Information Processing Letters 111 (2011) 440–446

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Userrank for item-based collaborative filtering recommendation ☆

Min Gao [a,*], Zhongfu Wu [b], Feng Jiang [c]

[a] School of Software Engineering, Chongqing University, Chongqing 400044, China
[b] College of Computer Science, Chongqing University, Chongqing 400044, China
[c] Chongqing Radio and TV University, Chongqing 400052, China

ARTICLE INFO

Article history:
Received 23 February 2010
Received in revised form 7 February 2011
Accepted 7 February 2011
Available online 15 February 2011
Communicated by J. Chomicki

Keywords:
Algorithms
Personalized recommendation
Userrank
Collaboration filtering
Item-based filtering

ABSTRACT

With the recent explosive growth of the Web, recommendation systems have been widely accepted by users. Item-based Collaborative Filtering (CF) is one of the most popular approaches for determining recommendations. A common problem of current item-based CF approaches is that all users have the same weight when computing the item relationships. To improve the quality of recommendations, we incorporate the weight of a user, userrank, into the computation of item similarities and differentials. In this paper, a data model for userrank calculations, a PageRank-based user ranking approach, and a userrank-based item similarities/differentials computing approach are proposed. Finally, the userrank-based approaches improve the recommendation results of the typical Adjusted Cosine and Slope One item-based CF approaches.

1. Introduction

With the recent explosive growth of the Web, recommendation systems have been widely accepted by users [1, 2]. Personalized recommendation approaches have gained great momentum both in the commercial and research areas [3,4]. There currently exist several recommendation systems, including MovieLens [5], Jester [6], Amazon [7,8], and Netflix [9].

Collaborative Filtering (CF) is the most popular approach used in recommendation systems [3,10,11]. User-based CF, the traditional CF method, was the most successful technique for building a recommendation system [3, 12]. However, it suffers from serious problem in scalability. It has been experimentally proven that item-based CF can solve the problem [5,12,21]. It is proposed to build offline an item–item similarity matrix for rating prediction. It uses a pre-computed model and will therefore be capable of recommending items quickly. Similar to the issue that items have the same weight in user-based CF [13], a problem of current item-based CF is that all users have the same weight when item similarities or differentials are computed. It is a common knowledge that some users' recommendations are more important than those of others in a social group. Thus, for item-based CF recommendations, some users (and their ratings) should be weighted higher than others are. In this paper, a userrank approach is proposed to compute the relative weights of users in order to solve this problem. This problem is approached as follows: (1) A user correlation graph model is presented for userrank calculations, (2) two rules are formulated for ranking users and a weighted PageRank algorithm is proposed for userrank calculation, (3) we incorporate userrank into approaches for computing item–item similarities and differentials, and (4) userrank-based approaches are experimentally proven to provide better recommendation results (i.e. recommendation coverage and stability) than typical item-based CF approaches.

The remainder of this paper is organized as follows. Section 2 presents an overview of item-based CF and the associated problems. Then Section 3 presents a data model

☆ Research partially supported by the Chongqing Municipal Education Commission, under grant KJ101602, and the Higher Education Press (Digital publishing support environment project).

* Corresponding author.
E-mail addresses: mingaoo@gmail.com (M. Gao), wzf@cqu.edu.cn (Z. Wu), jiangfeng@cqu.edu.cn (F. Jiang).

0020-0190/$ – see front matter
doi:10.1016/j.ipl.2011.02.003

for userrank calculation, which is a user ranking algorithm based on PageRank algorithm, and approaches for computing item–item similarities/differentials for item-based CF recommendation. Section 4 presents experimental evaluations of our approach on a popular database: MovieLens, and a comparison of our results with those of typical item-based CF approaches. Finally, Section 5 draws conclusions.

2. Overview of item-based CF and associated problems

The basic concept of traditional CF, also called user-based CF, is to predict the rating of an item for a target user based on the opinions of other like-minded users. While this approach was very successful, some potential problems have arisen [5] such as problems with scalability. It has been experimentally proven that item-based CF can solve these problems [12].

2.1. Item-based CF

Item-based CF proposed by Sarwar et al. [5], works by comparing items based on the pattern of their ratings across users. There are several algorithms for computing item similarities/differentials, such as Adjusted Cosine [21] and Slope One [14].

