Improve Tagging Recommender System Based on Tags Semantic Similarity

Chen Hang
Department of Computer, Tianhe College of Guangdong Polytechnic Normal University, Guangzhou, 510540, China
Tianhe College of Guangdong Polytechnic Normal University, THCGDPNU
Guangzhou, China
e-mail: toboby@126.com

Zhang Meifang
Department of English, Tianhe College of Guangdong Polytechnic Normal University, Guangzhou, 510540, China
Tianhe College of Guangdong Polytechnic Normal University, THCGDPNU
Guangzhou, China
e-mail: zhangmeifang1981@163.com

Abstract— **Collaborative Filtering (CF), widely applied in such personalized recommender systems as e-business, e-library, is one of the most successful techniques to date. However, this recommender system based on traditional CF seems to refuse to consider user preference, resulting in the inaccuracy of recommendation. In view of the above limitations, we propose, in this paper, a new collaborative filtering method CFBTSS (Collaborative filtering base on tag semantic similarity). This approach tries to better understand user interest by analyzing the relevance between tags and items and by dealing with the problems of the similarity between words and similarity between sentences. Experiment results tested on MovieLens dataset show that CFBTSS significantly improved its recommending efficiency and accuracy compared to the traditional one, which contributes to the excellent performance of personalized recommendation system.**

Keywords— *recommender system; collaborative filtering; tagging system; semantic similarity*

I. INTRODUCTION

Recommender systems enable users to navigate vast collections of items. More and more companies start offering personalized recommend services toward their users. Web site such as Fab[2], Reddit, Netflix[5] and iLike have added recommender system to promote their products and services. These systems employ different approaches for recommending item to users, ranging from content based and collaborative filtering techniques to hybrid methods. Though quite useful to their users, these traditional recommendation techniques still suffer from a number of problems [1;8]. Take collaborative filtering as an example, both Bob and Tom may rate the movie Transformers with five stars, which indicates they all like this movie very much. Nevertheless, as a 3D fan, Bob appreciates this movie for its high quality 3D animations, while Tom, as an action movie fan, may think that is a wonderful action movie.

Tagging systems[7] offer users an alternate way to address the recommend task. At the same time of providing a Web-based service, it allows users to provide short-phrase description or classifications on such web resources as Web page, news, movies and photos. For Example, Del.icio.us[4] is one of the most popular collaborative tagging systems, providing easy-to-use interface for users to publish and share their favorite bookmarks along with tags. Tagging technique has become a powerful tool for semantically describing shared resource. Additionally, it can be regarded as an important way to show the comparison of content similarity between tag and item. For instance, if you tag the movie “Godfather” with “crime”, then in your opinion, its topic is about crime. That is, if two items were tagging with similar tags, the two items are similar. By taking the semantic distance between tags and items into consideration, we propose a new collaborative filtering approach to CFBTSS (Collaborative Filtering Base on Tags Semantic Similarity) to improve the tagging system.

The remainder of this paper is organized as follows: section 2 provides details on related work, mostly concerning research work in related areas, section 3 describes our method to improve tagging recommender system. In section 4, we present the results of our study. Finally, summary and future directions of research are framed in Section 5.

II. RELATED WORK

To begin, we need a conceptual model of generic tagging systems capable of being formalized in order for tagging systems to execute recommendation based on empirical data and on the model. And at present, a well-accepted one adopted here is the tripartite model which has already been theorized. In this model theory, there are three main entities that constitute tagging system:

- The users of the system (people who actually do the tagging)
- The tags themselves
- The items (movies) being tagged

Each of these can be seen as an entity forming separate space and consisting of sets of nodes, which are linked by edges (See Fig.1). The first space, *the user space*, consists of the set of all users of the tagging system, where each node is a user. The second space is the tag space, the set of all tags, where a tag corresponds to a term (“action”) or phrase (“Vietnam war”) in natural language. The third space is the item space, the set of all items, where each item is normally denoted by a unique Item id. A tagging instance can be seen as the two edges that links a user to a tag and

978-1-61284-486-2/11/$26.00 ©2011 IEEE

the tag to a given movie or item. Note that a tagging instance can associate a date of tagging with its tuple of user, tag(s), and items.

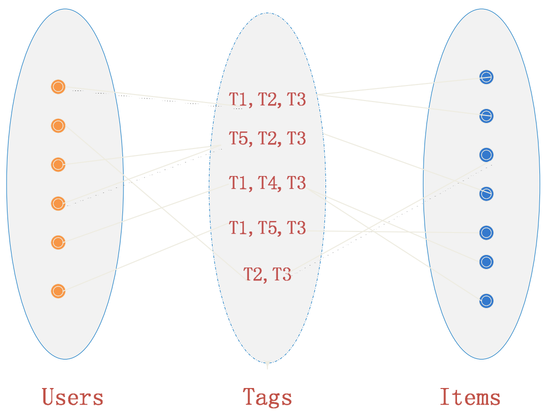

Figure 1. Tripartite graph structure of a tagging system

From the above model and Fig.1, we observe that tags provide the link between the users of the system and the items.

