Personalized Movie Recommendation Based on Social Tagging Systems

Lin Wang
LinYi University China
wanglin@lyu.edu.cn

Keywords: Social Tagging Systems, Movie Recommendation, Similarity Calculation, Personalized Recommendation

Abstract. In the Internet era, the movie website has become a mainstream platform for movies introduction and comment on the movie resources, annotation also has become a mainstream form of movie resources under the network environment of the new organization based on social tagging.. From the system point of view, the greatest advantage offered by tagging applications is the richness of the tag profiles. However, the freedom afforded users comes at a cost: an uncontrolled vocabulary can result in tag ambiguity hindering navigation. Thus, a key question is how to harvest tag semantics from these systems. We present an algorithm of tags clustering. With this algorithm, we clustering the tags into a semantic tree, then we turn every movie item into an induced tree. We propose a new method for movie recommendation, which based on semantic similarity of tags. When we retrieve the movie based on semantic similarity of tags, our algorithm shows the high precision.

Introduction

In the Internet era, the movie website has become a mainstream platform for movies introduction and comment on the movie resources, annotation also has become a mainstream form of movie resources under the network environment of the new organization based on social tagging. This new movie resource organization mode allows users to freely according to the self-cognition tag of movie resources, with freedom, convenience and other advantages of the height, this organization is used widely in our country by the movie fan favorite watercress movie website. In practice, the use of social tags to organize movies is not favorable for no harm, because of differences in the cognitive degree of the user, so that the label inevitably normative problem, weak semantic ambiguity, synonymy and diversity in the use process, so in the flooded massive movie resources, improve retrieval efficiency of movies has become an urgent film the problem.

Movie tags cover the two dimensions of movie resources external features and content features of description, embodies the basic cognitive concept of knowledge of the user groups of movie resources sharing, and users in the social tagging system in the organization and retrieval of movie resources of the media, therefore, after refining the film label set can be used as the core term of movie set.

From the social tagging system in the extraction of movie resources high frequency tag set and the set of movies, through the label cleaning and label merging access movies refining label set and the corresponding set, and divided into the core tag set movie set and edge label set movie set two.

The key points of personalized recommendation service include: the acquisition of user interest, the matching of user interest and information category. At present, the user's personal data acquisition is mainly divided into two ways: that is, the explicit input of the user's personalized features and implicit Web mining to track the user's behavior, and automatically access the user's personalized characteristics.

Tags clustering algorithm

From the system point of view, the greatest advantage offered by tagging applications is the richness of the tag profiles.

We exploit all the tags in tagging system to cluster terms into meaningful concepts. As we shall see

later, using these concepts, it greatly improves our ability to identify similar tags and hence find similar movies.

Tags tend to express the same concept if they occur together often. The cohesion of a concept-the connections between parameters inside the concept-should be strong; the correlation between concepts-the connections between parameters in different concepts-should be weak. Strongly connected tags in between are clustered into concepts. Not clustering frequent tags is consistent with the IR community's observation that such technique leads to the best performance in automatic search expansion.

Specifically, we define the rule:

$$t_1 \rightarrow t_2(s, c)$$

In this rule, t_1 and t_2 are two terms. The support, s, is the probability that t_1 occurs in the tagging system, i.e., $s = P(t_1)$.

The confidence, c, is the probability that t_1 occurs in a movie, given that t_2 is known to occur in the same movie, i.e., $c = P(t_1 \mid t_2)$.

Each object is initialized to be a cluster of its own. In general, at each iteration the two most similar clusters are merged until no more clusters can be merged. In our context, each term is initialized to be a cluster of its own, i.e., there are as many clusters as terms. The algorithm proceeds in a greedy fashion. It sorts the association rules in descending order first by the confidence and then by the support. Infrequent rules with less than minimum support t_s are discarded. At every step, the algorithm chooses the highest ranked rule that has not been considered previously. If the two terms in the rule belong to different clusters, the algorithm merges the clusters. Formally, the condition that triggers merging cluster *I* and *J* is

$$\exists i \in I, j \in J, i \rightarrow j(s > t_s, c > t_c)$$

where *i* and *j* are terms. The threshold t_s is chosen to control the clustering of terms that do not occur frequently.