In the former, the similarity of items is computed by $sim_{i,j} = \frac{\sum_{u\in U(i)\cap U(j)} (r_{u,i}-\bar{r_u})\times(r_{u,j}-\bar{r_u})}{\sqrt{\sum_{u\in U(i)\cap U(j)} (r_{u,i}-\bar{r_u})^2}\sqrt{\sum_{u\in U(i)\cap U(j)} (r_{u,j}-\bar{r_u})^2}}$ where $U(i)$ is the set of users who have rated on an item i. Formally, $U(i) = \{u \mid r_{u,i} \neq \varnothing\}$ is the average of user u's ratings. In addition, there are many ways to estimate a rating, the most important step in a CF recommendation system, including weighted sum

$$p_{u,i} = \frac{\sum_{j\in S(i)} sim(i,j) \times r_{u,j}}{\sum_{j\in S(i)} |sim(i,j)|} \tag{1}$$

and regression (see formula (2)). Here $S(i)$ is the set of items that are similar to item i.

Slope One is another typical item-based CF approach. It works by comparing the intuitive principle of popular differentials between items [14] rather than similarities. The difference between item i and j, $d_{i,j}$, is the average difference between the item arrays of i and j, $d_{i,j} = \sum_{u\in U(i)\cap U(j)} \frac{r_{u,i}-r_{u,j}}{|U(i)\cap U(j)|}$, where $\|$ denotes the cardinality of a set. In turn, the deviations of items are used to predict that of an unknown item, given their ratings of the others. The prediction is based on a linear regression model

$$p_{u,i} = \frac{\sum_{j\in r_u} r_{u,j} + d_{i,j}}{|r_u|}. \tag{2}$$

Here $p_{u,i}$ is a prediction rating and $\bar{r_u}$ is the average of all known ratings of user u.

It has been proven that the accuracy of the Slope One algorithm is comparable to that of the Adjusted Cosine and the Pearson scheme [14]. Slope One has thus attracted considerable academic and commercial interest on account of its simplicity and efficiency [26–29].

2.2. Weight problem of current item-based CF

The relationships between items are the basis of CF approaches. In current approaches for computing relationships, whether for item similarities or differentials, the weights of all users are the same. That is, the weights of users are not considered in these approaches. However, in any social group, some persons have higher prestige than others do because they have been in the group for a long time or they have made greater contributions to the group. Therefore, if user weights are taken into consideration in item-base CF, the similarities or differentials between items will become more realistic.

In this paper, we propose a userrank approach to rank the importance of users to make recommendations. The details of the approach and algorithm are discussed in the next section.

3. Userrank for item-based CF

In this section, we propose a data model for userrank calculation, present a PageRank-based user ranking approach, and incorporate userrank into approaches for computing item similarities/differentials.

3.1. Data model for user ranking

Just as there are various user relationships in any social group, there are different degrees of correlation between users in a recommendation system. We exploit this information for user ranking.

Essentially, the more items that have been rated by both user u_i and user u_j, the closer the users are [15]. This is the *first rule* of computing correlations between users.

We define $I(u_i)$ as the set of items rated by user u_i. $I(u_i, u_j)$ is the set of items rated by both u_i and u_j,

$$I(u_i,u_j) = \begin{cases} \{i_k\colon (r_{u_i,i_k} \neq \varnothing) \wedge (r_{u_j,i_k} \neq \varnothing)\} & (u_i \neq u_j), \\ \varnothing & (u_i = u_j). \end{cases}$$

Definition 3.1. *CRM* is a $(|U| \times |U|)$ correlation rating matrix that records the number of items that have been rated by each pair of users. $|U|$ denotes the cardinality of the set of users. *CRM* is formed by all $I(u_i, u_j)$.

CRM is a symmetric matrix; therefore, $I(u_i, u_j)$ is the same as $I(u_j, u_i)$. However, if u_i has rated many items and u_j has rated only a few items, their correlation values should differ. This is the *second rule*. According to this rule, the *CRM* matrix can be normalized to *CM*.

Definition 3.2. *CM* is a correlation matrix that records the relationships between users according to the number of items they have rated and the numbers of items rated by the others.

CM can be computed with the formula

$$CM_{u_i,u_j} = \frac{CRM_{u_i,u_j}}{\sum_{u_j\in U} CRM_{u_i,u_j}} = \frac{|I(u_i,u_j)|}{\sum_{u_j\in U} |I(u_i,u_j)|}. \tag{3}$$

Table 1
A rating matrix.

	i_1	i_2	i_3	i_4	i_5	i_6
u_1	5	∅	3	4	2	∅
u_2	3	∅	4	∅	3	4
u_3	∅	4	∅	2	∅	∅
u_4	3	2	3	4	5	3
u_5	∅	∅	3	∅	2	∅

Table 2
CRM of *RM*.

	u_1	u_2	u_3	u_4	u_5
u_1	0	3	1	4	2
u_2	3	0	0	4	2
u_3	1	0	0	2	0
u_4	4	4	2	0	2
u_5	2	2	0	2	0

Without loss of generality, suppose that $\sum_{u_j \in U} |I(u_i, u_j)| \neq 0$. Note that, CM_{u_i,u_j} can differ from CM_{u_j,u_i}; therefore, *CM* is an unsymmetrical matrix.

