Member Contribution-based Group Recommender System

Wei Wang, Guangquan Zhang, Jie Lu

Decision Systems and e-Service Intelligence Lab

Centre for Quantum Computation and Intelligent Systems,

Faculty of Engineering and Information Technology,

University of Technology Sydney, Australia

Wei.Wang-17@student.uts.edu.au; Guangquan.Zhang@uts.edu.au; Jie.lu@uts.edu.au

Abstract

Generating recommender systems (GRS) is a vital requirement in many online service systems. Unfortunately, GRS cannot be effectively supported using traditional individual recommendation techniques because it is very difficult to satisfy all members, given their conflicting preferences. Our goal is to generate recommendations by taking each group member's contribution into account through weighting members according to their degrees of importance. To achieve this goal, we first propose a member contribution score (MCS) model, which employs the separable non-negative matrix factorization technique on a group rating matrix, to analyze the degree of importance of each member. A Manhattan distance-based local average rating (MLA) model is then developed to refine predictions by addressing the fat tail problem. By integrating the MCS and MLA models, a Member Contribution-based Group Recommendation (MC-GR) Approach is developed. Experiments show that our MC-GR approach achieves a significant improvement in the performance of group recommendations. Lastly, using the MC-GR approach, we develop a group recommender system called GroTo that can effectively recommend activities to web-based tourist groups.

Keywords: Recommender systems, Group recommender systems, Collaborative filtering, Tourism, e-services.

1. Introduction

Many online services, such as e-commerce, e-government and e-learning, suffer from the

information overload problem, i.e. the massive amount of information available for users makes it very difficult to locate the information that users most require. [1-3]. Recommender systems are one of the most successful techniques proposed to address this problem through the analysis of user information to model individual preferences and target relevant related information.

Although significant advances have been made to improve recommender systems, most prior recommender system studies have focused on providing recommendations to individual users (a business or a customer). Group recommender systems (GRSs) have been proposed more recently to produce recommendations for groups of users. GRSs must respond to members' up-to-date preferences and produce recommendations to satisfy the whole group. GRSs have been designed and implemented in many service domains. Sharon et al. [4] designed an internet browser GRS which recommends related links for a set of browsers which have a similar navigation history. Another example called GRec_OC, proposed by [5], can recommend textual information and suggest books for an online reading community. Other than textual recommendation, multimedia content can also be recommended. For example, [6] recommends TV programs for a family instead of an individual viewer; [7] can suggest movies for a group of friends; and MusicFX in [8] is designed to play music that suits the tastes of all the people in a gym. An even more complicated situation arises when recommending a tourism plan for heterogeneous tourist groups (such as families with children and elderly) [9].

From the formation perspective, there are two main types of group in GRSs, regardless of system domain: stable groups and random groups [10]. Members of stable groups may actively join or leave groups, and may specify their preferences. In such groups, members become highly internally correlated, so that group preferences can be centralized over time, and items can then be easily found that satisfy every member in the group. For instance, a reading group might narrow the range of reading to ultimately focus on realist novels or

poems. In contrast, random groups are passively formed by members who have no opportunity to specify their preferences or negotiate a consensus preference. These random groups may be homogenous and have highly conflicting group preferences; for example, the type of music that should be recommended for all the people at a party.

Most of the work on modeling group preferences in GRSs is based on rating information, which may not be accurate when the rating matrix is sparse or when groups are large. Many researchers have attempted to solve this problem but have only focused on building complex individual preferences by introducing additional information, such as social network information, tags or context information, to depict member interaction or personality [7, 11-14]. However, there is no generally-accepted additional information available across application domains, and in many scenarios there is no opportunity to access additional information about members in a random group.

The type of group affects the design of the GRS, and a major issue in making recommendations to random groups is the conflict of preferences that arises when members pursue their individual preferences without considering those of other members. This problem worsens when larger random groups are involved, because finding a compromise for diverse interests is more difficult to model, and recommendations are consequently more difficult to produce. An appropriate solution to reduce the conflict is to consider and numerically evaluate the relationships between group and individual members and to model the group profile according to the preferences of the representative members. The preferences of more representative members outweigh those of less representative members, which ensures that GRSs are able to build a high level of compromise between group profiles. However, similar to tackle preference conflicts, most of the work on computing representative preferences requires additional information, such as social relationships or trust networks [15].

This study aims to develop a Group Recommendation approach which can maximize satisfaction within random groups by modeling preferences through the analysis of contributed member ratings alone. Our proposal measures each member's importance in terms of the sub-rating matrix which makes it practical even when the matrix is highly incomplete and sparse. This approach consists of two main phases: (1) a group profile generator and (2) a recommendations generator. We first propose a member contribution score (MCS) model for Phase 1. In Phase 2, a Manhattan distance-based local average rating (MLA) model is developed to address the fat tail problem by estimating group ratings on a reduced set of items which are close to the target item. By integrating the MCS and MLA models, a Member Contribution-based Group Recommendation (MC-GR) Approach is developed. Lastly, a group recommender system and its application in online tourist groups is presented.

The contributions of this paper can be summarized as follows:

1) A member contribution-based group recommendation (MC-GR) approach is proposed to tackle the general group recommendation problem in which the group profile is generated according to member contributions, considering only the rating information without the need for additional information. Experimental results show that this approach significantly outperforms comparable baselines.

2) An MCS model is developed to measure member contributions in terms of a sub-rating matrix in which separable non-negative matrix factorization (SNMF) is employed to identify representative members and calculate corresponding contributions to the group profile. The group profile can thus be modeled accurately even when the rating matrix is highly incomplete and sparse.

