model [9] [10]. Experiments show a good performance and high efficiency in our testing dataset.

The rest of the paper is organized as follows. In section 2, we introduce the related work of the tag recommendation briefly. Section 3 presents the approach to choose the tags for a weblog. Section 4 shows our experimental results. Lastly, section 5 concludes our work.

II. Related Work

Most of the exiting tag recommendation systems are originated from collaborative filtering recommender systems (CF) [11] [12], which are widely used in E-commerce [13]. In [14] and [6], CF is used in folksonomies for tag recommendations. [14] extends CF method so that a graph based recommender is applied in the folksonomy datasets. [6] describes some criteria

[1] http://blog.sina.com.cn/
[2] http://blog.sohu.com/
[3] http://del.icio.us/
[4] http://www.flickr.com/

for good tags and the categories of tags based on the attributes of themselves.

More works concern about the tag suggestion for weblogs. [3] presents a system, AutoTag, which offers a small number of tags when given a weblog post. The system finds similar tagged posts and recommends some associated tags to a user for selection. [5] uses some technology to enhance the performance of tag suggestion, and gives five different scoring parameters for tag evaluation. [4] proposes an approach to recommend tag automatically for weblogs. This solution makes use of a simple Vector Space Model (VSM) to find similar documents, and extracts the potential tags with extra information for user to select. [1] analyzes the effectiveness of tags for classifying blog entries and finds that tags are useful for grouping articles into broad categories.

Other important works pay attention on the personalized tags (P-TAG) [8] and the real-time automatic tags [7] for web pages. P-TAG produces keywords relevant both to its textual content and to the data expressing a personalized viewpoint. In [8], user interest profiling information is exploited from a very rich document corpus using a document oriented, a keyword oriented and a hybrid approach, respectively, then the tags are suggested based on the personalized web page annotations. [7] implements a real-time automatic tag recommendation system. In the system, a two bipartite graph is constructed to represent the relationship among documents, tags, and words, then Spectral Recursive Embedding (SRE) is used to partition the graph into multi-class clusters. Finally, a Poisson Mixture Model (PMM) is applied to learn the document distributions for each class. The tags are suggested based on the joint probabilities between the tags and the document.

III. Automatic Tag Recommendation

In this paper, we partition weblog tags into two parts, Tag-In (TI) and Tag-Out (TO), respectively. As the name suggests, Tag-In references the words which appear in the body text of the weblog and are chosen to be tags. It means that the tags of Tag-In are parts of the words that make up the weblog. Tag-Out, however, references the words which are selected to be tags but none of which occurs in the weblog text. It means that the tags of Tag-Out may be some abstract words that are used to summarize the content of a weblog.

Therefore, our algorithm to extract the tags of a weblog typically follows two steps. The first step is to collect the tags of TI in the context of a weblog, and the second step is to obtain the tags of TO based on the relationship between TI and TO.

A. Extracting Tags For Tag-In

In order to extract the tags of Tag-In, we have analyzed three attributes of a word, the Term Frequency (TF), the Paragraph Frequency (PF) and the 1st Occurrence Paragraph Position (1stOPP), respectively. Figure 1 shows the differences of the three attributes in our experimental dataset. According to our analysis, most of the tags have higher TF and PF than the non-tags in a weblog. It means that the tags have much more opportunity to appear in the weblog text. On the other hand, the 1stOPP of tags are mainly concentrated in the beginning and middle of a weblog context. On the contrary, most of the non-tags's 1stOPP are located in the middle and end of the weblogs.

With the help of the three attributes, it is easy to extract some keywords from the context of a weblog as the tags of TI. We propose to use Poisson distribution to estimate the distributions of the three attributes for both of tags and non-tags. Then assuming that TF, PF and 1stOPP are independent with each other, the probability whether a word is a tag can be calculated by Naive Bayes learning framework:

$$v_{NB} = \arg \max_{v_j \in \{0,1\}} P(v_j) \cdot \prod_{i=1}^{3} P(a_i|v_j) \tag{1}$$

where v_{NB} is the objective value, and the possible value is *0* and *1*, referring to *non-tag* and *tag*, respectively. $P(v_j)$ is the prior probability whether a word v_j is a tag. In this paper, we assume that $P(v_j = 0) = P(v_j = 1) = 0.5$. $P(a_i|v_j)$ is the conditional probability for each attribute a_i.