Theorem 1. *For a user u_j, the sum of his/her correlation values with the other users will be* 1. *Formally,* $\sum_{u_j \in U} CM_{u_i,u_j} = 1$.

Proof. There is

$$\sum_{u_j \in U} CM_{u_i,u_j} = \sum_{u_j \in U} \frac{CRM_{u_i,u_j}}{\sum_{u_j \in U} CRM_{u_i,u_j}} = \sum_{u_j \in U} \frac{|I(u_i,u_j)|}{\sum_{u_j \in U} |I(u_i,u_j)|}.$$

Here, $\sum_{u_j \in U} |I(u_i,u_j)| = |I(u_i,u_1)| + |I(u_i,u_2)| + \cdots + |I(u_i,u_n)|$ is a constant. Given $c = \sum_{u_j \in U} |I(u_i,u_j)|$, then

$$\sum_{u_j \in U} CM_{u_i,u_j} = \sum_{u_j \in U} \frac{|I(u_i,u_j)|}{c} = \frac{1}{c} \times \sum_{u_j \in U} \big|I(u_i,u_j)\big| = \frac{1}{c} \times c = 1. \qquad \square$$

CM is regarded as a weighted connective matrix of a correlation graph, G. Nodes in G correspond to users and there is a link (u_i, u_j) from u_i to u_j if the weight of the link does not equal to zero ($CM_{u_i,u_j} \neq 0$). G is a valuable model to further exploit userrank.

For example, Table 1 shows a rating matrix (RM). There are 5 users, 6 items, and several ratings. A rating is marked from 1 to 5 to indicate the preference of a user for an item. A rating of $r_{u,i} = \varnothing$ means that item i is not rated by user u.

Table 2 shows the *CRM* of the *RM*. Every element in the matrix is the number of items that have been rated by each pair of users. *CRM* is a symmetrical matrix.

Table 3 shows the *CM* of the *RM*. *CM* is an unsymmetrical matrix. For each user pair (u_i, u_j), where $i \neq j$, CM_{u_i,u_j} cannot be equal to CM_{u_j,u_i}, e.g. $CM_{u_1,u_2} = 3/10 = 0.3$, $CM_{u_2,u_1} = 3/9 = 0.333$, $CM_{u_1,u_2} \neq CM_{u_2,u_1}$. The sum of each row in *CM* is 1, e.g., $\sum_{u_j \in U} CM_{u_i,u_j} = 0.3 + 0.1 + 0.4 + 0.2 = 1$.

Table 3
CM of *RM*.

	u_1	u_2	u_3	u_4	u_5
u_1	0	3/10	1/10	4/10	2/10
u_2	3/9	0	0	4/9	2/9
u_3	1/3	0	0	2/3	0
u_4	4/12	4/12	2/12	0	2/12
u_5	2/6	2/6	0	2/6	0

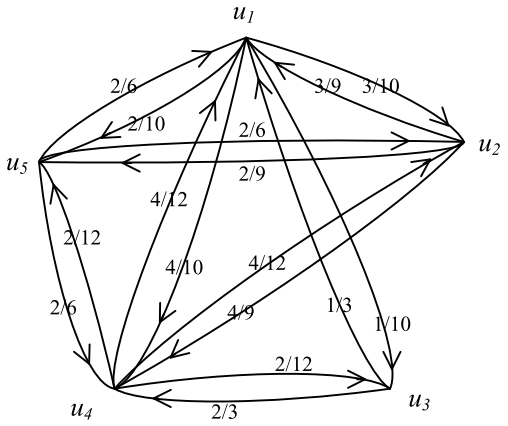

Fig. 1. Date model of the *RM*.

Fig. 1 shows the data model for userrank in this case.

3.2. User ranking algorithm

Given a graph such as that shown in Fig. 1, we would like to measure the rank of each of the vertices. The ranks are spread throughout the graph; therefore, it is important to properly control the rank flow in order to transfer a user's scores to others strongly related to him/her.

