Mining User Preferences for Recommendation: A Competition Perspective

Shaowei Jiang, Xiaojie Wang, Caixia Yuan, and Wenfeng Li

Center for Intelligence Science and Technology,
Beijing University of Posts and Telecommunications, Beijing, China
{shaowei.jiang,lwfeng115}@gmail.com, {xjwang,yuancx}@bupt.edu.cn

Abstract. Mining user preferences plays an important role in building personalized recommender systems. Instead of mining user preferences with the item content or the user-item-rating matrix, we exploit Bradley-Terry model to mine user preferences as pairwise comparisons. In this paper we assume that the user preference on each item can be represented by the combination of different content features, which brings a direct bridge between features and user preferences. Experimental results show that the method based on pairwise comparisons outperforms baseline approaches with less recommendation time.

Keywords: user preference, pairwise comparison, Bradley-Terry model, recommender system.

1   Introduction

Recent years have witnessed the unprecedented prevalence and significance of building recommender systems, which aim to recommend proper items for users with respect to their personal preferences. On the Internet, there are a large number of items with ratings of particular users such as books, movies and food. The ratings could effectively reflect user preferences on the items, and become important resources to mine user preferences and implement recommendations.

Collaborative filtering (CF) and content-based recommendation (CBR) are two commonly used approaches for recommendation. CF analyzes the similarity of users' preferences or items and recommends items for active users, using data rated by a great quantity of users. However, since it does not directly analyze the content of items, it may suffer from the so called cold start problem, which makes it fail to recommend items that have not been previously rated in the community [1]. Meanwhile, the CBR approach focuses on analyzing the similarity between item contents and user preferences, so it can process new items without user ratings. Traditional CBR method is generally divided into three steps: Firstly, it confirms item representation through extracting content features. Secondly, it mines the user preference and represents the preference with features by leveraging the past rating information. Finally, it recommends the most similar item set to the user, by comparing the user preference representation and the representation of the new item [2]. However, the CBR approaches

always excessively rely on the results of content analysis, which cannot take full advantage of the explicit rating data.

In this paper, we propose a novel approach for mining user preferences by exploiting the competition perspective from game theory, which sufficiently makes use of both the content of items and user ratings. Unlike previous works, which each level rating is considered as independent category, this paper discusses to mine user preferences with competition relationship of items rated different ratings. It is assumed that each item consists of several content features. Ratings on items are used to estimate a unique user preference value for each feature by Bradley-Terry model. The rating for a new item can therefore be predicted by user preference values of all features in the item. The basic idea is: a user U gives a higher rating to item A than that of B for the reason that A is better than B in the competition of U's preferences. If A and B are represented by textual features, then it is supposed that features of A have overall stronger competitiveness than those of B. Therefore, the competition between two items with various ratings is decomposed into competitions among content features of two items. The user preference value of each feature could be estimated, which is obtained by competitions for several times. Thus, when a new text C turns up, the user preference value for C can be calculated through summing up the value of each feature in C. In this way, this model can integrate both user ratings and the content of items for mining user preferences.

The remainder of this paper is organized as follows. In Section 2 we review related work. The user preferences mining approach based on pairwise comparisons is presented in Section 3. Section 4 reports experiments and results. Finally, Section 5 summarizes our conclusions and discusses directions of future work.

2   Related Work

Generally, the related works of this paper can be grouped into three categories. The first is the CBR. CBR algorithms usually apply vector space model (VSM) to represent items. Recommendation results are normally obtained by machine learning algorithms, such as the nearest neighbor method (find K rated items most similar with the new item, use the user preference of K items to judge the preference of the new item), Rocchio algorithm (obtain user preference vector to features from liked and disliked items, calculate the similarity of the vector and the feature value vector of the new item as user preference of the new item), the decision tree algorithm (build the tree structure to use its branch to classify the new item with features), the linear classifier algorithm (find a plane in the higher dimensional space to separate class points) and the Naive Bayes algorithm (calculate probabilities of features in all categories to predict the new item). But these methods are too dependent to the content, with rating information as an auxiliary tool.