3) A Manhattan distance-based model is presented to capture the local approximation of the group average rating and improve prediction accuracy, thus alleviating the potential fat tail problem.

The rest of this paper is organized as follows. We review individual recommendation approaches and the key improvements to group recommendation approaches in Section 2. Section 3 presents our MC-GR approach in detail. The experiments and results analysis are demonstrated in Section 4. A group recommender system, GroTo, is developed for web-based tourist groups, and its framework is shown in Section 5. The conclusion and further study are presented in Section 6.

2. Literature Review

In this section, we present the two general approaches to generating individual and group recommendations. We review both types of approach, because to aggregate individual recommendations or build a group profile requires knowledge of individual recommendation approaches. We also present several detailed methods related to these approaches, and follow with a discussion of these methods and the limitations of existing methods.

2.1 Individual Collaborative Filtering-based Approaches

Most GRSs allow users to specify their preferences as scalar ratings (e.g. from 1 to 5) or binary ratings (e.g. thumb for posts). Collaborative filtering (CF) techniques [16], which rely on ratings, are widely applied in GRSs. Some advanced individual recommendation approaches [17] are beyond the scope of this paper and will not be introduced; rather, we review the two most popular families of CF recommendation approaches: item-based CF (ICF) and user-based CF (UCF). ICF approaches recommend items similar to a user's previously preferred items [18], while UCF approaches recommend items preferred by people who have common interests. The unknown ranks can be predicted by aggregation methods such as weighted average, average z-score and average deviation from mean [19, 20].

2.1.1 Item-based Approaches

ICF approaches first measure the pairwise similarities between items. Once these similarities have been obtained, unknown ratings can be predicted and items which are similar to past preferred items can be identified. ICF approaches aim to recommend the top-*k* closest items, as shown in Equation 1. We show that, to predict the unobserved rating $r_{u,i}$ for user $u \in U$ of item $i \in I$, $r_{u,i}$ can be estimated by the weighted average of the observed ratings of *u* weighted by the corresponding item similarities. We can easily make suggestions when *u* has rated enough items to model their preference.

$$r_{u,i} = \overline{r_i} + \frac{\sum(r_{u,j} - \overline{r_j}) \times Similarity(i,j)}{\sum|Similarity(i,j)|} \tag{1}$$

2.1.2 User-based Approaches

By contrast, UCF approaches first measure the similarities between users. The unobserved rating $r_{u,i}$, which is derived from user *u* for item *i*, is predicted by ratings from users who share similar preferences to *u*. The prediction equation is shown in Equation 2.

$$r_{u,i} = \overline{r_u} + \frac{\sum(r_{v,i} - \overline{r_v}) \times Similarity(u,v)}{\sum|Similarity(u,v)|} \tag{2}$$

ICF and UCF are also called neighbor-based approaches, because they identify similar items or users respectively. Clearly, once we can model a pseudo user whose profile represents the preference of the whole group, the UCF approach can be used to generate group recommendations.

2.2 Group Recommendation Approach

The group-defining procedure can be active or passive according to the application scenario. Some scenarios allow users to actively announce that they are in a specific group, while in others, users are passively allocated to a group. For example, members in a reading group actively form the group and then obtain book recommendations for all members. On

the other hand, when people passively become a group as a result of attending a music show, recommendations for other music shows cannot be determined simply on the basis of that single attendance. In either case, a group recommender system can be defined as R, when it provides generalized items, such as books or music, for system users. The system then determines all the members in the group and makes recommendations for them as a single entity after the group has been formed. We denote all the items in R as I and all the users as U, and a group as G, in which $G \in U$ is a collection of the users gathered actively (e.g. people who choose the same reading group) or passively (e.g. people who attend a show) while their preferences or profiles are collected by R. The group recommender system can be represented as three tuples $\langle R, G, S \rangle$ that select a number of items S of which $S \in I$ matches as many preferences of G as possible.

Many GRSs are reviewed in [21] and, generally, most existing recommendation approaches in GRSs can be classified into two categories, as illustrated in Figure 1: (a) aggregating individual preferences, in which the profile of a pseudo user is modeled by aggregating individual members' preferences to represent the preferences of the whole group, and the pseudo user's profile is then used to generate group recommendations; (b) aggregating individual recommendations, in which individual members' recommendations are generated independently and group recommendations are produced by aggregating individual recommendations [22]. These two categories of approach are compared in [23], and it is suggested that the former approach is slightly better than the latter. The challenge of the pseudo user approach is that group members may not always share the same preferences, and preference conflicts may occur when a group profile is modeled to represent the preference of all the group members. In general, many strategies are required to alleviate and minimize the dissatisfaction caused by preference conflicts.
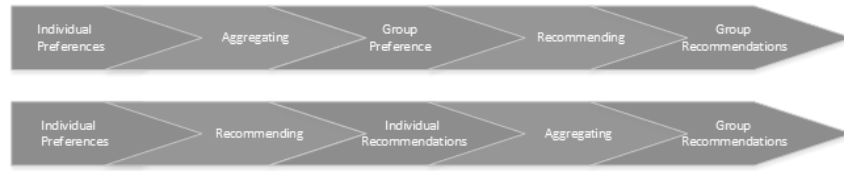
Figure 1. The two basic approaches to making group recommendations. The top approach aggregates individual preferences and the bottom approach aggregates individual recommendations.

Many strategies have been employed to model the group profile, most of which are summarized in [24]. We classify these strategies into four categories to reflect the different points of view.