In this paper, our assumption is that all of the attributes obey Poisson distribution, then

$$P(a_i|v_j) = \frac{e^{-\widetilde{\lambda_i}} \widetilde{\lambda_i}^{a_i}}{a_i!} \tag{2}$$

and the mean of each attribute a_i in training dataset is used to estimate the parameter $\widetilde{\lambda_i}$ of Poisson distribution.

All of the words whose $v_{NB} = 1$ are the candidate tags of TI and can be ranked in descending order by their posterior probability. The top *M* ranked words are selected as the final tags of TI, where *M* is a user input value as a parameter of the recommender system, and in our system, $M = 9$.

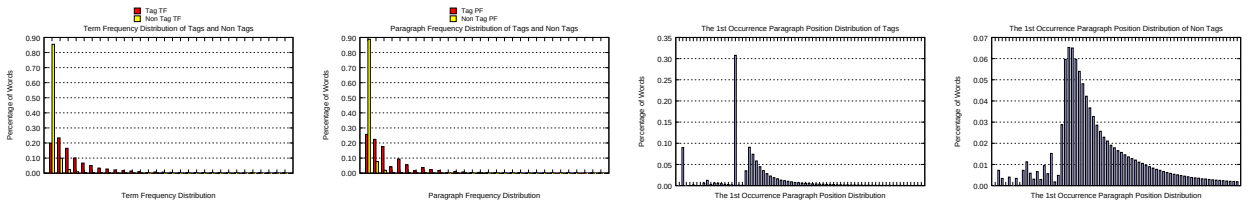
B. Extracting Tags For Tag-Out

The second step of our recommender system is to select the tags of Tag-Out. Because of the TO tag's absence in the weblog context, we have to obtain them using the relationship between TI and TO. In order to construct the relationship, the co-occurrence values of tags between TI and TO are mined in the training dataset, which means that a bipartite graph needs to be created in the training step. One part vertices of the graph represent TI and the other represent TO.

A graph $G = (V, E, W)$ is *bipartite* if it contains two vertex sets, *X* and *Y*, which satisfies that $V = X \cup Y$ and $X \cap Y = \phi$, and each edge, $e_{ij} \in E$, satisfies that one end point, *i*, belongs to *X*, and the other, *j*, belongs to *Y*. The weight of an edge e_{ij} is $w_{ij} \in W$.

In our system, the vertex set, *X*, represents the tags of TI, and the vertex set, *Y*, represents the tags of TO, and the weight w_{ij} denotes the co-occurrence value between TI and TO in the training dataset. Given the tags of TI, the weighted adjacency matrix $A_{m \times s}$ can be determined according to the bipartite graph. $A_{m \times s}$ is defined as:

$$A_{m \times s} = (a_{ij})_{m \times s} = \begin{cases} \frac{w_{ij}}{\sqrt{d_i \cdot d_j}} & \text{if i and j are adjacent,} \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$



(a) Term Frequency Distribution of Words (b) Term Paragraph Frequency Distribution of Words (c) The 1st Occurrence Paragraph Position of Tags (d) The 1st Occurrence Paragraph Position of Non Tags

Fig. 1. Attributes of Words

where m is the number of the TI tags returned from the first step. s is the total number of the vertices which are adjacent with the m TI tags. d_i is the out degree of the vertex i, and $d_i = \sum_{j \in V} w_{ij}$.

The LSI model is applied to generate the TO tags. More concretely, Singular Value Decomposition (SVD) is used on the weighted adjacency matrix $A_{m \times s}$:

$$A_{m \times s} = USV^T \tag{4}$$

where U is an m-order unitary matrix, V is an s-order unitary matrix and S is a r-order diagonal matrix that r is the rank of A. Then a rank k ($k \leq r$) approximation is implemented by keeping the first k columns of U and V and the first k columns and rows of S, represented as $U_{m \times k} = [u_1, u_2, \ldots, u_m]^T$, $V_{s \times k} = [v_1, v_2, \ldots, v_s]^T$ and $S_{k \times k} = sub_{k \times k}(S_{r \times r})$, respectively. Each row, v_i, of $V_{s \times k}$ can be used to represent the *i-th* vertex of the total s adjacent vertices.