To ensure that we obtain clusters with high cohesion, we merge two clusters only if they satisfy a stricter condition, called cohesion condition. Given a cluster C, a term is called a kernel term if it is closely associated with at least half of the remaining terms in C. Our cohesion condition requires that all the terms in the merged cluster be kernel terms. Formally, we merge two clusters *I* and *J* only if they satisfy the cohesion condition:

$$\forall i \in I \cup J, \left\|\{j \mid j \in I \cup J, i \neq j, i \rightarrow j(c > t_c)\}\right\| \geq 0.5(\|I\| + \|J\| - 1)$$

where $\|I\|$ is the number of terms in cluster *I*.

Algorithm is as follows:

```
procedure MergeKeywords(T,R)
//T is the term set, R is the association rule set
for(i=1 to n) Ci={ti}; //initiate clusters
sort(R);
//sort R first by the descending order of confidence,
//then by the descending order of support value
for each(r: t1 → t2(s > ts, C > tc) in R)
  if(t1 ∈ I, t2 ∈ J, I ≠ J)
    if(‖{j | j ∈ I ∪ J, i ≠ j, i → j(c > tc)}‖ ≥ 0.5(‖I‖ + ‖J‖ − 1))
      Merge(I,J);
end for
return result clusters;
```

Movie characterization

With the above algorithm, we get $C = \{c_1, c_2, \mathbf{K}, c_i, \mathbf{K}\}$. Then, we adopt a threshold $t_c' < t_c$ and run the clustering algorithm repeatedly. At each step, we adopt a smaller threshold, re-collect association rules, and then re-run the clustering algorithm. This process continues until finally one cluster is generated.

In the above process, a concept tree is generated as shown in Fig.1.

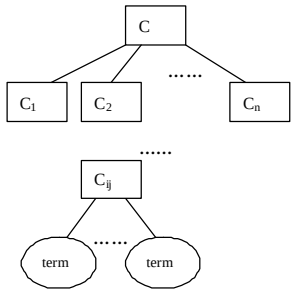

Figure 1. An example of concept tree

Then, each movie is represented as a sub-tree of the above concept tree, tags of movie are corresponding to leaf nodes of the sub-tree. Thus, for movie which can be represented by tag vector (t_3, t_4, t_{11}, t_{30}, t_{39}), the corresponding tree representation is shown in Fig.2.

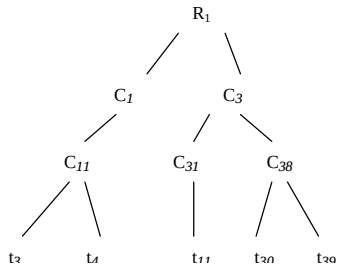

Figure 2. Tree representation of tags vector (t_3, t_4, t_{11}, t_{30}, t_{39})

Personalized Movie Recommendation

In cosine similarity computing methods, the unit vectors of all leaf nodes are vertical to each other, among which the dot matrix result of any two unit vectors is 0 and 1 if the two vectors are the same. There are some great defects of the method, for example, the similarity computed using this method between movies $M_1(t_1, t_2, t_3)$ and $M_2(t_4, t_5, t_6, t_7)$ is 0, but the two pieces of movie have great similarity, which we don't wish $sim(R_i, R_j)$ equals to 0.

Here we describe a new method for computing similarities of movies.

The depth of node t_i, $depth(ti)$ can be defined as the length from root node to this node in the tree, that is, the amounts of the total edges. The height of the tree is represented with h. To two leaf nodes t_i and t_j of tree U, $LCA(t_i, t_j)$ represents their smallest common ancestor. Two leaf nodes have at least a common ancestor node — root node. In figure2, $LCA(t_1, t_7) = C_{11}, LCA(t_1, t_8) = C_1$.