- Consensus-based: Fairness, Average

- Majority-based: Plurality Voting

- Borderline: Least Misery, Most Pleasure

- Dictatorship: Most Respected Person

Of these four categories, majority-based strategies are often used to aggregate individual recommendations, while the other three categories are used to aggregate individual preferences to build a group profile. As previously mentioned, two aspects are of key concern in modeling the group profile. The first is the common interest of the group and the second is the disappointment caused by preference conflicts; these two aspects drive the basic design principles for generating group recommendations: maximizing satisfaction and minimizing disappointment. Consensus-based, Majority-based and Dictatorship strategies are widely used to maximize satisfaction; for example, the average strategy in [25], and the variation on average strategy used in [26] to aggregate rankings. Borderline strategies, such as the least misery strategy in [27], are used to minimize disappointment. A combined strategy called "average without misery" has been proposed [8, 28] which balances the two principles by taking both aspects into consideration. However, this strategy needs to determine a threshold that will explicitly exclude members who do not meet requirements.

Let a group be G and any user belonging to G be u, $Profile_G$ and $Profile_u$ are corresponding profiles. Strategies can be summarized as Equation 3.

$$Profile_G = \sum_{u \in G} \vec{\omega}_u \, Profile_u, \qquad\qquad (3)$$

where $\vec{\omega}_u$ is the weight vector for u and different $\vec{\omega}_u$ leading to different strategies. When $\vec{\omega}_u = {}^1/_{|G|}$, Equation 3 becomes an average strategy. When only one member's weight vector's elements are equal to 1 and other members' are zero vectors, Equation 3 becomes a dictatorship strategy.

Many improvements have been proposed by providing complex models to calculate weights. To model the group profile for a complex group, many systems need extra information, such as social relations or tags. Social relations are used in [29] to identify the most representative measures. The basic idea behind this is that users tend to purchase those products that are preferred by the user's social contacts. A group recommendation method was proposed by [30, 31] which combines both the social and content interests of the group members. Group profiles with tags were built by [11, 32, 33]. These improvements suffer from the problem that they do not work when the required additional information is unavailable, and the problem may be worse when a random group is involved. For example, it is difficult to identify the social relationships between a group of strangers on an airplane, and it is not feasible for passengers to tag their preferences in advance. Other extra information, such as domain knowledge, is also applied in modeling group profiles. User prototypes for tourism activities were predefined by [9, 34] to model the pseudo user profile for a random group, which was demonstrably useful; however, it was also necessary to introduce domain knowledge into the system. A more complex example including domain knowledge is presented in [35], in which three support vector machines are trained for the different preference aspects of TV viewers. The overall viewing preference is constructed by combining three aspects with case-based reasoning. Note that the dictatorship strategies we

have mentioned all depend on incorporating extra information. In [23], for example, the family-log model weights users by their number of ratings. All such improvements need additional information to incorporate with ratings.

Another way to improve the effectiveness of modeling a group profile is to provide interactive functions for group members to explicitly specify their preferences [36, 37], but these functions are not always available when a group is formed randomly.

2.3 Matrix Factorization

Traditional matrix factorization (MF), including non-negative matrix factorization (NMF), is not new in the recommender system domain. The original rating matrix is noted as R, and MF decomposes R into two matrixes, U and V, in which R is represented by multiplying of U and V, $R' = UV$. After obtaining U and V, the unknown rating can be predicted. The difference between R and R' is the recovery error, and in most cases MF aims to minimize this error to generate more precise predictions. In GRSs, there are generally massive missing ratings which cause difficulty in appropriately modeling the group profile. Singular value decomposition (SVD) is employed in [38] to decompose the rating matrix and model the group profile by aggregating the decomposed user profiles. However, this method suffers from the strategy selection problem when modeling the group profile. It is important to note that no stable solution is guaranteed by traditional matrix factorization because factorization results are significantly affected by initial values and matrix update protocols, therefore it is not possible to build a stable and unique group profile. Another problem is the high computational cost and low quality when faced with high dimensional sparse data, which makes it impractical for real recommender scenarios.

3. Member Contribution-based Group Recommendation Approach

In this section, we first introduce the notion of a member contribution score (MCS) and a novel MCS-based group recommendation approach that generates recommendations with a

low level of disappointment among members. As illustrated in Figure 2, the approach consists of two main phases: (1) a group profile generator and (2) a recommendations generator. In the group profile generator, an MCS-based model is proposed to generate the group profile, and in the recommendations generator, a Manhattan distance-based model is proposed to generate group recommendations. The models are described in the following subsections.

- *Group profile generator*: instead of treating all individual members equally when aggregating the group profile, we argue that the group members contribute differently according to their representative status. Hence, we introduce a notion of a measure to evaluate the representative level of each member and propose a member contribution score (MCS) model to numerically define the representative status of members and generate the group profile considering the MCS results. The group profile is then used as the input for the individual recommendation approach for generating group recommendations.
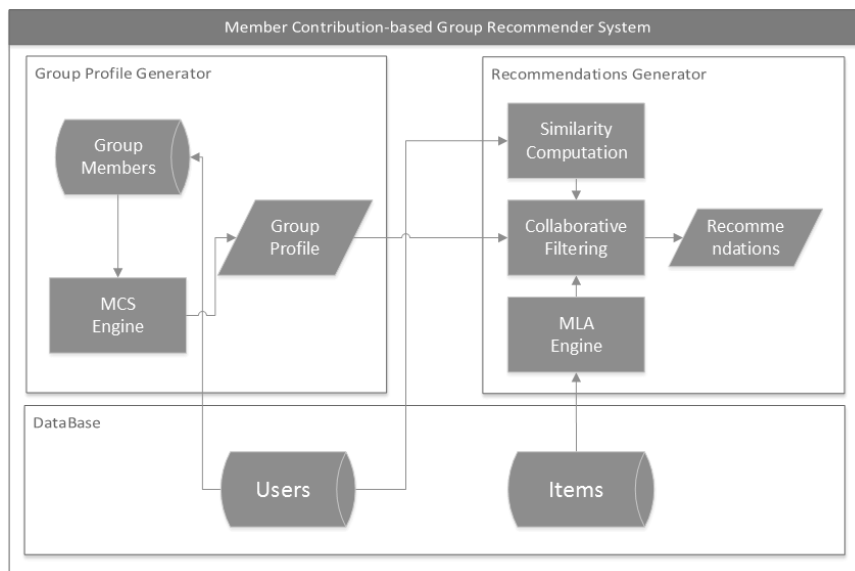