An m-order initial query vector q, $q_i = 1, \forall 1 \leq i \leq m$, represents as the returned m tags of TI. It updates its value in the LSI model as follow:

$$q_k^T = q_m^T U_{m \times k} S_{k \times k}^{-1} \tag{5}$$

Lastly, the cosine similarity between the query vector and each adjacent vertex can be calculated as follow:

$$similarity(q_k, v_i) = \frac{q_k \bullet v_i}{\| q_k \| \cdot \| v_i \|}, \forall 1 \leq i \leq s. \tag{6}$$

All of the s adjacent vertices are considered as the candidate tags of TO and are ranked in descending order by the similarity calculated by Eq.(6). The top N ranked tags are selected as the final tags of TO, where N is a user input value as a parameter of the recommender system, and in our system, $N = 6$.

At last, totally M+N tags are recommended for a weblog, M tags of TI and N tags of TO, respectively.

IV. Experimental Results

We evaluate our algorithm by conducting the experiment in a tagging weblog dataset, all weblogs of which are crawled on two sites, Sina[1] and Sohu[2]. The dataset is divided into two parts, one for training and the other for testing. The parameters of the algorithm, M and N, are 9 and 6 respectively, so totally 15 tags will be returned.

The Vector Similarity (VS) approach is used as a baseline for comparison, which recommends the top u tags from the most similar v documents. In the experiment, both u and v are selected as 15.

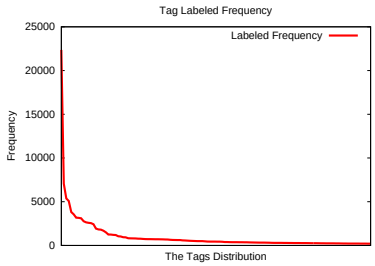


Fig. 2. Tag Frequency Distribution

A. Evaluation Metrics

We use the following metrics [7] to evaluate our algorithm.

1) Top-k accuracy: Percentage of documents correctly annotated by *at least* one of the top kth returned tags.

2) Tag-recall: Percentage of correctly recommended tags among all tags annotated by the users.

3) Tag-precision: Percentage of correctly recommended tags among all tags recommended by the algorithm.

B. Tag Recommendation

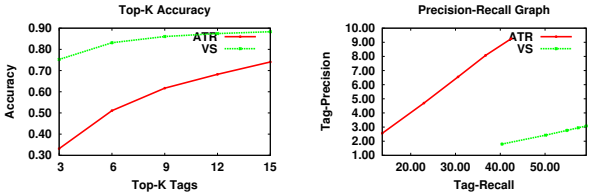
In the experimental dataset, totally 85120 weblogs are downloaded, and 62595 distinct tags are contained, 51946 of which are used in the experiment. 43.4% tags in a weblog are the words that appear in the context. Figure 2 is the statistic of the frequency that a word is labeled as a tag in the dataset no matter whether a tag occurs in a weblog. It is sorted in the descending order by the frequency. We can see that only a few words are often labeled as tags, and the highest frequency is 22361. Most of the words are labeled no more that 10 times.

Table I shows the efficiency comparison of real-time tag recommendation between our algorithm and vector similarity approach. Different proportions, from 60% to 90%, of weblogs in the dataset are used as training set. As the result shows, on average, only 0.018 seconds is needed for a testing weblog with our algorithm. On the other hand, the average tagging time of VS is 0.346 seconds. As the number of training weblogs rises, the time spent in VS is significantly increased. However, there is not any noticeable effect on the performance of our algorithm.

The reason of our high efficiency is that the algorithm does not depend on the amount of weblogs in the training set when recommending tags. In the tag recommendation stage, there are only two operations in the algorithm, one for Tag-In tags

TABLE I
TEST TIME (SECONDS)

%Train	ATR	VS
90	0.020	0.412
80	0.019	0.369
70	0.018	0.322
60	0.017	0.280
Average	0.018	0.346



(a) Top-K Accuracy (b) Tag-recall and Tag-precision

Fig. 3. Performance of Algorithm

and the other for Tag-Out ones. When extracting Tag-In tags, the time complexity is concentrated on the computation of posterior probability for each word in the weblog. It is *O(n)*, where *n* is the number of the total words in the weblog. When extracting Tag-Out tags, the time complexity is lay on the SVD of the weighted adjacency matrix, $A_{M\times S}$. It is $O(\min\{MS^2, SM^2\})$, where *M* is the number of Tag-In tags and *S* is the number of candidate Tag-Out tags. However, both *M* and *S* are much smaller than *n*. Therefore, the total time complexity of the algorithm tends to be *O(n)*.