The spreading algorithm follows two rules: (1) if a user u_i is linked by highly ranked users with high weights, then u_i will also have high rank and (2) a user has to transfer his rank throughout the graph, but this effect decreases in power as it spreads increasingly further away. Moreover, if u_i is connected to two or more nodes, these nodes share the boosting effect according to the weights of the links computed in *CM*.

The propagation and attenuation rules of userrank are similar to those of the PageRank algorithm. Eigenvector centrality provides a principled method to combine the 'importance' of a vertex with those of its neighbors in ranking [16]. PageRank is such a very successful application of eigenvector centrality. It computes a rank vector to rate the importance of all Web pages by analyzing their hyperlinks. The PageRank model also captures the importance of users when it is applied to a human interaction network [23]. With the significant research into PageRank computing [17–19], we can compute userrank in an efficient manner.

Given a graph $G = (V, E)$, where V is the set of nodes connected by directed links in E, PageRank computes an importance score, $PR(n)$, for each node according to the

graph connectivity: a node will be important if it is connected by important nodes with a low out-degree. Therefore, the PageRank score [20] for node n is defined as $PR(n) = (1-\alpha)\cdot\frac{1}{|V|} + \alpha\cdot\sum_{q:\ (q,n)\in E}\frac{PR(q)}{O(q)}$, where $O(q)$ is the out-degree of node q and α is a decay factor [13].

As in the case of the PageRank, the userrank (UR) is iteratively defined as follows:

$$UR(u_n) = (1-\alpha)\cdot\frac{1}{|V_w|}\cdot UR(u_n) + \alpha\cdot\sum_{u_k:\ (u_k,u_n)\in E}\frac{UR(u_k)}{O_w(u_k)}.$$

Here, u_k is a node linked from u_n. $O(u_k)$ is only the number of outputs of node u_k. In other words, it is only the number of nodes links to node u_k. However, as can be seen from the correlation graph, the different links may have different weights. Therefore, we replace $O(u_k)$ with $O_w(u_k)$, the inverse weighted ratio of the link (u_n, u_k), as follows:

$$O_w(u_k) = \frac{\sum_{u_m:\ (u_k,u_m)\in E} w_{u_k,u_m}}{w_{u_k,u_n}} = \frac{\sum_{u_m:\ (u_k,u_m)\in E} CM_{u_k,u_m}}{CM_{u_k,u_n}}.$$

Consequently, the importance value of node n is

$$UR(u_n) = (1-\alpha)\cdot\frac{1}{|V_w|}\cdot UR(u_n) + \alpha\cdot\sum_{u_k:\ (u_k,u_n)\in E}\frac{UR(u_k)}{O_w(u_k)}.$$

Because $\sum_{u_m:\ (u_m,u_k)\in E} CM_{u_m,u_k}$ is always 1 (see Theorem 1), the formula for $UR(u_n)$ is simplified to

$$UR(u_n) = (1-\alpha)\cdot\frac{1}{|V_w|}\cdot UR(u_n) + \alpha\cdot\sum_{u_k:\ (u_k,u_n)\in E} UR(u_k)\times CM_{u_k,u_m}.$$

As in the case of $O_w(u_k)$, the weighted $\frac{1}{|V_w|}\cdot UR(u_n)$ is $UR(u_n)$ because the sum of the weights of u_n's outputs is 1: $Vw = \sum_{u_m:\ (u_m,u_n)\in E} CM_{u_m,u_n} = 1$. Thus, the calculation for userrank is simplified finally to

$$UR(u_n) = (1-\alpha)\cdot UR(u_n) + \alpha\cdot\sum_{u_k:\ (u_k,u_n)\in E} UR(u_k)\times CM_{u_k,u_m}. \quad (4)$$

Given that the initial userrank is 1/5, $\alpha = 0.5$, according to formula (4) and the data model (Fig. 1), the iteration values of userrank are listed in Table 4.

The values of userrank converge [20] and are as expected. For instance, u_1 and u_4 have greater ranks than the others because they have more correlation users than the others; u_4's rank is greater than u_1's because u_4 has greater correlation items than u_1; u_3's rank is the lowest because he/she has the least correlated users.

Table 4
User ranks in *RM*.