The second is the CF recommendation. CF algorithms can be divided into memory-based and model-based algorithms. The user-based and item-based algorithms are common memory-based methods. The user-based algorithm calculates the similarity between the active user and other users, then recommends

favorite items of users who are very similar to the active user [3]. The item-based algorithm analyzes items rated by the active user, and recommends items that are similar to ones that he rated [4]. In order to avoid overfitting phenomenon from the item-based algorithm, Lemire and Maclachlan [5] proposed a slope one algorithm. In terms of model-based algorithms, to reduce the dimension of the user-item-rating matrix, latent semantic models such as probabilistic latent semantic analysis, Latent Dirichlet allocation, the Single Value Decomposition (SVD) algorithm and its follow-up improvement algorithms appeared [6-11]. Yang et al [12] studied user choice behavior in a series of items under the background of CF, obtaining better recommendation effect. No matter what kind of CF algorithms, there is always the cold start problem.

The third is hybrid recommendation. It could combine CBR and CF recommendation results, and make feature augmentation, such as using content features to offset users' simple ratings, or training users' ages and genres of movies in a classifier [13-16], and so on. Although there are many combination methods, they are not effective in a specific issue all the time.

This article proposes a new way to estimate preference values of micro content features from ratings data of items, basing on a pairwise comparison model. Preference values of features are then used to rate new items.

3   The User Preferences Mining Approach Based on Pairwise Comparisons for Recommendation

3.1   Motivation Discussion

Given a user u and an item set $\overline{I} = \{1, 2, \dots , I\}$, $r(u, i)$ denotes the rating for an item $i \in \overline{I}$ by u. To solve the problem of rating a new item $i_a \notin \overline{I}$, the user preference value of u on i is introduced, denoted by $p(u, i)$. For any two items i_A and i_B, if $r(u, i_A)$ is similar with $r(u, i_B)$, then $p(u, i_A)$ is assumed to be similar with $p(u, i_B)$. The higher the rating is, the stronger the preference is, and vice versa. For example, items with rating 5 are supposed to have similar preferences, and the preference for the item rated with 5 is stronger than the one for the item rated with 4.

Now we have $p(u, i)$ as the preference value of a user u on an item i. Considering an item consists of some content features, it is therefore assumed that preference value of u on i correspondingly consists of preference values of u on features in i. Given a feature set $\overline{F} = \{1, 2, \dots , F\}$ for all items, if i consists of feature subset $F_i \subset \overline{F}$ and each feature $f_i \in F_i$ is independent, $p(u, i)$ is determined by combination of $\{p(u, f_i)\}$. So, for a new item i_a, its preference value can be calculated by $\{p(u, f_{ia})\}$.

There are three stages of our model. First, $p(u, f)$, $f \in \overline{F}$ for all features are estimated. Second, $p(u, i)$ of an item i is calculated. Finally, $r(u, i)$ is acquired basing on $p(u, i)$. The details are given in 3.2, where Bradley-Terry model is used for $p(u, f)$ estimation and $p(u, i)$ calculation. The nearest neighbor method (KNN) is used to get $r(u, i)$ from $p(u, i)$.

3.2   Model Description

Bradley-Terry model is first given for $p(u, f)$ estimation and $p(u, i)$ calculation, and then KNN is used for getting $r(u, i)$ from $p(u, i)$.

Out of personal interest, a user has various ratings to different items. In turn, items have matches on the level of user interest, resulting in the survival of the fittest. There is obvious competitive relationship among them. For the comparison between things, the Bradley-Terry (BT) model [17] is a popular competitive relationship probability model. This model is used to measure the ability of competitive object in the pairwise comparison. It assumes that the contestant's win rate is proportional to his own competitiveness. And it trains and quantifies the capacity of the contestant basing on the assumption. The competitor's winning probability is described by the original BT model in the pairwise comparison. It is shown in Formula (1).

$$P(o \text{ beats } q) = \frac{\gamma_o}{\gamma_o + \gamma_q} \, , \tag{1}$$

where γ_o refers to overall competition level of the individual o. Different from an individual, the content of a item consists of multiple features. So this article makes use of the generalized BT model for multi-player team competition. It is illustrated by Formula (2).

$$P(1\text{-}2\text{-}3 \text{ wins against } 4\text{-}5 \text{ and } 2\text{-}6\text{-}7) = \frac{\gamma_1\gamma_2\gamma_3}{\gamma_1\gamma_2\gamma_3 + \gamma_4\gamma_5 + \gamma_2\gamma_6\gamma_7} \, . \tag{2}$$

Note one feature can appear in multiple items here. Through competitions among items for many times, the competitiveness of each feature may be ultimately determined, that is, the user preference value.