Figure 3 shows the performance with different top-*k* selections between our algorithm and VS method. In the experiment, we use top 3, 6, 9, 12, 15 tags to evaluate the performance. As the figure 3(a) implies, we lose the test of accuracy. The accuracy of VS almost stabilizes in the interval between 70% and 90%. However, our algorithm has a poor performance when the number of tags is less than 10. Top-12 and top-15 accuracy of our algorithm are acceptable. Figure 3(b) shows the performance of Tag-recall and Tag-precision. The same as the evaluation of accuracy, our Tag-recall performance is worse than VS. The results of top 12 and 15 are better than the other selections in our algorithm. The Tag-precision of our algorithm is much higher than the ones of VS no matter how many tags are selected. The performance of our algorithm increases sharply when more tags are chosen, and VS varies gently.

The best performance of our algorithm appears when more than 10 tags are chosen. The reason is that the second step of our algorithm depends on the amount of tags returned from the first step. In the second step, the algorithm mines the relationship of tags between Tag-In and Tag-Out using SVD of the weighted adjacency matrix, and returns Tag-Out tags based on the input Tag-In tags. Therefore, the more Tag-In tags are returned, the more accurately Tag-Out tags will be calculated, and a better performance there will be.

Generally speaking, our algorithm can be used in a large-scale and real-time dataset. First of all, the training dataset can be updated incrementally. What have to do in the training step is to estimate the parameters of some probability distributions and to record the co-occurrence of tags. These operations can be easily updated when expanding training dataset. Secondly, the time spending in the tag recommendation step will not be influenced by the size of the training dataset directly, and the performance is acceptable when a certain number of tags are selected.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed an algorithm for weblog tag recommendation. The approach combined the statistical learning methods with LSI model and suggested tags for a real-word weblog collection effectively and efficiently, indicating its capability of processing large-scale real-time dataset.

Future work will focus on the situation of small amount of tags in a page, and extend our algorithm to a general text application.

REFERENCES

[1] C. H. Brooks and N. Montanez, "Improved annotation of the blogosphere via autotagging and hierarchical clustering," in *WWW '06: Proceedings of the 15th international conference on World Wide Web*, 2006, pp. 625–632.

[2] G. Begelman, P. Keller, and F. Smadja, "Automated tag clustering: Improving search and exploration in the tag space," in *Collaborative Web Tagging Workshop at WWW2006, Edinburgh, Scotland*, 2006.

[3] G. 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*, 2006, pp. 953–954.

[4] S. O. K. Lee and A. H. W. Chun, "Automatic tag recommendation for the web 2.0 blogosphere using collaborative tagging and hybrid ann semantic structures," in *ACOS'07: Proceedings of the 6th Conference on WSEAS International Conference on Applied Computer Science*, 2007, pp. 88–93.

[5] S. Sood, S. Owsley, K. Hammond, and L. Birnbaum, "Tagassist: Automatic tag suggestion for blog posts," in *Proceedings of the International Conference on Weblogs and Social Media (ICWSM 2007)*, 2007.

[6] Z. Xu, Y. Fu, J. Mao, and D. Su, "Towards the semantic web: Collaborative tag suggestions," in *Proceedings of the Collaborative Web Tagging Workshop at the WWW 2006*, 2006.

[7] Y. Song, Z. Zhuang, H. Li, Q. Zhao, J. Li, W.-C. Lee, and C. L. Giles, "Real-time automatic tag recommendation," in *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 2008, pp. 515–522.

[8] P. A. Chirita, S. Costache, W. Nejdl, and S. Handschuh, "P-tag: large scale automatic generation of personalized annotation tags for the web," in *WWW '07: Proceedings of the 16th international conference on World Wide Web*, 2007, pp. 845–854.

[9] C. H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala, "Latent semantic indexing: a probabilistic analysis," *J. Comput. Syst. Sci.*, vol. 61, no. 2, pp. 217–235, 2000.

[10] T. K. Landauer, P. W. Foltz, and D. Laham, "An introduction to latent semantic analysis," *Discourse Processes*, vol. 25, pp. 259–284, 1998.

[11] P. Resnick and H. R. Varian, "Recommender systems," *Commun. ACM*, vol. 40, no. 3, pp. 56–58, 1997.

[12] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: an open architecture for collaborative filtering of netnews," in *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, 1994, pp. 175–186.

[13] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Item-based collaborative filtering recommendation algorithms," in *WWW '01: Proceedings of the 10th international conference on World Wide Web*, 2001, pp. 285–295.

[14] R. Jäschke, L. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme, "Tag recommendations in folksonomies," 2007, pp. 506–514.