Figure 2. System architecture

- *Recommendations generator* we argue that a local average rating focusing on the target item can alleviate the low accuracy problem caused by fat tail ratings. A Manhattan

distance-based local average rating (MLA) model is proposed in which a reduced set of items relevant to the target item is first identified by Manhattan distance-based measure and the local average rating over the reduced set is calculated. Using the group profile and the local average ratings, the group ratings are predicted by the individual recommendation approach and the top-k items are recommended according to the group predictions.
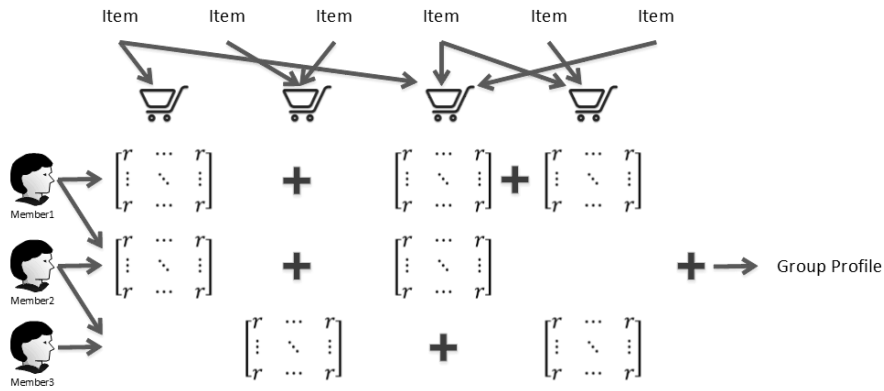
3.1 Group Profile Generator

Figure 3. Description of the MCS model

This phase includes the necessary computations to generate a single profile to represent the overall preference. To build the profile for a group, especially a complex random group, the preferences of the most representative members should be considered above others. Unfortunately, these representative members are difficult to identify because of the uncertainty in the system, i.e. the sparsity. To address this problem, we argue that sampling and aggregating architecture over the item space is employed. The rating vectors of users can be perceived as high dimensional data, with each dimension representing one item. For example, suppose a movie recommender system consists of only four movies. A rating vector of user u is $vector = [r_{u,1}, r_{u,2}, r_{u,3}, r_{u,4}]$ and the dimensions are the ratings of each movie. Instead of considering the vectors over the whole item space, sampling selects a reduced set of items for which members can provide a rating matrix without missing values, after which

the representative members can be precisely evaluated on this partial rating matrix. After multiple sampling, the representative members across the global item space can be approximated by aggregating the results from all the samplings. In our work, this sampling and aggregation process is implemented in the MCS model. Note that, in our work, no side information is needed. A high level of the group profile compromise equation with respect to MCS can then be written as

$$Profile_G = \sum_{u \in G} MCS_u\, Profile_u. \tag{4}$$

Equation 4 is similar to Equation 3, with the important difference that in Equation 4, the MCS model is much more concerned about contribution differences than strategy selection.

The steps included in this phase are illustrated in Figure 3 and described in detail below.

Step 1: computing contribution in one sampling

Traditionally, the contribution of a member to the group profile is highly correlated with the strategy adopted by the system. For instance, when adopting the least misery strategy, the member who gives the lowest rating for each item is selected out to build the group profile. When rating matrixes are incomplete, the massive missing values make it difficult to generate the group profile according to a specific strategy. An example of a group is shown in Table 1. The unknown ratings for each item make the generated group profile less reliable. One method of addressing this problem is matrix completion, which predicts missing ratings before building the group profile and introduces new uncertainty to the system. Rather than concerning ourselves with a specific strategy, we focus in this work on the representative members. A sampling consists of a projection of a rating matrix and corresponding members who have no missing values. When there are no missing values, the representative members can be measured precisely.

A sampling is noted as I_s, and any item that belongs to it is randomly selected out with equal probability. To select out members, which are denoted U_s, corresponding to I_s, filtering

is carried out to exclude members who have missing ratings on I_s. After we have determined I_s and U_s, the partial rating matrix M_s, which has no missing value, can be projected from the original matrix M to items belonging to I_s and users belonging to U_s. M_s is used as the input of the MCS model to calculate the representative members in U_s.

Intuitively, a member is not representative when his/her preference is highly correlated with and can be represented by the preferences of others. Taking all the members' profiles as data in high dimensional vectors, a finite set of vertices can be selected to define a convex hull and all the other data in the convex hull can be linearly represented. These vertices, i.e. preferences, are more representative than the preferences in the convex hull. This is the motivation for our proposal of the "contribution score" concept to depict the representative degree of a member. Taking this point of view, the representative measuring problem in our work is converted into the identification of the set of preferences on hull vertices.

Once we have obtained the partial rating matrix M_s, separable non-negative matrix factorization (SNMF) [39, 40] is employed in the MCS model to identify the vertices. Compared to traditional matrix factorization techniques that rely heavily on initial values, a stable solution and representative degrees can be guaranteed by the SNMF. The SNMF on M_s is defined as

$$M_s = W M_s^B, \tag{5}$$

where M_s^B is the basic matrix, W is the weight matrix. The basic matrix M_s^B consists of a number of rows from M_s which can be used to recover M_s more accurately than other rows. Members with profiles in the basic matrix can be seen as representative members. The representative member set for U_s is denoted as U_s^B.

