User-Weight Model for Item-based Recommendation Systems

Min Gao, Yunqing Fu
College of Software Engineering, Chongqing University, Chongqing 400044, China
Email: {gaomin, yqfu}@cqu.edu.cn

Yixiong Chen
College of Computer Science, Chongqing University, Chongqing 400044, China
Email: chenyx@cqu.edu.cn

Feng Jiang
College of Civil Engineering, Chongqing University, Chongqing 400044, China
Construction Engineering Department, Chongqing Technology and Business Institute, Chongqing 400050, China
Email: jiangfeng@cqu.edu.cn

Abstract—Nowadays, item-based Collaborative Filtering (CF) has been widely used as an effective way to help people cope with information overload. It computes the item-item similarities/differentials and then selects the most similar items for prediction. A weakness of current typical item-based CF approaches is that all users have the same weight in computing the item relationships. In order to improve the recommendation quality, we incorporate users’ weights based on a relationship model of users into item similarities and differentials computing. In this paper, a model of user relationship, a method for computing users’ weights, and weight-based item-item similarities/differentials computing approaches are proposed for item-based CF recommendations. Finally, we experimentally evaluate our approach for recommendation and compare it to typical item-based CF approaches based on Adjusted Cosine and Slope One. The experiments show that our approaches can improve the recommendation results of them.

Index Terms—personalized recommendation, collaboration filtering, item-based filtering, relationship model

I. INTRODUCTION

Due to the explosive growth of the Web, recommendation systems have been widely accepted by users. Users offer feedback on purchased or consumed items, and recommendation systems use the information to predict their preferences for yet unseen items and subsequently recommend items with the highest predicted ratings for users [1, 2]. Personalized recommendation approaches have gained great momentum both in the commercial and research areas [3]. There have been several famous recommendation systems, such as Amazon [4, 5] and Netflix (http://www.netflix.com/).

A problem of current item-based CF is that all users have the same weight when item-item similarities or differentials are computed. There is a common sense that some users’ words are more important than others’ in a social group. For item-based CF recommendation, that is, some users (and their ratings) will have higher weights than the others. In this paper a novel calculation approach is proposed to compute the weights of users to improve the recommendation results of typical CF approaches.

The contribution of the paper includes four points. Firstly, a novel user relationship graph model is presented. Secondly, an algorithm based on three rules is given for weighting users. Thirdly, we incorporate users’ weights into computing item-item similarities and differentials. Fourthly, the weight-based approach is proved experimentally helpful to improve the recommendation results of item-based CF approaches.

In the next sections, the state of the art in item-based CF is reviewed first (Section 2). Then Section 3 proposes a model of users’ relationship, a user weighting method, and weight-based item-item similarities/differentials computing approaches for item-based CF recommendations. And then we evaluate experimentally our approaches on a popular dataset MovieLens and compare them to typical item-based CF approaches (Section 4). At last, Section 5 draws conclusions.

II. BACKGROUND AND PROBLEM OF ITEM-BASED CF

CF is the most successful recommendation technique to date [6, 7]. In a typical CF scenario, there is a rating m*n matrix which includes a list of m users and a list of n items and lots of ratings. Items represented any kind of products. A rating r_{ui} means how the user u likes the item i. It is supposed that users mainly interested in high ratings. The key step of CF is to extrapolate unknown ratings [8].
The basic idea of traditional CF is to predict the rating of an item for a target user based on the opinions of other like-minded users. Item-based CF is the most popular approach in recommendation systems. It builds an item-item similarity/differential matrix for recommendations (See Fig. 1 [1]).

A. Item-based CF

Item-based CF, proposed by Sarwar et al. [7], is to compute the similarity between items and then to select the most similar items for prediction. Since it uses a pre-computed model, it recommends items quickly. There are several approaches to compute the similarities between items, such as Adjusted Cosine (See formula 1), and to compute the differences between items (See formula 3), e.g. Slope One [6].

\[
d_{i,j} = \frac{\sum_{u \in U(i) \cap U(j)} (r_{u,i} - \bar{r}_i) (r_{u,j} - \bar{r}_j)}{|U(i) \cap U(j)|}
\]

(3)

In turn, the deviations of items will be used to predict an unknown item, given their ratings of the other. The prediction is based on a linear regression model (See formula 4). Here \( p_{u,j} \) is a prediction rating; \( \bar{r}_i \) is the average of all known ratings of user \( u \); and \( \bar{d} \) is the average of all differentials \( d_{i,j} \). \( r_{u,i} - d_{i,j} \) is the prediction for \( r_{u,j} \) according to \( r_{u,i} \).

\[
p_{u,j} = \frac{\sum_{i \in S(i)} (r_{u,i} - \bar{d}_{i,j})}{|r_u|} = \bar{r}_u + \bar{d}
\]

(4)

B. Weight Problem

The Slope One is a typical item-based collaborative filtering approach. It works on comparing the intuitive principle of popular differentials between items [6]. Note that it computes the differentials between items rather than similarities. The differential of item \( i \) and \( j \) \( d_{i,j} \) is the average difference between item arrays of \( i \) and \( j \) (See formula 3). \(| | \) denotes the cardinality of a set.