	$Iteration_1$	$Iteration_3$	$Iteration_5$	$Iteration_6$	$Iteration_7$
u_1	0.233	0.248	0.250	0.250	0.250
u_2	0.202	0.217	0.223	0.224	0.224
u_3	0.128	0.086	0.078	0.076	0.076
u_4	0.268	0.297	0.300	0.300	0.300
u_5	0.168	0.152	0.150	0.150	0.150

3.3. Userrank-based approaches for computing item similarities and differentials

In this research, $UR(u)$ is regarded as the weight of user u, $w_u = UR(u)$. We then combine userrank with Adjusted Cosine and Slope One to calculate item similarities and differentials as follows:

$$sim_{i,j} = \frac{\sum_{u\in U(i)\cap U(j)}(r_{u,i}-\bar{r_u})\times(r_{u,j}-\bar{r_u})\times w_u^2}{\sqrt{\sum_{u\in U(i)\cap U(j)}(r_{u,i}-\bar{r_u})^2\times w_u^2}\sqrt{\sum_{u\in U(i)\cap U(j)}(r_{u,j}-\bar{r_u})^2\times w_u^2}}.$$

and

$$d_{i,j} = \frac{\sum_{u\in U(i)\cap U(j)}(r_{u,i}-r_{u,j})\times w_u}{\sum_{u\in U(i)\cap U(j)} w_u}.$$

Then, the relevant formulas, as shown in Section 2.1,

$$p_{u,i} = \frac{\sum_{j\in I(u)} sim(i,j)\times r_{u,j}}{\sum_{j\in I(u)}|sim(i,j)|}$$

and

$$p_{u,i} = \frac{\sum_{j\in I(u)} r_{u,j} + d_{i,j}}{|I(u)|}$$

are applied to predict ratings for Adjusted Cosine and Slope One respectively.

4. Experimental evaluation

4.1. Dataset

MovieLens and Netflix are widely used datasets for recommendation systems [21,26]; however, downloads via Netflix is ceased in 2010. Therefore, we used the MovieLens dataset to evaluate our approaches. The dataset consists of 100,000 ratings (1–5) from 943 users on 1682 movies. Each user has rated at least 20 movies. The dataset was randomly divided into a training set (80,000 ratings) and a test set (20,000 ratings) 50 times during our experiments. The training and test sets are named $U_i base$ and $U_i test$ $(i = 1, \ldots, 50)$.

4.2. Experimental metrics and evaluation methodology

Three metrics and their shifts are used to evaluate the algorithms: mean absolute error (MAE), coverage, and F-measure. Of these three, the most important metrics are MAE and coverage [23]. For recommendation systems, MAE represents the accuracy of predicted rating, an important metric for customers. Coverage indicates that how well the

system discovers items that are desirable to a user, an important metric for resource providers because their objective is to sell more products. Coverage is similar to the recall of an information retrieval domain; however, coverage does not make sense if the recommendation precision is too low. Therefore, a comprehensive indicator, F-measure, which considers both the precision and the recall, is used to measure the recommendation classification accuracy.

MAE is a widely used metric for the deviation of predictions from their true values. Therefore, MAE is used to measure the prediction precision of our algorithms. For all predictions $\{p_1, p_2, \ldots, p_n\}$ and corresponding real ratings, $\{r_1, r_2, \ldots, r_n\}$, $MAE = \frac{\sum_{i=1}^{N} |p_i - r_i|}{N}$ [22] is the average of the absolute error between all $\{p_i, r_i\}$ pairs. The lower the MAE, the better is the approach.

Coverage is the percentage of correctly predicted "high" ratings (A) among all of the ratings known to be "high" (B) [24], $Coverage = \frac{A \cap B}{B}$. The "high" ratings in the experiments are the ratings more than 3.

F-measure (F) considers both the precision (P) and the recall (R) of the test as follows: $F_\beta = \frac{(1+\beta^2)\cdot(P\cdot R)}{\beta^2 \cdot P + R}$ where P is the percentage of truly "high" ratings (B) among those that were predicted to be "high" by a recommendation system (A) [24]: $P = \frac{A \cap B}{A}$. β is a regular certain value of 0.5, 1, or 2. As β increases, the weight of the recall in the measure increases. When $\beta = 1$, the F-measure, F_1, is the harmonic mean of the precision and the recall.