MCS measures the representative preferences of each member according to Equation 6. For each member $u \in U_s$, the MCS of u is defined as

$$MCS_u^{I_s} = \begin{cases} 1, & u \in U_s^B \\ 0, & otherwise \end{cases}. \tag{6}$$

Step 2: aggregating MCSs for each member

Note that each sampling selects a portion of items and only selects members who have no missing ratings. Members involved in one sampling may not cover the group. The representative degree of a group member over the whole item space can be approximated by aggregating the MCS results of all the samplings in which he/she is involved.

Theoretically, all the possible samplings should be considered to evaluate each member's contribution accurately. However, in practice, it is impossible to complete this task within the time limit, considering the infinite projection probability for a rating matrix of a group. To address this issue, a portion of the projections for any group size are selected out as the samplings, i.e. all the item-pair subspaces, to measure MCS. For the whole item space I, the ith sampling is S_i $i \in [1 \dots d]$. The items and members involved in S_i are I_{s_i} and U_{s_i} respectively, and the MCS of each member u, irrespective of whether or not it is involved in G on S_i, can be represented as

$$MCS_u^{S_i} = \begin{cases} MCS_u^{S_i}, & u \in U_{s_i} \\ 0, & otherwise \end{cases}. \tag{7}$$

The contribution of member u, $u \in U_{s_i}$, is 1 or 0 depending on whether u can be identified as representative members, and is always 0 when u is not involved. For each group member $u \in G$, we can aggregate all the MCS of u on all I_{s_i} into a single MCS.

$$MCS_u = \sum_{i=1}^{d} \frac{2}{n} MCS_u^{S_i}, \tag{8}$$

[41], proposed an efficient method to resolve the SNMF problem. Equation (9) is the specific SNMF method employed to compute the representative members by the maximum and minimum angles between the 2D random projections of the n data points and the horizontal axis in a 2D plane which match the item-pair projection,

$$MCS_u^{S_i} = \begin{cases} 1, & u = \arg\max_i arctan2(u\beta_{sub_{j1}}^T, u\beta_{sub_{j2}}^T) \\ 0, & else \end{cases}, \tag{9}$$

where $\beta_{sub_{j1}}$ and $\beta_{sub_{j2}}$ are two unit vectors of the plane and

$$arctan2(y,x) = \begin{cases} \arctan(y/x), & x > 0; \\ \arctan(y/x) + \pi, & y \geq 0, x < 0; \\ \arctan(y/x) - \pi, & y < 0, x < 0; \\ \pi/2, & y > 0, x = 0; \\ -\pi/2, & y < 0, x = 0. \end{cases} \qquad (10)$$

Let I_G be all the items that have been rated at least once by a group member. The number of all possible subspaces is $C_{|I_G|}^2$. In practical systems, users tend to give a small number of ratings and $C_{|I_G|}^2$ should be acceptable. Combined with Equation 8, the final MCS is defined as

$$MCS_u = \sum_{i=1}^{C_{|I_G|}^2} \frac{2}{n} MCS_u^{S_i} \qquad (11)$$

Step 3: aggregating group profile according to MCS

Once the MCS for each group member has been obtained, it is normalized for further group profile calculation.

$$\omega_u = \frac{MCS_u}{\sum_{u \in G} MCS_u}. \qquad (12)$$

The group profile $Profile_G$ is represented as a vector and every dimension represents an item only when it has been rated by group members. Let I_G be all the items that have been rated by group members, $item_i \in I_G$ $i \in [1..|I_G|]$, $Profile_G = [r_{G,item_1}, r_{G,item_2}, \ldots, r_{G,item_{|I_G|}}]$. For each $item_i$, let U_{item_i} be all the members who have rated $item_i$, then the group rating for $item_i$ is computed as follows:

$$r_{G,item_i} = \sum_{u \in U_{item_i}} \omega_u r_{u,item_i}. \qquad (13)$$

In Algorithm 1, we summarize these three steps to show how to compute the group profile and give a detailed description of the MCS calculation. We give a numerical case of Algorithm 1 in Example 1.

Algorithm 1 Group profile generator algorithm

Input: rating matrix M, Item set I, group G

Output: group profile of G

[Begin]

  Set all the members' $MCS_{u \in G} = 0$

  Get the item set I_G that has been rated by members

  For $sub = 1$ to $C_{|I_G|}^2$ do

    Step1:

      Get all the members involved in U_{sub}

      Get items involved in I_{sub}

$$MCS_{u \in U_{sub}}^{I_{sub}} = \begin{cases} 1, & u = \arg\max_i arctan2(u\beta_{sub_j}^T, u\beta_{sub_j}^T) \\ 0, & else \end{cases}$$

    Step2:

    For each member in U_{sub}, aggregate MCS_u

$$MCS_{u \in U_{sub}} = MCS_{u \in U_{sub}} + \frac{|I_{sub}|}{|I_G|} MCS_{u \in U_{sub}}^{I_{sub}}$$

  End for

  Step3:

    Normalize $MCS_{u \in G}$ to $\omega_{u \in G}$

    For item i in I_G

      Group rating $r_{G,i} = \sum_{u \in U_i} \omega_u r_{u,i}$

    End for

    Group profile $Profile_G = [r_{G,i}], i \in I_G$

[End]

Example 1: Let $G = \{User_1, User_2, User_3, User_4\}$, $I_G = \{Item_1, Item_2, Item_3, Item_4\}$, as

shown in Table 1.