The basic idea of traditional CF is to predict the rating of an item for a target user based on the opinions of other like-minded users. Item-based CF is the most popular approach in recommendation systems. It builds an item-item similarity/differential matrix for recommendations (See Fig. 1 [1]).

The basic idea of traditional CF is to predict the rating of an item for a target user based on the opinions of other like-minded users. Item-based CF is the most popular approach in recommendation systems. It builds an item-item similarity/differential matrix for recommendations (See Fig. 1 [1]).

The basic idea of traditional CF is to predict the rating of an item for a target user based on the opinions of other like-minded users. Item-based CF is the most popular approach in recommendation systems. It builds an item-item similarity/differential matrix for recommendations (See Fig. 1 [1]).

The basic idea of traditional CF is to predict the rating of an item for a target user based on the opinions of other like-minded users. Item-based CF is the most popular approach in recommendation systems. It builds an item-item similarity/differential matrix for recommendations (See Fig. 1 [1]).

The Slope One scheme achieves accuracy comparable to the Adjusted Cosine and Pearson scheme. The Slope One has won the wide attention of researchers and companies due to its simplicity and efficiency [9, 10].

The Slope One scheme achieves accuracy comparable to the Adjusted Cosine and Pearson scheme. The Slope One has won the wide attention of researchers and companies due to its simplicity and efficiency [9, 10].

The Slope One scheme achieves accuracy comparable to the Adjusted Cosine and Pearson scheme. The Slope One has won the wide attention of researchers and companies due to its simplicity and efficiency [9, 10].

The Slope One scheme achieves accuracy comparable to the Adjusted Cosine and Pearson scheme. The Slope One has won the wide attention of researchers and companies due to its simplicity and efficiency [9, 10].
similarities/differentials computing approaches for further prediction.

A. Data Model for Weighting Users

There are various relationships between users in any social group; so are in a recommender system. Now we exploit this information for the calculation of users’ weights.

Traditionally, the rating similarity is seen as user similarity; however, the rating similarity is only one aspect of the user similarity. There are some other relationships behind the ratings. For example, there are many items are rated by both user u and user v; the ratings are very different though. In this case, the rating similarity of them is very low. However, there should be a relationship between them that is high since the rated rating sets of them are similar. In the paper, this relationship is called interesting similarity. Rating similarity is the most used relationship between users in recommendation systems [7]. Therefore, it is not introduced in detail in the paper. The interesting similarity is analyzed as below.

Usually used calculation algorithms of rating similarity are Adjusted Cosine (See Section II.A) and Pearson Correlation. The rating similarity is the most used calculation algorithm for a correlation graph $G$. Nodes in $G$ correspond to users in $U$ and there will be a link $(u_i, u_j)$ from $u_i$ to $u_j$ if $URM_{u_i,u_j} \neq 0$. The weight of the link is $URM_{u_i,u_j}$. The graph $G$ is valuable model to further exploit correlation between users.

For example, given a rating matrix RM is as Table I. There are 5 users, 6 items, and several ratings in the matrix. A rating is marked from 1 to 5 that represents how a user prefers an item. $r_{ui}=\emptyset$ means the item $i$ is not rated by the user $u$.

<table>
<thead>
<tr>
<th></th>
<th>$i_1$</th>
<th>$i_2$</th>
<th>$i_3$</th>
<th>$i_4$</th>
<th>$i_5$</th>
<th>$i_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>4</td>
<td>$\emptyset$</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>$u_2$</td>
<td>3</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>$u_3$</td>
<td>$\emptyset$</td>
<td>4</td>
<td>$\emptyset$</td>
<td>2</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>$u_4$</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>$u_5$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>3</td>
<td>$\emptyset$</td>
<td>4</td>
<td>$\emptyset$</td>
</tr>
</tbody>
</table>

Table II shows the IM of the RM. Every element in the matrix is the number of items which have been rated by each pair of users (see formula 6). IM is a symmetric matrix. For each user pair $(u_i,u_j)$, $IM_{u_i,u_j} = ISM_{u_j,u_i}$, e.g., $ISM_{u_1,u_2} = IM_{u_1,u_2} = 3$.
Table III shows the correlation matrix $IM$ of the RM. $IM$ is an unsymmetrical matrix. For each user pair $(u_i,u_j)$, where $i \neq j$, $IM_{u_i,u_j}$ can be not equal to $IM_{u_j,u_i}$, e.g., $IM_{u_1,u_2} = 3/10 = 0.3$, $IM_{u_2,u_1} = 3/9 \approx 0.333$, $IM_{u_i,u_j} \neq IM_{u_j,u_i}$. The sum of each row in the $IM$ is 1, e.g., $\sum_{u_j \in U} IM_{u_i,u_j} = 1$.