As there are all competitive relations among items got different ratings, any two items with various ratings can play a game. The content describing the item can be analogous to a team participating in a competition, and features acted like players. The content of the item that is higher rated by the user is the winning team in the game. During the competitions, the user preference values of features in the winning team increase, meanwhile the values of features in the losing team correspondingly reduce. In the end, through iterative calculation of all sessions, user preference values of features settle out.

The estimation process of $\{p(u, f)\}$ is described as follows. First of all, we denote user preference values of n features with parameters $p(u, f_1)$, $p(u, f_2)$, ... , $p(u, f_n)$. $p(u, i_1)$, $p(u, i_2)$, ..., $p(u, i_K)$ are user preference values of K items in a competition. If an item i consists of M features, $p(u, i)$ could be computed by Formula (3).

$$p(u, i) = \prod_{m=1}^{M} p(u, f_m) \, . \tag{3}$$

In other words, the preference value of each item is the product of features' values, following the assumption of the generalized BT model [18]. The N independent match results among contents, R_1, R_2, ..., R_N, are get from the corpus. So the probability of the game j result can be written as Formula (4).

$$P(R_j) = \frac{p(u, i_{win})}{\sum_{k=1}^{K} p(u, i_k)} \; . \tag{4}$$

The objective of the model is to maximize the probability of these games' results, and the objective function $\boldsymbol{L}$ is shown by Formula (5).

$$L = \prod_{j=1}^{N} P(R_j) \; . \tag{5}$$

$\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{f_w})$ could be iteratively calculated, by setting the feature $\boldsymbol{w}$ as the only variable in one iteration. Thus Formula (4) could be rewritten as Formula (6).

$$P(R_j) = \frac{f(p(u, f_w))}{g(p(u, f_w))} \; . \tag{6}$$

$\boldsymbol{f}(\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{f_w}))$ and $\boldsymbol{g}(\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{f_w}))$ have $\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{f_w})$ as a variable and other $\{\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{f_m})\}$ as constants. So the logarithm of $\boldsymbol{L}$ is shown in Formula (7).

$$\log L(p(u, f_w)) = \sum_{j=1}^{N} \log f(p(u, f_w)) - \sum_{j=1}^{N} \log g(p(u, f_w)) \; . \tag{7}$$

Applying the Minorization-Maximization algorithm [19], we start to build the approximation function $\boldsymbol{m}$ at an initial point $\boldsymbol{p^0}(\boldsymbol{u}, \boldsymbol{f})$ to make $\boldsymbol{m}(\boldsymbol{p^0}(\boldsymbol{u}, \boldsymbol{f})) = \boldsymbol{\log L}(\boldsymbol{p^0}(\boldsymbol{u}, \boldsymbol{f}))$ and $\boldsymbol{m}(\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{f})) \leq \boldsymbol{\log L}(\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{f}))$ for $\{\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{f})\}$. Then the next point to be computed is found, at which the approximation function value is max. As $\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{f_w})$ either occurs in winners or losers, we set $\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{f_w})$ as $\boldsymbol{x}$, and omit constants to get the minorizing function (8)

$$m(x) = WN_w \log x - \sum_{j=1}^{N} \frac{g'(x)x}{g(x)} \; , \tag{8}$$

$\boldsymbol{WN_w}$ is the sum of occurrence number of $\boldsymbol{w}$ in all winning teams. $\boldsymbol{g}(\boldsymbol{x})$ is the total preference value of all teams in the game. With the thought that maximizes $\boldsymbol{m}(\boldsymbol{x})$, the iterative formula (9) of $\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{f_w})$ is

$$p(u, f_w)^{(k+1)} \leftarrow \frac{WN_w}{\sum_{j=1}^{N} \frac{g'(p(u, f_w)^{(k)})}{g(p(u, f_w)^{(k)})}} \; . \tag{9}$$

All of the preference values are confirmed when $\boldsymbol{L}$ is maximized.