	$Item_1$	$Item_2$	$Item_3$	$Item_4$
$User_1$	5	4	4	?
$User_2$	4	4	?	?
$User_3$	?	?	5	2
$User_4$	3	1	?	3

All the item subspaces are sampled: $\{I_{1,2}, I_{1,3}, I_{1,4}, I_{2,3}, I_{2,4}, I_{3,4}\}$. After calculating MCS in

the samplings, i.e. step 1, we have $MCS_{user4}^{I_{1,2}}=1$, $MCS_{user1}^{I_{1,3}}=1$, $MCS_{user4}^{I_{1,4}}=1$, $MCS_{user1}^{I_{2,3}}=1$,

$MCS_{user4}^{I_{2,4}}=1$ and $MCS_{user3}^{I_{3,4}}=1$.

In $I_{1,2}$, only $user4$'s contribution is 1 when $user1$ and $user2$ are 0, shows that not all the members involving in an sampling means they are representative in it.

In step 2, we aggregate MCS from all the samplings.

$$MCS_{user1} = \frac{2}{4}\left(MCS_{user1}^{I_{1,2}} + MCS_{user1}^{I_{1,3}} + MCS_{user1}^{I_{1,4}} + MCS_{user1}^{I_{2,3}} + MCS_{user1}^{I_{2,4}} + MCS_{user1}^{I_{3,4}}\right)$$

$$= \frac{2}{4}(0 + 1 + 0 + 1 + 0 + 0) = 1$$

$$MCS_{user2} = \frac{2}{4}(0 + 0 + 0 + 0 + 0 + 0) = 0$$

$$MCS_{user3} = \frac{2}{4}(0 + 0 + 0 + 0 + 0 + 1) = \frac{1}{2}$$

$$MCS_{user4} = \frac{2}{4}(1 + 0 + 1 + 0 + 1 + 0) = \frac{3}{2}$$

In step 3, contribution is normalized and the group profile is then modeled taking member contributions into consideration.

$$\omega_{user1} = \frac{MCS_{user1}}{MCS_{user1} + MCS_{user2} + MCS_{user3} + MCS_{user4}} = \frac{1}{1 + \frac{1}{2} + \frac{3}{2}} = 1/3$$

$$\omega_{user2} = \frac{0}{1 + \frac{1}{2} + \frac{3}{2}} = 0$$

$$\omega_{user3} = \frac{1/2}{1 + \frac{1}{2} + \frac{3}{2}} = 1/6$$

$$\omega_{user3} = \frac{3/2}{1 + \frac{1}{2} + \frac{3}{2}} = 1/2$$

$$r_{G,item1} = \omega_{user1}r_{user1,item1} + \omega_{user2}r_{user2,item1} + \omega_{user4}r_{user4,item1}$$

$$= \frac{1}{3} \times 5 + 0 \times 4 + \frac{1}{2} \times 5 = 3.17$$

$$r_{G,item2} = \frac{1}{3} \times 4 + 0 \times 4 + \frac{1}{2} \times 1 = 1.83$$

$$r_{G,item3} = \frac{1}{3} \times 4 + \frac{1}{6} \times 5 = 2.17$$

$$r_{G,item4} = \frac{1}{6} \times 2 + \frac{1}{2} \times 3 = 0.83$$

The output group profile is $Profile_G = [3.17, 1.83, 2.17, 0.83]$.

3.2 Recommendations generator

The group profile is expressed as the group rating vector over the items that have been rated by group members. Once this has been generated, it is used in this phase to predict unknown group ratings. A similarity measure, PCC similarity, is adopted in our work to identify neighbors close to the group. To minimize the error caused by the fat tail, a Manhattan distance-based measure is applied to compute local average ratings for the group. By combining PCC similarities and local average ratings, a user-based individual collaborative filtering approach is applied to predict unknown group ratings.

Step 1: compute profile similarity between group members and non-group members

Once the group profile, $profile_G$, has been obtained, it can be seen as a preference of a pseudo user. It is then possible to compute the similarities between the pseudo user and non-member group users. Pearson Correlation Coefficient (PCC), which has been widely used in a number of recommendation systems, is employed for the similarity computation. Let *g* be the pseudo user and *u* be a non-group system user. Let I_u be the item set that has been rated by *u*. The PCC similarity between *g* and *u* is computed based on their common ratings as follows:

$$Sim(g,u) = \frac{\sum_{i \in (I_G \cap I_u)} (r_{g,i} - \overline{r_g})(r_{u,i} - \overline{r_u})}{\sqrt{\sum_{i \in (I_G \cap I_u)} (r_{g,i} - \overline{r_g})^2 \sum_{i \in (I_G \cap I_u)} (r_{u,i} - \overline{r_u})^2}}, \tag{14}$$

where $I_G \cap I_u$ is the set of common rated items by both *g* and *u*, $r_{g,i}$ and $r_{u,i}$ represent known ratings for item *i*, $\overline{r_g}$ is the average rating of *g* and $\overline{r_u}$ is the average rating of *u*. The similarity $Sim(g,u)$ between two users ranges from -1 to 1, where a large value indicates a higher similarity.

Step 2: compute group local average rating

Before we adopt user-based collaborative filtering to predict unknown group ratings, the fat tail issue should be considered. Neighbor-based collaborative filtering methods may suffer from the fat tail distribution of user ratings, i.e. many ratings are far away from the pseudo user's mean rating. Equation 2 is widely employed to predict the unknown ratings, and it is difficult to predict fat tail target item ratings because too many unrelated items are used to calculate the average rating. A model to obtain a local average rating focus on the target item can alleviate the fat tail problem.