The spreading algorithm follows three rules. Firstly, if a user $u_i$ is linked by high ranked users with high weights, then $u_i$ will also have high weight. Secondly, users have to transfer their positive influence through the graph, but this effect decreases its power if it spreads further and further away. Thirdly, if the user $u_i$ is connected to two or more nodes, these nodes share the boosting effect according to the weights of the connections as computed in $URM$.

The rules are just similar to the propagation and attenuation of PageRank. We can compute users’ weights in a very efficient way based on the method for PageRank computing [12-14].

The PageRank score for the node $n$ is defined as formula (5), where $O(q)$ is the out-degree of the node $q$, $\alpha$ is a decay factor, a common choice of $\alpha$ is 0.85 [15].

$$PR(n) = (1 - \alpha) \cdot \frac{1}{|V|} + \alpha \cdot \sum_{q \in V, E(n \rightarrow q)} PR(q)$$  \hspace{1cm} (5)

So the weight value for a node $u_k$ can be computed by Formula (6).

C. Computing Methods for Item Similarities and Differentials

As can be see from Fig. 3, we combine users’ weights and the procedures of item similarities and differentials algorithms, such as Adjusted Cosine and Slope One (See Formulas 7 and 8). Then we apply formula (2) and (4) to predict ratings.

![Fig. 3 Weighted Item-based Collaborative Filtering](image-url)
IV. EXPERIMENTAL EVALUATION

A. Data Set
In the experiments, we used MovieLens dataset from the well-known MovieLens project (http://MovieLens.umn.edu) to evaluate our approach. MovieLens is a widely used benchmark to evaluate scoring algorithms applied to recommender systems. The data consists of 100,000 ratings (1-5) from 943 users on 1682 movies. Each user has rated at least 20 movies. The data set was divided into training set (80% of the data) and test set (20% of the data) five times.

Each training and test sets are named Unbase and Untest \((n=1,\ldots,5)\). Every Unbase was used to compute user ranks, and the ratings in each Untest were the target ones to predict.

B. Evaluation Metric
Mean Absolute Error (MAE) is a widely used metric for deviation of predictions from their true values. So we used MAE values to measure the prediction precision of our algorithm and the real ratings, and then compare it with Adjusted Cosine and Slope One algorithms. For all predictions \(\{p_1, p_2, \ldots, p_n\}\) and their real ratings \(\{r_1, r_2, \ldots, r_n\}\). MAE is the average of absolute error between all \(\{p_i, r_j\}\) pairs (See formula 8 [16]). The lower the MAE, the more accurately the predictions are, and the better the recommendation approach is.

\[
Sim_{u,j} = \frac{\sum_{u \in U \cap (U \cup \{j\})} (r_{u,i} - \bar{r}_{u}) \times (r_{u,j} - \bar{r}_{u}) \times w_u^2}{\sqrt{\sum_{u \in U \cap (U \cup \{j\})} (r_{u,i} - \bar{r}_{u})^2 \times w_u^2} \sqrt{\sum_{u \in U \cap (U \cup \{j\})} (r_{u,j} - \bar{r}_{u})^2 \times w_u^2}}
\]

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

\[
MAE = \frac{\sum_{i=1}^{N} |p_i - r_i|}{N}
\]

The precision is the percentage of truly “high” ratings (B) among those (A) that were predicted to be “high” by a recommender system (See formula 10). That is the number of correct results divided by the number of all returned results.

\[
\frac{A \cap B}{A}
\]

And the recall is the percentage of correctly predicted “high” ratings among all the ratings known to be “high” (see formula 11). That is the number of correct results divided by the number of results that should have been returned.

F-measure is a measure of a test's accuracy. It considers both the precision and the recall of the test to compute the score (See formula 12). The parameter \(\beta\) has regular certain values of 0.5, 1, and 2.

\[
F_{\beta} = \frac{(1 + \beta^2) \cdot (\text{precision} \cdot \text{recall})}{\beta^2 \cdot \text{precision} + \text{recall}}
\]

C. Procedure and Results
(A). Comparison of prediction results
To compare weight-based algorithms (weight-based Adjusted Cosine and weight-based Slope One) with typical algorithms (Adjusted Cosine and Slope One), we performed the experiment where we computed MAE, precision, recall, and f-measure \((F_{0.5}, F_1, \text{and } F_2)\) for all of them. Our results are shown in Fig.4 and Fig.5. The blue columns (the left columns) are the metric values for one of weight-based algorithms; the purple ones (the right columns) are for one of typical algorithms. It can be observed from the charts that our weight-based algorithms out performs typical algorithms at prediction error (MAE) and recommendation accuracy (F-measure).
Table IV shows the MAE values of the algorithms. The lower the MAE, the more accurately the predictions are.