III. Inferring Preference

This section explores algorithms that calculate a user's preference for a tag based on her interactions with items related to the tag (Fig.2). We found that inference algorithms performed better when they took account of the relevance of a tag to an item. For example, if we wish to infer a user's preference for the tag "cars", we might treat her interactions with each of 46 movies with cars as equally important. However, cars may be more relevant to certain movies than others. For instance, cars account for 9 of the 38 tag applications for "Gone in 60 Second", but only 1 of the 36 applications for "Cast Away".

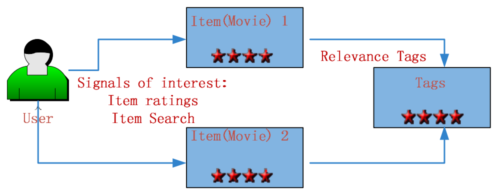

Figure 2. Inferring a user's preference for a tag based on her interactions with the items having a tag such as her rating of items with the tag.

To account for the difference of a tag's relevance, each inference algorithm in this section includes a weighting quantifying the relevance of a tag to a movie similarly to Vig et al. [11] [13]. We found that applying a sigmoid transformation improved the performance of weighting if measured by using tag quality. Suppose w(i,t) represents the relevance weighting between a item and tag:

$$w(i,\ t)=\frac{1}{e^{-tag-quality(i,t)}} \qquad (1)$$

For simplicity, we use TF-IDF to weight the quality of tags. And then we normalize w(i,t) so that the weights for each item's tag sum to 1.0.

There are six different algorithms[12] for calculating a user's preference for a tag based on her interactions with items having the tag. The six algorithms can be grouped according to the type of signal of item interest that they use. The first two algorithms (item-clicks, item-log-log-odd-clicks) use clicks on item hyperlinks as a signal of a user's interest in an item. The Third and fourth algorithms (item-r-click, item-r-log-odds-click), analyze the specific items a user choose to rate. The Last two algorithms (item-ratings, item-bayes) draw on a user's numeric rating for item. We use the item-rating algorithm. Because rating information of items can explicitly express user's preference. For example, Alice consistently rated three animated movies five stars. Item-ratings draw on this signal by predicting that a user's preference for a tag is the user's average rating for item with the tag. We draw on tag quality for the tag relevance weighting w(i,t), and $r_{u,i}$ is user u's rating for item i:

$$item\text{-}rating(u,\ t)=\frac{\sum_{i\in M_t} w(i,t)\bullet r_{u,i}}{\sum_{i\in M_t} w(i,t)} \qquad (2)$$

Here M_t is the set of all items with tag t. The sums in both the numerator and denominator ignore items the user has not rated. However, in such applications as Del.icio.us, we may not be able to get explicit rate. But we can consider that a user like an item if the user add the item in her favorites.

IV. Tag-based recommendation

We now shift our focus to use those inferred tag preference to recommend item for users. There are five tag-based recommendation algorithms-two based on implicit data, and three based on explicit data. We then describe one implicit tag-based recommendation algorithm. We call it implicit-tag methodology. The implicit-tag algorithm is inspired by algorithms from information retrieval that calculate the similarity between a user's profile vector and a document's term vector[12]. In information retrieval, the columns in each vector correspond to words. In the implicit-tag implicit algorithm, the columns correspond to tags. To generate a prediction for a item i, implicit-tag calculates the dot product between user's preference for item i's tags and the weighting w(t,i) between tag t and item m. If the ntp(u,t) is user u's normalized inferred tag preference for tag t, the user u's predicted score for item m is:

$$implicit\text{-}tag(u,\ i)=\sum_{t\in T_i} ntp(u,t)\bullet w(i,t) \qquad (3)$$

V. WordNet-based tag semantic similarity

WordNet[6] is public lexical database that provides a large repository of English lexical items. Each word in WordNet is stored in a structure called synset which is the basic unit of the whole dictionary. Every synset includes the word, its meaning and the corresponding synonyms. The meaning of a word is often referred to as gloss which actually defines the specific concept. Different meanings of a word correspond to different synset. Terms with the synonymous meanings lie in the same synset. For example, the word "love" and "passion" constitute a synset with the gloss: any object of warm affection or devotion. All the synset are organized by some basic semantic relations, such as "the part of" and "is a kind of". Therefore, the whole dictionary can be treated as a large graph with each node being a synset and edges representing the semantic relations.