After the above steps, $\{\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{f})\}$ are known. The user preference value $\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{i_a})$ of the new item $\boldsymbol{i_a}$ is calculated by Formula (3). If $\boldsymbol{f_{ia}} \notin \overline{\boldsymbol{F}}$, $\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{f_{ia}})$ is set as 1. According to our first assumption, $\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{i_a})$ can predict $\boldsymbol{r}(\boldsymbol{u}, \boldsymbol{i_a})$, which could be achieved by KNN easily. It is to find $\boldsymbol{k}$ items $\boldsymbol{I_k} \subset \overline{\boldsymbol{I}}$ of which $\{\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{i_k}), \boldsymbol{i_k} \in \boldsymbol{I_k}\}$ (also calculated by Formula (3)) are nearest to $\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{i_a})$. As $\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{i})$ is a scale value, the absolute value of difference between $\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{i_A})$ and $\boldsymbol{p}(\boldsymbol{u}, \boldsymbol{i_B})$ is used as the similarity measure. So the similarity formula (10) is

$$Sim(i_A, i_B) = |p(u, i_A) - p(u, i_B)| \; . \tag{10}$$

Then $\boldsymbol{r}(\boldsymbol{u}, \boldsymbol{i_a})$ is set as the most frequent $\boldsymbol{r}(\boldsymbol{u}, \boldsymbol{i_k})$ in $\boldsymbol{I_k}$.

4   Experiments

Firstly, prediction precision based on CBR and CF recommendation algorithms are compared with our method on two movie datasets. Secondly, the relationship between competitive scales and recommendation effect is discussed.

4.1   Recommendation Effect Experiments

The algorithms are evaluated on two datasets, MovieLens and Netflix [20-21], which are the most famous datasets in the field of recommender system. The MovieLens dataset is the real data crawled by the MovieLens movie recommender system (`http://movielens.umn.edu`). It contains rating data scored by numbers from 1 to 5 which contains 943 unique users' 10,000 ratings to 1,682 movies. The data of 600 users who rate movies more than 45 times are selected in the experiment. The content of each movie consists of director names, major movie star names and film styles (such as action, sci-fi, etc.), which are crawled from the IMDB website (`http://www.imdb.com`). The feature set includes all words in the content of items. The Netflix dataset is the Netflix Prize data, 1 to 5 rating data of about 10 billion times rating from 480,189 anonymous users on about 17,770 movies. About 100,000 users are randomly selected as experimental samples, with the same setting of movie content and source as the MovieLens dataset. On account of the large amount of data in the Netflix dataset, there are lots of internal competition relations. To train parameters faster, the training part of the competitive method is calculated by parallel processing in Beijing Computing Center (`http://www.bcc.ac.cn`).

Mean absolute error (MAE) and root mean squared error (RMSE) are used to evaluate the recommendation performance.

MAE. It calculates the average difference between prediction ratings and actual ratings in the test set.

$$|\overline{E}| = \frac{\sum_{i=1}^{N} |p_i - v_i|}{N} \ , \tag{11}$$

p_i is the prediction rating of a test sample, v_i is the actual rating, N is the number of test samples.

RMSE. It represents the deviation of the prediction rating than the actual rating, more emphasizing on large errors. The formula is as follows.

$$|\overline{E}| = \sqrt{\frac{\sum_{i=1}^{N} (p_i - v_i)^2}{N}} \ . \tag{12}$$

The baseline approaches are a CBR algorithm and RSVD in CF methods. The CBR recommendation algorithm uses term frequency-inverse document frequency (TF-IDF) values generated from VSM as content feature values, and then makes use of KNN to predict ratings. So it is named TFIDF-KNN here. It also includes all words in its feature set. Our method names BT-KNN. Three methods are evaluated on average errors of predict ratings by 5 cross-validation.