The shifts of these three metrics are used to show the stability of the algorithms:

$$MAEShift = \sum_{i=1}^{n} \frac{|MAE'_i - MAE_i|}{n},$$

$$CoverageShift = \sum_{i=1}^{n} \frac{|Coverage'_i - Coverage_i|}{n},$$

and

$$F_1Shift = \sum_{i=1}^{n} \frac{|F'_{1_i} - F_{1_i}|}{n}.$$

In the experiments, the first step was to use $U_i base$ values to compute the userrank. Then, we predicted the ratings of all users in $U_i test$ for the algorithms in order to compare their MAE, coverage, and F-measure values. To evaluate the stability of the recommendations of the algorithms, we extracted two subsets from every $U_i test$ by selecting ratings of more than 3 and more than 4, referred to as R3 and R4, respectively. When we evaluated the stability of the algorithms, all of the ratings in the subsets were predicted in order to compute MAEShift, CoverageShift, and F_1 Shift values. For the computation of the shifts, we considered $n = 50$ implying that there were 50 R3 and R4 subsets, denoted as $R3_i$ and $R4_i$, respectively. MAE'_i, $Coverage'_i$, and F'_{1_i} are the metric values obtained when the algorithm predicts all ratings in R3 subsets. MAE_i, $Coverage_i$, and F_{1_i} are the metric values obtained when the algorithm predicts all ratings in R4 subsets. We did not perform the same for "low" ratings because a user would be only interested in "high" ratings on a recommendation list.

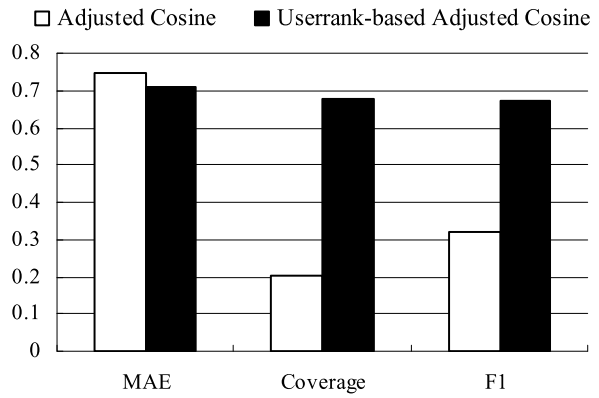

Fig. 2. Experimental results of Adjusted Cosine and userrank-based Adjusted Cosine recommendation algorithms.

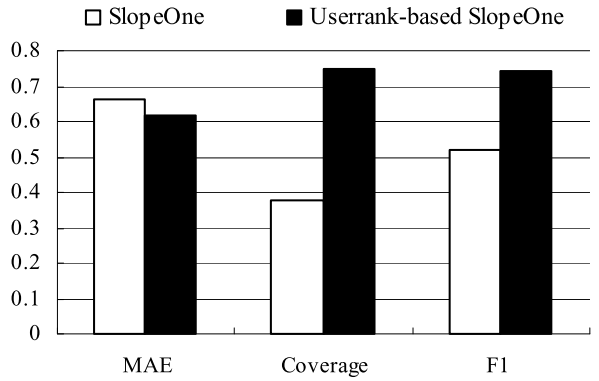

Fig. 3. Experimental results of Slope One and userrank-based Slope One recommendation algorithms.

Table 5
MAE of algorithms.

	Adjusted Cosine	Userrank-based Adjusted Cosine	Slope One	Userrank-based Slope One
MAE	0.837	0.806	0.740	0.702

4.3. Experimental procedure and results

4.3.1. Comparison of prediction results

To compare userrank-based algorithms (i.e. userrank-based Adjusted Cosine and userrank-based Slope One) with typical algorithms (i.e. Adjusted Cosine and Slope One), we experimentally predicted all ratings in $U_i test$ and computed MAE, coverage, and F_1 for the algorithms. The results are shown in Fig. 2 and Fig. 3. The white columns correspond to those of typical algorithms and the black ones correspond to those of userrank-based algorithms. As can be observed, the userrank-based algorithms outperform typical algorithms in all three metrics.

1) Table 5 lists the MAE values of the algorithms. The lower the MAE, the better are the algorithms.

Table 6
Coverage (C) and F_1 of algorithms.

	Adjusted Cosine	Userrank-based Adjusted Cosine	Slope One	Userrank-based Slope One
C	20.3%	67.9%	38%	74.8%
F_1	32.2%	67.4%	52.2%	74.5%

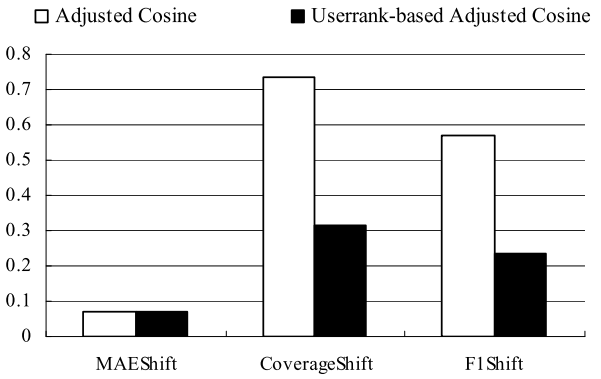

Fig. 4. Metric shifts of Adjusted Cosine and userrank-based Adjusted Cosine recommendation algorithms.