We propose a MLA model to estimate the pseudo user's local average ratings with respect to the target items instead of computing the global average rating on the whole item space. The relevance of a specific target item in relation to other items is ranked by a Manhattan distance-based measure. By locating items with similar distribution using this measure, the local average rating of a member on these items can be seen as a closer estimation of target item than the global measure, therefore the local average rating can be used to predict the unknown group ratings to alleviate the fat tail problem. After computing the item's relevance, we select a portion of the ratings with higher relevance to compute the local average rating.

In our recent work [42], we used the Manhattan distance-based function to measure the relevance between two users. Here, we use the same method to measure the relevance of two items. Item relevance aims to find items that are close to the target item even when fat tail exists. Manhattan distance is sensitive to fat tail because it measures relevance according to absolute rating differences.

Let $r_i = (r_{1,i}, r_{2,i} \cdots r_{k,i})$ and $r_j = (r_{1,j}, r_{2,j} \cdots r_{k,j})$ represent the rating vectors of the target item i and the non-target item j on common k users. Let Δr_p, $p \in [1..k]$ be the absolute

difference of $r_{p,i}$ and $r_{p,j}$, as $\Delta r_p = |r_{p,i} - r_{p,j}|$. The Manhattan distance of these two vectors is $D_{i,j} = ||r_i - r_j||_1 = \sum_{p=1}^{k} \Delta r_p$. One limitation of this distance is that it is difficult to achieve a unified threshold for different systems, because one might choose a rating scale of 1 to 5 while another might choose a scale of 1 to 10. We first normalize Δr_p before calculating relevance. We divide the possible Δr_p into three relevance levels with respect to the different systems. If the system allows a user to rate an item from r_{MIN} to r_{MAX}, then Δr_p ranges from 0 to r_{MAX} - r_{MIN} and the three levels are $[0, \frac{(r_{MAX} - r_{MIN})}{3})$, $[\frac{(r_{MAX} - r_{MIN})}{3}, \frac{2(r_{MAX} - r_{MIN})}{3})$ and $[\frac{2(r_{MAX} - r_{MIN})}{3}, r_{MAX} - r_{MIN}]$. Clearly, if Δr_p is more close to 0, i and j are more relevant to user p. We define a subsection function to represent the relevance for each level.

$$\Delta r_p^* = \begin{cases} 1, & 0 \leq \Delta r_i < (r_{MAX} - r_{MIN})/3; \\ 0.5, & (r_{MAX} - r_{MIN})/3 \leq \Delta r_i < 2(r_{MAX} - r_{MIN})/3; \\ 0, & 2(r_{MAX} - r_{MIN})/3 \leq \Delta r_i < (r_{MAX} - r_{MIN}). \end{cases} \quad (15)$$

The Manhattan distance of i and j becomes $D_{i,j} = \sum_{p=1}^{k} \Delta r_p^*$. Evidently, $D_{i,j}$ can vary from 0 to k, the number of dimensions of the rating vector, and different k give these distances a different scale across all users. We address this problem by averaging the Manhattan distance as the final relevance of items i and j.

$$Rel_{i,j} = \frac{D_{i,j}}{k} \quad (16)$$

A threshold T is set to determine whether items i and j are sufficiently close and whether $r_{u,j}$ is taken into consideration to compute the average rating of group G for item i. The local average rating of group G for target item i is calculated by averaging all the ratings relevant to i, where the relevance is greater than T, which is defined as Equation 17.

$$\overline{r_{G,i}}' = avg(r_{G,j}) \ if \ Rel_{i,j} \geq T \quad (17)$$

Step 3: predict group rating for items

After obtaining the similarities and local average ratings of the pseudo user, we can predict the unknown group ratings. In our work, user-based collaborative filtering is adopted, and unknown group ratings are calculated by the weighted sum of deviations from the average rating of similar neighbors. Let $r_{G,i}$ be the unknown group rating for item i, and $i \notin I_G$, and $r_{G,i}$ can be computed by Equation 18.

$$r_{G,i} = \overline{r_{G,i}}' + \frac{\sum_{u \in Neighbors}(r_{u,i} - \overline{r_u}) \times Sim(g,u)}{\sum_{u \in Neighbors}|Sim(g,u)|}, \tag{18}$$

where $\overline{r_{G,i}}'$ denotes the local average rating of group G obtained by the MLA model, and neighbors close to the pseudo user are selected out according to PCC similarity. User v is one of the neighbor users of active user u and $\overline{r_v}$ is the corresponding average rating of v. The final recommendations are selected as the top-k items with the highest predictions.

In Algorithm 2, we summarize these three steps to show how to generate group recommendations given the group profile, and give a detailed description of the local average rating computation.

Algorithm 2 Recommendations generator algorithm

Input: rating matrix M, group profile $Profile_G$, parameter for local average rating T

Output: Recommendations

[Begin]

  Get the item set I_G that has been rated by members

  For $i \notin I_G$ do

    Step 1: similarities between the pseudo user g and non-member user u are computed by PCC similarity

$$Sim(g,u) = \frac{\sum_{i \in (I_G \cap I_u)}(r_{g,i}-\overline{r_g})(r_{u,i}-\overline{r_u})}{\sqrt{\sum_{i \in (I_G \cap I_u)}(r_{g,i}-\overline{r_g})^2 \sum_{i \in (I_G \cap I_u)}(r_{u,i}-\overline{r_u})^2}}$$

    Step 2: calculate local average rating

      $Rel_{i,j}$ is mean normalized Manhattan distance over whole user space.