Table 1. Recommendation effect of methods on the MovieLens dataset

MAE	K=1	K=5	K=10	K=20	K=50	K=100	K=200	K=500
BT-KNN	0.819	0.821	0.822	0.822	0.822	**0.819**	0.819	0.819
TFIDF-KNN	0.978	0.920	0.887	0.866	0.852	**0.848**	0.848	0.848
RSVD	**1.131**							
RMSE	K=1	K=5	K=10	K=20	K=50	K=100	K=200	K=500
BT-KNN	1.110	1.110	1.109	1.108	1.106	**1.105**	1.105	1.105
TFIDF-KNN	1.316	1.251	1.217	1.192	1.176	1.172	**1.171**	1.171
RSVD	**1.378**							

Table 2. Recommendation effect of methods on the Netflix dataset

MAE	K=1	K=5	K=10	K=20	K=50	K=100	K=200	K=500
BT-KNN	**0.755**	0.758	0.761	0.764	0.769	0.769	0.769	0.770
TFIDF-KNN	0.911	0.848	0.817	0.795	0.776	0.769	**0.766**	0.766
RSVD	**0.905**							
RMSE	K=1	K=5	K=10	K=20	K=50	K=100	K=200	K=500
BT-KNN	1.040	**1.040**	1.042	1.045	1.048	1.049	1.049	1.050
TFIDF-KNN	1.251	1.182	1.148	1.124	1.103	1.095	1.092	**1.091**
RSVD	**1.109**							

Since BT-KNN uses independent personal data, the metrics are finally per-user average values. The experimental results are shown in Table 1 and Table 2.

As can be seen from the tables, the recommendation effect of BT-KNN is relatively stable, both on the MovieLens dataset and the Netflix dataset. It achieves good performance even at small K, and is better than TFIDF-KNN significantly at the same time. The predictive rating error of TFIDF-KNN gradually decreases along with the increase of the neighbor number, and stabilizes at a big K. When $K = 1$, BT-KNN improves the MAE performance of TFIDF-KNN by 19.4% on the MovieLens dataset and by 20.7% on the Netflix dataset. BT-KNN improves the RMSE performance of TFIDF-KNN by 18.6% on the MovieLens dataset and by 20.3% on the Netflix dataset. About the methods' best recommendation effect, comparing to TFIDF-KNN, MAE of BT-KNN is increased by 3.5% on the MovieLens dataset and by 1.5% on the Netflix dataset, and RMSE is increased by 6.0% on the MovieLens dataset and 4.9% on the Netflix dataset. Compared to RSVD, BT-KNN has increased MAE by 38.1% and RMSE by 24.7% on the MovieLens dataset, while MAE by 19.9% and RMSE by 6.6% on the Netflix dataset.

BT-KNN also has the advantage on the recommendation time. The comparisons on time are in Table 3.

Table 3. Average recommendation time of methods

Avg. Time (ms)	BT-KNN	TFIDF-KNN	RSVD
MovieLens	1.040	7.739	0.141
Netflix	3.578	111.525	64.397

From the experimental results, the pairwise comparisons based method has higher predictive accuracy and stronger robustness. It reduces prediction impacts of the case brought by CBR, which items' content vectors are similar with inconsistent user preferences. CF focuses on decreasing the overall rating matrix error, so the adaptability of predicting personalized ratings is not strong. And competitive approach is faster than others in recommending.

4.2   The Relationship between Competitive Scales and Recommendation Effect

The quantity of ratings reflects the competitive scale. The fewer ratings are, the lower competition scale is. For example, if the rating scale is only two indicated by 'good' and 'bad', there is only one type of competitive relationship. For 5 ratings scale, every two level rating items have competitive relationships to each other.