2) Table 6 lists the coverage and F-measure values of the algorithms. The higher the metrics, the better are the algorithms.

4.3.2. Comparison of stability of algorithms

Stability indicates the shift in a system's ratings for items in different test subsets. To compare the stability of the algorithms, we experimentally predicted all of the ratings in R3 and R4 and computed the MAE, coverage, and F_1 of these predictions for R3 and R4 respectively. MAE'_i, $Coverage'_i$, and F'_{1_i} are the metric values for each R3 subset, whereas MAE_i, $Coverage_i$, and F_{1_i} are the metric values for each R4 subset. Their difference yields the values of MAEShift, CoverageShift, and F_1Shift. Fig. 4 shows the corresponding results for Adjusted Cosine (white columns) and userrank-based Adjusted Cosine (black columns) algorithms. Fig. 5 shows the same for Slope One (white columns) and userrank-based Slope One (black columns) algorithms. As can be seen from the figures, the prediction stability of userrank-based algorithms is better than that of the typical algorithms. The details are as follows.

As can be seen from Fig. 4, the MAE shifts of the Adjusted Cosine and userrank-based Adjusted Cosine algorithms are very similar. For the CoverageShift metric, the respective algorithms have values of more than 70% and only approximately 30%. For the F_1 shifts metric, the respective values are almost 60% and only approximately 20%. Therefore, both shifts are large.

As can be seen from Fig. 5, Slope One and userrank-based Slope One algorithms exhibit similar trends. However, the shifts of the two Slope One algorithms are lower than those of the two Adjusted Cosine algorithms.

In summary, all of the metrics, including MAE, coverage, F_1 and their respective shifts, were better in the case of the userrank-based algorithms than in the case of the typical algorithms. A possible reason is that all ratings for computing item similarities and differentials have the same weight in the baseline approaches, in other words, the weights of users are not taken into consideration.

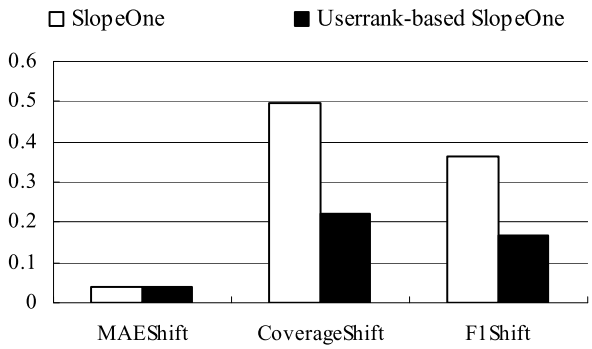

Fig. 5. Metric shifts of Slope One and userrank-based Slope One recommendation algorithms.

4.3.3. Comparisons with other PageRank-based approaches

Massa and Avesani [25] proposed a trust-aware recommendation approach, PageRank-based MoleTrust2, and tested it against a dataset derived from Epinions.com. Their approach aimed to solve the new-user problem, essentially, the cold-start problem of recommendation systems. Our approach aims to improve the prediction results and recommendation stability for all users. Gori et al. [13] presented PageRank-based ItemRank to improve traditional user-based CF methods. Although ItemRank is a user-based recommendation approach, our userrank-based approaches employ item-based CF, and therefore, we did not compare ItemRank with our approaches.

5. Conclusions

Currently, item-based collaborative filtering approaches, Adjusted Cosine and Slope One being well known examples of the same, are popularly employed in recommendation systems. In this paper, we analyzed the ranking of users and prediction of ratings based on userrank. Experimental results show that userrank information helps improve the prediction results and the stability of the typical algorithms.

References

[1] M. Gao, K. Liu, Z. Wu, Personalisation in web computing and informatics: theories, techniques, applications, and future research, Information Systems Frontiers 12 (2009) 607–629, doi:10.1007/s10796-009-9199-3.
[2] F. Jiang, M. Gao, Collaborative filtering approach based on item and personalized contextual information, in: Proceedings of the International Symposium on Intelligent Information Systems and Applications, Qingdao, P.R. China, 2009, pp. 63–66.
[3] M. Gao, Z. Wu, Personalized context-aware collaborative filtering based on neural network and slope one, in: Lecture Notes in Computer Science, vol. 5738, 2009, pp. 109–116
[4] M. Eirinaki, M. Vazirgiannis, Web mining for web personalization, ACM Transactions on Internet Technology 3 (2003) 1–27.
[5] B. Sarwar, G. Karypis, J. Konstan, J. Reidl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th International Conference on World Wide Web, Hongkong, 2001, pp. 285–295.