As discussed above, tagging provides users with means to categorize content autonomously, independent from any central administration. However, since tagging systems do not enforce fixed or controlled vocabularies for tag selection, the free choice of tags can result in two problems. First, multiple tags can have same meanings, which are referred to as synonymy. Two tags may be morphological variation (apple vs. apples) or semantically similar (love vs. passion). Second, a single tag can have multiple meanings, which are often referred to as polysemy. For instance, an item tagged with "apple" may be post about fruits or can be interpreted as the introduction of iPod.

To solve these problems, we first adopt Porter's stemming algorithm[9] to remove the common morphological and inflexional ending of tags. Then we use Satanjeev Banerjee's algorithm [3] to get rid of the semantic ambiguity of a particular tag in certain contexts. The basic idea of this approach is to count the number of words that are shared between tow given glosses. The more common words they share, the more closely they are related.

Now, we can calculate semantic similarity between tags. We analyzed the dataset of MovieLens and found that more than 50% of tags are composed of more than one word, even a sentence. Therefore, semantic similarity between two words and two sentences should be needed to calculate.

A. Semantic similarity between two words

There are many proposals for measuring semantic similarity between two synsets: Wu & Palmer, Leacock & Chodorow, and R.Resnik. We can get the semantic similarity between two words:

$$\textit{sim-words}(W_1, W_2)=MAX\{sim(C_{1i}, C_{2j})\} \tag{4}$$

Here $sim(C_{1i}, C_{2j})$ represents semantic similarity between the ith synset of C_1 and the jth synset of C_2.

B. Semantic similarity between two sentences

We will now focus on the strategy to capture semantic similarity between two sentences. Given two sentences X and Y, we denote m to be the length of X, n to be length of Y. The major steps can be described as follows:

- Tokenization
- Perform word stemming.
- Perform part of speech tagging.
- Word sense disambiguation.
- Building a semantic similarity relative matrix R[m,n] of each pair of word senses, where R[i,j] is the semantic between the most appropriate sense of word at position i of X and the most appropriate sense of word at position j of Y. Thus, R[i,j] is also the weight of edge connecting from i to j. If a word doesn't exist in the dictionary, we output a lower associated weight; for example: an abbreviation like CTO (Chief of Technology Officer). Another solution for abbreviation is using abbreviation dictionary or abbreviation pattern reorganization rules.
- We formulate the problem of capturing semantic similarity between sentences as the problem of computing a maximum total matching weight of a bipartite graph, where X and Y are two sets of disjoint nodes. We use the Hungarian method to solve this problem.
- Then, we combine the previous calculation into a single value which will be used as value of similarity for two sentences. Though many strategies are used to acquire an overall combined similarity value for sets of matching pairs. Dice coefficient is made here:

$$sim-sentences(X,Y)=\frac{2\times|X\cap Y|}{|X+Y|} \tag{5}$$

- This strategy returns the ratio of the number of tokens that can be matched over the total of tokens. In this strategy we need to predefine a threshold to select the matching pairs that have values exceeding the giving threshold. For example, given two sentence X and Y, X and Y have lengths of 3 and 2, respectively. The bipartite matcher return that X[1] has matched Y[1] with a score of 0.8, X[2] has matched Y[2] with a score of 0.7. We set threshold to be 0.5 and employ Dice, since both matching pairs have scores greater than the threshold, so we have totally 2 matching pairs. The overall score is: 2*(1+1)/ (3+2) =0.8.

VI. Tag semantic similarity based recommendation

Now, we can predict the score for items i' that user u didn't choose or prefer to tag (If items are movies, i' is the set of movies user didn't see). In above section, we inferred user u's preference for a tag ntp(u,t), so we can easy acquire u's prefer tags set *u-pref-tags*. We predefine a threshold α,for a tag t that u applied, if ntp(u,t)> α,then add t to the *u-pref-tags* set. To the same, we also need find the most relevant tags *i-rel-tags* of each item in the i' set, we define a threshold β,if w(i,t)> β,then we can add the tag to *i-rel-tags* set. So we get two sets, one is the u's prefer tags, one is the most relevant tags of item. We can calculate the similarity between the two sets of tags, and acquire u's score for item i:

$$ss-rec(u,i)=sim-sentences(u-pref-tags,i-rel-tags) \qquad (6)$$

Here, I want to note the relation between the two sets and sentence. If we connect all the tags in one set, then we can get a string. So, we can use the formulation of semantic similarity between sentences.

VII. Experimental evaluation

We demonstrate the working of our approach on the datasets from MovieLens[10], which contains 10000054 ratings and 95580 tags applied to 10681 movies by 71567 users of the online movie recommender service. We performed two experiments to compare the performance of CFBTSS with that of the classic rating-based algorithm.