To understand the characteristics of the user preferences mining approach based on pairwise comparisons, the recommendation effect of the method is researched under different competitive scales. Using the MovieLens dataset, we simulated 2 ratings data with $\{\{1, 2, 3\}, \{4, 5\}\}$ (original ratings), as well as 3 ratings data with $\{\{1, 2\}, \{3\}, \{4, 5\}\}$. Combined with the previous 5 ratings data, there are 3 different kinds of competitive scale data. Improved rates of BT-KNN with respect to TFIDF-KNN are observed in 3 cases. The relationship between competitive scale and recommendation effect is discovered. Recommendation effect of BT-KNN and TFIDF-KNN are demonstrated in Fig. 1 and Fig. 2, using 2 ratings data and 3 ratings data.
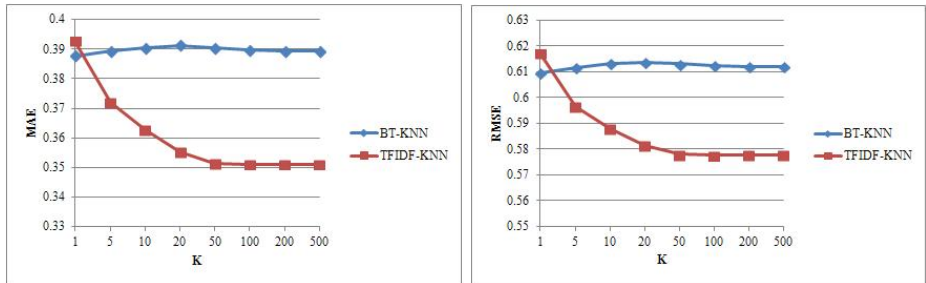

Fig. 1. The comparison of BT-KNN and TFIDF-KNN on the 2 ratings dataset

According to the above experimental results, recommendation effect relative variations of BT-KNN can be obtained in 3 kinds of competitive scale data. Fig. 3 shows them as follows.

It could be seen that recommendation effect of BT-KNN is worse than TFIDF-KNN on 2 ratings data under the condition of a small competitive scale. But as the increasing of the competitive scale, effect of BT-KNN is better than TFIDF-KNN on both 3 and 5 ratings data, and it is more obvious on 5 ratings data.
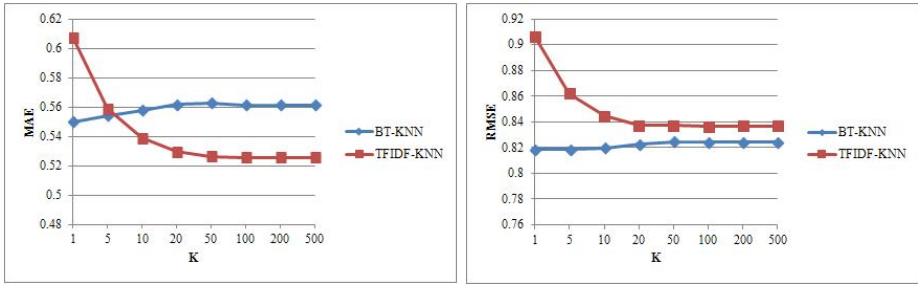
Fig. 2. The comparison of BT-KNN and TFIDF-KNN on the 3 ratings dataset
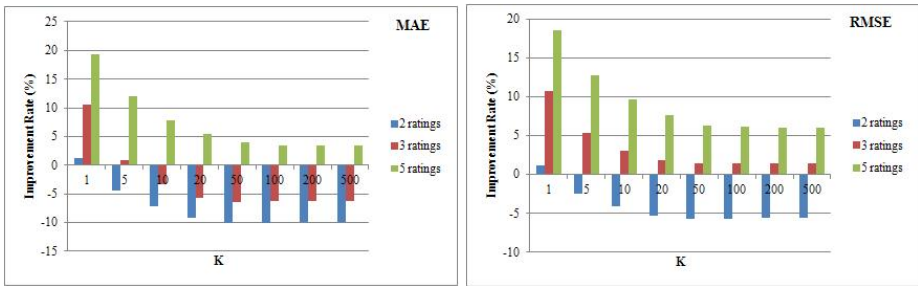

Fig. 3. The improved rates of BT-KNN contrasting with TFIDF-KNN on metrics

The result illustrates the method's recommendation effect is better on the larger competition scale.

5   Conclusions

This article describes the basic idea of the user preferences mining approach based on pairwise comparisons. It unites the content and user rating of an item together with the competitive relationship, and obtains a unique identification of the user preference on a feature. The method provides better recommendation accuracy and less recommended time. And it demonstrates that the larger competitive scales of data, the better recommendation effect.