[6] D. Gupta, M. Digiovanni, H. Narita, K. Goldberg, Jester 2.0 (poster abstract): evaluation of an new linear time collaborative filtering algorithm, 1999, pp. 291–292.
[7] G. Linden, B. Smith, J. York, Amazon.com recommendations: item-to-item collaborative filtering, IEEE Internet Computing 7 (2003) 76–80.
[8] A.S. Das, M. Datar, A. Garg, S. Rajaram, Google news personalization: scalable online collaborative filtering, in: Proceedings of the 16th International Conference on World Wide Web, 2007, pp. 271–280.
[9] Netflix, http://www.netflix.com/, 2010.
[10] J.A. Konstan, B.N. Miller, D. Maltz, J.L. Herlocker, L.R. Gordon, J. Riedl, GroupLens: applying collaborative filtering to Usenet news, Communications of the ACM 40 (1997) 77–87.
[11] Y. Li, L. Lu, L. Xuefeng, A hybrid collaborative filtering method for multiple-interests and multiple-content recommendation in E-Commerce, Expert Systems with Applications 28 (2005) 67–77.
[12] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions, Knowledge and Data Engineering, IEEE Transactions 17 (2005) 734–749.
[13] M. Gori, A. Pucci, V. Roma, I. Siena, Itemrank: A random-walk based scoring algorithm for recommender engines, in: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI), Hyderabad, India, 2007, pp. 778–781.
[14] D. Lemire, A. Maclachlan, Slope One predictors for online rating-based collaborative filtering, in: Proceedings of the SIAM Data Mining Conference, Newport Beach, California, 2005, pp. 471–475.
[15] F. Zhang, Research on trust based collaborative filtering algorithm for user's multiple interests, Journal of Chinese Computer Systems 29 (2008) 1415–1419.
[16] Y. Jing, S. Baluja, PageRank for product image search, in: WWW 2008, 2008, pp. 307–316.
[17] A.N. Langville, C.D. Meyer, Deeper inside PageRank, Internet Mathematics 1 (2004) 335–380.
[18] T. Haveliwala, Efficient computation of PageRank, Technical report 1999-31, Stanford University, 1999, http://dbpubs.stanford.edu.
[19] S.D. Kamvar, T.H. Haveliwala, C.D. Manning, G.H. Golub, Extrapolation Methods for Accelerating PageRank Computations, ACM, New York, NY, USA, 2003, pp. 261–270.
[20] C.L. Page, S. Brin, R. Motwani, T. Winograd, The PageRank citation ranking: Bring order to the web, Technical report, Stanford Digital Libraries, 1998.
[21] T. Segaran, Programming Collective Intelligence: Building Smart Web 2.0 Applications, O'Reilly Media, Inc., 2007.
[22] 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.
[23] D. Schall, S. Dustdar, Expertise ranking in human interaction networks based on PageRank with contextual skill and activity measures, http://citeseerx.ist.psu.edu/, 2009.
[24] K.S. Esmaili, M. Neshati, M. Jamali, H. Abolhassani, J. Habibi, Comparing performance of recommendation techniques in the blogsphere, in: ECAI 2006 Workshop on Recommender Systems, Riva del Garda, Italy, 2006, pp. 40–44.
[25] P. Massa, P. Avesani, Trust-aware bootstrapping of recommender systems, in: ECAI 2006 Workshop on Recommender Systems, Riva del Garda, Italy, 2006, pp. 29–33.
[26] G. Louppe, Collaborative filtering: Scalable approaches using restricted Boltzmann machines, Master's thesis, University of Liège, 2010.
[27] D. Jannach, K. Hegelich, A case study on the effectiveness of recommendations in the mobile internet, in: Proceedings of the third ACM Conference on Recommender Systems (RecSys '09), 2009, pp. 205–208.
[28] R.J. Nadolski, B. Van den Berg, A.J. Berlanga, H. Drachsler, H.G.K. Hummel, R. Koper, P.B. Sloep, Simulating light-weight personalised recommender systems in learning networks: A case for pedagogy-oriented and rating-based hybrid recommendation strategies, Journal of Artificial Societies and Social Simulation 12 (2009) 4–25.
[29] M. Gao, Z. Wu, Incorporating personalized contextual information in item-based collaborative filtering recommendation, Journal of Software 5 (2010) 729–736.