In the first experiment, the movies which previously tagged by the user are withheld. We then generate a top-N list of recommendations and record whether each withheld movie should appear or not. For each active user U_i ,I_u, represents the sets of movies that U_i has tagged. Let R_{Ui} be the set of movies that appear in the corresponding recommendation list. The average precision[13] is thus defined as:

$$\textit{Top-n-precision}=(\sum_{U_i\in U}\frac{|R_{u_i}|}{|I_{u_i}|})/|U| \qquad (7)$$

We go through the whole user space and do the same experiment on each user. Then the statistical average is calculated for all users. This experiment measures the algorithms' performance when given as much data as possible from active user. High top-n-precision corresponds to greater accuracy in the algorithm's recommendations. In our experiment, for each active user, we search for top-5 most similar neighbors and generate a top-10 recommendation list each time. The results in Fig.3, where the average ranking value reflects the average position, clearly show that CFBTSS outperforms other methods[12] with higher accuracy in recommendation.

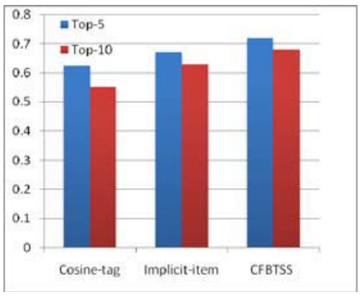

Figure 3. Compare precision between CFBTSS ,Cosine-tag, Implicit-item

In second Experiment, we report the average score of recommendation list. The higher the score is, the more satisfied with the recommendation list the user is. Like the first experiment, we also calculate the average score of the top-5 and top-10 recommendation list. The result of average score is shown in table 4(Fig.4). We claim that this feature of CFBTSS makes it possible to select better, more representative and more accurate items.

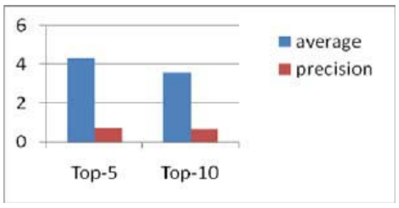

Figure 4. Average rating and precision in CFBTSS

VIII. Conclusion

Incorporating the semantic information of the tags associated with the shared resources into collaborative filtering can significantly improve predictions of a recommender system. In this paper, we have provided an effective way to achieve this goal and shown how collaborative filtering based on tag-semantic similarity outperforms the traditional collaborative filtering method. CFBTSS use the semantic similarity among the tags to significantly improve recommendation effectiveness, thus to help user to find the most favorite items, which is a critical step for collaborative recommender system and directly determines the accuracy of the final recommendation. For the future work, we seek to further improve the calculation of semantic distance. And we also need to improve the algorithm for inferring how users prefer tags and for inferring the relevant tag of the items.

Acknowledgment

On the completion of my paper, I would like to express my deepest gratitude to all those whose kindness and advice have made this work possible, I am greatly indebted to my partner Deng Shuangyi who gave me shrewd comments and quick corrections. And I am grateful to Professor He who extended great help in language in the process of fulfilling this paper.

References

[1] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, IEEE transactions on knowledge and data engineering. 17(2005) 734-749.

[2] M. Balabanovi , Y. Shoham, Fab: content-based, collaborative recommendation(1997).

[3] S. Banerjee, T. Pedersen. Extended gloss overlaps as a measure of semantic relatedness. International Joint Conference on Artificial Intelligence 18, 805-810. 2003. LAWRENCE ERLBAUM ASSOCIATES LTD.

[4] G. Begelman, P. Keller, F. Smadja. Automated Tag Clustering: Improving search and exploration in the tag space. Collaborative Web

Tagging Workshop at WWW2006, Edinburgh, Scotland . 2006. Citeseer.

[5] R.M. Bell, Y. Koren, Lessons from the Netflix prize challenge, ACM SIGKDD Explorations Newsletter. 9(2007).

[6] C. Fellbaum, WordNet: An electronic lexical database, MIT press Cambridge, MA, 1998.

[7] S. Golder, B.A. Huberman, The structure of collaborative tagging systems, Arxiv preprint cs.DL/0508082.(2005).

[8] J.L. Herlocker, J.A. Konstan, L.G. Terveen, J.T. Riedl, Evaluating collaborative filtering recommender systems, ACM Transactions on Information Systems. 22(2004) 5-53.

[9] M. Porter, The porter stemming algorithm, Accessible at http://www.tartarus.org/martin/PorterStemmer.

[10] J. Riedl, J. Konstan. Movielens dataset. 1998.

[11] S. Sen, J. Vig, J. Riedl, Learning to recognize valuable tags(2009).

[12] S. Sen, J. Vig, J. Riedl. Tagommenders: connecting users to items through tags. Proceedings of the 18th international conference on World wide web , 671-680. 2009. ACM New York, NY, USA.

[13] J. Vig, S. Sen, J. Riedl, Tagsplanations: explaining recommendations using tags(2009).