The unit vector of the leaf node t is defined as $\vec{t}$ and we don't wish $\vec{t}_1 \cdot \vec{t}_2$ equals to 0, therefore in this model definition $\vec{t}_1$ and $\vec{t}_2$ are not vertical to each other. Considering the hierarchical structure,

we make $\vec{t}_1 \cdot \vec{t}_2 = \frac{2}{3}$, because the distance from root node to their common ancestor occupies $\frac{2}{3}$ of the total distance. In the similar way, $\vec{t}_1 \cdot \vec{t}_8 = \frac{1}{3}$, while $\vec{t}_1 \cdot \vec{t}_9 = 0$.

To any two leaf nodes, we define $\vec{t}_i \cdot \vec{t}_j = \frac{depth(LCA_T(t_i, t_j))}{h}$, whose value is continuous, range from 0 to 1. When $LCA(t_i, t_j)$ is the root node, its value is 0 and 1 when $t_i = t_j$.

The vectors corresponding to movie A and B are represented as $\vec{A} = \sum_i \vec{t}_i$ and $\vec{B} = \sum_j \vec{t}_j$. Their dot matrix is defined as $\vec{A} \cdot \vec{B} = \sum_{i=1}^{n} \sum_{j=1}^{n} \vec{t}_i \cdot \vec{t}_j$ and the similarity between A and B is defined as $sim(\vec{A} \cdot \vec{B}) = \frac{\vec{A} \cdot \vec{B}}{\sqrt{\vec{A} \cdot \vec{A}} \sqrt{\vec{B} \cdot \vec{B}}}$.

To the movie user selected, the system intensively calculates the similarity of every one, ordering by their similarity and returns movies whose $sim > t_{sim}$.

The clustering algorithm applied in this paper is suitable for movie recommendation.

References

[1] J. Bar-Ilan, S. Shoham, A. Idan, Y. Miller, and A. Shachak.Structured vs. unstructured tagging—a case study. In Proc.WWW Collaborative Web Tagging Workshop, 2006.

[2] G. Begelman, P. Keller, and F. Smadja. Automated Tag Clustering: Improving search and exploration in the tagspace. Proc. of the Collaborative Web TaggingWorkshop at WWW, 6, 2006.

[3] H. Chen and S. Dumais. Bringing order to the Web: automatically categorizing search results. Proceedings of theSIGCHI conference on Human factors in computing systems,pages 145–152, 2000.

[4] J. Gemmell, A. Shepitsen, B. Mobasher, and R. Burke.Personalization in Folksonomies Based on Tag Clustering.Intelligent Techniques for Web Personalization &Recommender Systems, 2008.

[5] J. Gemmell, A. Shepitsen, B. Mobasher, and R. Burke.Personalizing navigation in folksonomies using hierarchical tag clustering. Proceedings of the 10th International Conference on Data Warehousing and Knowledge Discovery, 2008.

[6] T. Hammond, T. Hannay, B. Lund, and J. Scott. Social Bookmarking Tools (I). D-Lib Magazine, 11(4):1082–9873, 2005.

[7] C. Hayes and P. Avesani. Using tags and clustering to identify topic-relevant blogs. International Conference on Weblogs and SocialMedia, March 2007.

[8] P. Heymann and H. Garcia-Molina. Collaborative Creation of Communal Hierarchical Taxonomies in Social Tagging Systems. Technical report, Technical Report 2006-10, Computer Science Department, April 2006.

[9] S. Bao, G. Xue, X. Wu, Y. Yu, B. Fei, and Z. Su. Optimizing web search using social annotations. In Proc. 16th Intl. Conf. on World Wide Web, pages 501–510, 2007.

[10] G. Begelman, P. Keller, and F. Smadja. Automated tag clustering: Improving search and exploration in the tag space. In Proc. WWW Collaborative Web Tagging Workshop, 2006.

[11] P. Boldi, M. Santini, and S. Vigna. Do your worst to make the best: Paradoxical effects in pagerank incremental computations. Internet Mathematics, 2(3):387–404, 2005.

[12] S. Brin and L. Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. Computer Networks and ISDN Systems, 30(1-7):107–117, April 1998.