Currently, when the same feature is in different items, each feature is assumed to have just a user preference value, regardless of the influence of its context on its user preference. In the future, to use the binary features with the context will be investigated to verify the reasonableness of the assumption. And the overall value is the product of its members' values, which is the hypothesis of the generalized BT model, is rather simple. Hereafter, the composite mode adapted data will be studied, to improve the recommendation performance of the approach.

References

1. Cacheda, F., Carneiro, V., Fernndez, D., Formoso, V.: Comparison of collaborative Filtering algorithms: limitations of current techniques and proposals for scalable, high-performance recommender systems. ACM Transactions on The Web (TWEB) 5(1), 1–33 (2011)
2. Lops, P., Gemmis, M.D., Semeraro, G.: Content-based recommender systems: State of the art and trends. In: Ricci, F., Rokach, L., Shapira, B., Rokach, L., Kantor, P.B. (eds.) Recommender Systems Handbook, pp. 73–105. Springer, New York (2011)
3. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: an open architecture for collaborative filtering of netnews. In: The 1994 ACM Conference on Computer Supported Cooperative Work, pp. 175–186. ACM Press, New York (1994)
4. Sarwar, B., Karypis, G., Konstan, J., Reidl, J.: Item-based collaborative filtering recommendation algorithms. In: Proc. of the WWW 2001, pp. 285–295. ACM Press, New York (2001)
5. Lemire, D., Maclachlan, A.: Slope one predictors for online rating-based collaborative filtering. In: SIAM Data Mining Conference, SDM 2005 (2005)
6. Hofmann, T.: Probabilistic Latent Semantic Analysis. In: Proc. of 15th Conf. Uncertainty in Artificial Intelligence, pp. 289–296 (1999)
7. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet Allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
8. Sarwar, B., Karypis, G., Konstan, J., Reidl, J.: Application of dimensionality reduction in recommender systems-a case study. In: The ACM WebKDD Workshop (2000)
9. Funk, S.: Netflix update: Try this at home,
   `http://sifter.org/~simon/journal/20061211.html`
10. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proc. of the KDD 2008, pp. 426–434. ACM Press, New York (2008)
11. Koren, Y.: Collaborative Filtering with Temporal Dynamics. In: Proc. of the KDD 2009, pp. 447–456. ACM Press, New York (2009)
12. Yang, S.H., Long, B., Smola, A.J., Zha, H., Zheng, Z.: Collaborative competitive filtering: learning recommender using context of user choice. In: Proc. of the SIGIR 2011, pp. 295–304. ACM Press, New York (2011)
13. Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., Sartin, M.: Combining content-based and collaborative filters in an online newspaper. In: The ACM SIGIR 1999 Workshop Recommender Systems: Algorithms and Evaluation. ACM Press, New York (1999)
14. Pazzani, M.J.: A framework for collaborative, content-based, and demographic filtering. Artificial Intelligence Review 13(5-6), 393–408 (1999)
15. Melville, P., Mooney, R.J., Nagarajan, R.: Content-Boosted collaborative filtering for improved recommendations. In: Proc. of the AAAI 2002, pp. 187–192 (2002)

16. Basu, C., Hirsh, H., Cohen, W.: Recommendation as classification: Using social and content-based information in recommendation. In: The AAAI 1998, pp. 714–720 (1998)
17. Bradley, R.A., Terry, M.E.: Rank analysis of incomplete block designs: I. The method of paired comparisons. Biometrika 39, 324–345 (1952)
18. Coulom, R.: Computing Elo Ratings of Move Patterns in the Game of Go. In: Herik, H.J.V.D., Uiterwijk, J.W.H.M., Winands, M.H.M., Schadd, M.P.D. (eds.) Computers Games Workshop 2007, pp. 113–124 (2007)
19. Hunter, D.R.: MM algorithms for generalized Bradley-Terry models. The Annals of Statistics 32(1), 384–406 (2004)
20. Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: Proc. of the SIGIR 1999, pp. 230–237. ACM Press, New York (1999)
21. Bennett, J., Lanning, S.: The netflix prize. In: KDD Cup and Workshop (2007)