### Table IV
THE MAE OF EXPERIMENTS

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adjusted Cosine</td>
<td>0.837</td>
</tr>
<tr>
<td>Weight-based Adjusted Cosine</td>
<td>0.806</td>
</tr>
<tr>
<td>Slope One</td>
<td>0.740</td>
</tr>
<tr>
<td>Weight-based Slope One</td>
<td>0.702</td>
</tr>
</tbody>
</table>

Although the values of precision of the weight-based algorithms are somewhat lower than the traditional algorithms, the results are high significant for the recall of the weight-based algorithms. So that the F-measure values are higher than the typical algorithms, that means the accuracy of the algorithms is improved. The F-measure and recall results are shown in Table V.

### Table V
RESULTS OF F-MEASURE AND RECALL

<table>
<thead>
<tr>
<th></th>
<th>Adjusted Cosine</th>
<th>Weight-based Adjusted Cosine</th>
<th>Slope One</th>
<th>Weight-based Slope One</th>
</tr>
</thead>
<tbody>
<tr>
<td>F0.5</td>
<td>49.7%</td>
<td>67.2%</td>
<td>67.3%</td>
<td>74.4%</td>
</tr>
<tr>
<td>F1</td>
<td>32.2%</td>
<td>67.4%</td>
<td>52.2%</td>
<td>74.5%</td>
</tr>
<tr>
<td>F2</td>
<td>23.9%</td>
<td>67.7%</td>
<td>42.6%</td>
<td>74.7%</td>
</tr>
<tr>
<td>Recall</td>
<td>20.3%</td>
<td>67.9%</td>
<td>38%</td>
<td>74.8%</td>
</tr>
</tbody>
</table>

(B). Comparison of the prediction ability for user relevant items

To compare the prediction ability for user relevant items of the algorithms, we performed the experiment where we computed MAE results for the user relevant items in the test set. The user relevant items are those items with 4 or 5 score rated by the user. The table VI shows the MAE results. As can be seen from the table, the prediction abilities of weight-based algorithms are better than the traditional algorithms.

### Table VI
MAE RESULTS OF PREDICTIONS FOR USERS’ RELEVANT ITEMS

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adjusted Cosine</td>
<td>0.75</td>
</tr>
<tr>
<td>Weight-based Adjusted Cosine</td>
<td>0.71</td>
</tr>
<tr>
<td>Slope One</td>
<td>0.66</td>
</tr>
<tr>
<td>Weight-based Slope One</td>
<td>0.62</td>
</tr>
</tbody>
</table>

(C). Comparisons of algorithm stability

To compare the stability of the algorithms, we then performed the experiment where we computed MAE, precision, recall, and f-measure of the prediction results for the ratings more than 3 (R3 for short) and more than 4 (R4 for short) in the test set respectively. Fig. 6 to Fig. 9 shows the deviations of the MAE, precision, recall, and f-measure results of Adjusted Cosine, Weight-based Adjusted Cosine, Slope One, and Weight-based Slope One respectively. The metric values of the prediction for the items in R3 and the items in R4 are shown by the left columns and the right columns in the figures. As can be seen from the figures, the prediction stability of weight-based algorithms is better than the typical algorithms.
As can be seen from Fig.7, the changement trend of all metric values of weight-based Adjust Cosine is similar with Adjust Cosine (Fig.4), but the changement rangeability is lower. The maximum of the change in Fig.5 is only more than 30%, but it is more than 70% in Fig.4.

The Fig.8 and Fig.9 are similar with Fig.6 and Fig.7. Compared with weight-based Adjusted Cosine, the changement rangeability of weight-based Slope One are lower.

In summary, 1) except MAE values, all the other metrics including prediction precision, prediction accuracy, prediction ability for user relevant items, and the stability of the weight-based algorithms are better than the typical algorithms, and 2) the weight-based Slope One are better then user-based Adjusted Cosine at these metrics.

V. CONCLUSION AND FUTURE WORK

The Recommendation systems help users find items they would be interested in. Currently, item-based collaborative filtering approaches are most popular in recommender systems. The typical Adjusted Cosine and Slope One are well-known algorithms of them. In this paper we analyzed how to weight users predict ratings for items based on user ranks. Experimental results show that the information of users’ weights is helpful to improve the prediction results, the prediction ability for user relevant items, and the stability of typical algorithms.

ACKNOWLEDGMENT

We gratefully acknowledge valuable input from Tao Zhou and Peidong Li, MSc students, from the School of Computer Science, Chongqing University.

REFERENCES