      Local average rating $\overline{r_{G,i}}' = avg(r_{G,j})\ if\ Rel_{i,j} \geq T$

    Step 3: using traditional user-based collaborative filtering to predict unknown group ratings

$$r_{G,i} = \overline{r_{G,i}}' + \frac{\sum_{u \in Neighbors}(r_{u,i}-\overline{r_u}) \times Sim(g,u)}{\sum_{u \in Neighbors}|Sim(g,u)|}$$

  End for

  Recommendations are selected out with higher group rating

[End]

In summary, we propose a new model to build the group profile considering the degree of representation. In addition, the proposal alleviates the preference conflict issue by using local average rating.

4. Experiments and Result Analysis

In this section, we present an empirical study of our approach on real datasets. The datasets are introduced and pre-processed in Section 4.1. The group generating protocol is described in Section 4.2. The metrics employed to evaluate the performance of the proposal are shown in Section 4.3. The comparison between results from the proposal and baseline methods and discussion are presented in Section 4.4.

4.1 Datasets and Pre-processing

To the best of our knowledge, no benchmark datasets have been designed and implemented to assess the performance of group recommendations. For this reason, we employ MovieLens datasets (http://www.grouplens.org) and the Jester dataset, which are benchmark datasets that can be employed to assess individual recommendation methods and develop offline experiments. MovieLens datasets contain integer ratings and tags applied to movies by users of an online recommender service and were collected by the GroupLens Research Project at the University of Minnesota. We employ the ML100K and ML1M MovieLens datasets to evaluate performance. Jester asks users to rate jokes that enable Jester datasets to contain real number ratings. In experiments, only users who have rated between 15 and 35 jokes are selected to avoid the group profile covering all the items. The key statistics of these three sets are shown below:

dataset	User	Item	Rating	Sparsity	RatingRange
ML100K	943	1682	100,000	93.7%	1-5
ML1M	6040	3706	1,000,209	95.5%	1-5
Jester	24938	100	616912	75.3%	-10.0-10.0

The datasets are each split into two parts consisting of a training dataset and a test dataset. For items that have been rated by group members but will not be recommended to the group again, a small test set may cause fewer recommended items to be found in the test dataset, leading to poor quality evaluation. Therefore, we randomly select 50% of the data for the training set and 50% for the test set.

Note that because these datasets are composed of ratings provided by individual users to assess individual recommender system performance, they contain no real group information. Therefore, if we want to evaluate our approach, a group generating protocol and appropriate metric are needed.

4.2 Group Generation Protocol

Although the MC-GR method may handle all kind of groups, our experiments focus on random groups which may have a higher level of preference conflict. Two important features

affect the nature of groups when they are generated: group size and internal member relevance. The larger a group is, the more difficult it is to model the group profile. Most previous group recommendation approaches have focused on relatively small groups, less than 10 in number, with which it is easy to achieve a compromise opinion. In our experiments, we initially set the group size as 5 and increase it each time by 5 until 30 is reached to assess the feasibility of our proposal for both small and large groups.

Apart from size, internal member relevance is another important feature that can affect the effectiveness of the recommendation approach. The preference conflict can be greater when a group is generated randomly, because in this case, members have no knowledge about other members. Therefore, we focus on a random group in this work. In our experiments, groups are formed by randomly selecting users who have no explicit shared preference relevance, such as people traveling on the same airplane.

4.3 Metrics

To evaluate the approach based on a list of recommendations, we adopt both Normalized Discounted Cumulative Gain (nDCG) and F measure. Widely used in information retrieval, nDCG has been adopted by many researchers to measure the performance of group recommender algorithms. It attempts to measure the rank performance between predicted group ratings and real values. F measure, which is widely employed in individual recommender systems, is also employed to evaluate accuracy by considering missing labeling data.

nDCG is more appropriate than RMSE and MAE because it not only considers accuracy but also takes recommendation order into account. Let $l_1 \cdots l_k$ be the recommendation list obtained and u be a user. DCG is defined as

$$DCG_{u,k} = r_{u,l_1} + \sum_{i=2}^{k} \frac{r_{u,l_i}}{log_2(i)}. \tag{19}$$

and the corresponding nDCG is defined as

$$nDCG_{u,k} = \frac{DCG_{u,k}}{IDCG_{u,k}}, \tag{20}$$

where IDCG is the optimal possible gain value for user u where recommendations are re-ordered in descending order based on their relevant scores in the obtained list. DCG defined in Equation 19 measures the accuracy of a list of recommendations that is ordered by score (predicted rating). An item's score will be penalized for logarithmically proportional to the position of each item in the list. nDCG can then be used to measure the performance of the recommendation list. Clearly, given that nDCG ranges from 0 to 1, the higher the nDCG obtained, the better recommendations have been made.

As a result of the lack of ground truth required to assess the recommendations generated for the group, we calculate the average nDCG value for each of the group members. In our experiments, we compute nDCG on all the items in the test set of the user, sorted according to the ranking computed by the recommendation algorithms. In other words, we compute nDCG on the projection of the recommendation list on the test set of the users. For example, imagine that $rec = \{item_1, item_2, \ldots, item_6\}$ is an ordered list of recommendations for a group G. User u is a member of G and we can find corresponding $r_{u,item_1}$, $r_{u,item_3}$ and $r_{u,item_5}$ in the test set. We compute the nDCG score for u only by $\{item_1, item_3, item_5\}$.

F measure is used to evaluate the missing prediction and group rating classification for members. For example, let us predict the group rating for an item as 4. Let the threshold for members to accept one item be set to 3. If one member rates this item as 2, we may recommend this item whereas we should drop it. In the experiments, we use "3" as the threshold to label whether we should accept one item for both datasets rating ranging from 1 to 5. True positive (TP) for an item is when all the member ratings for that item are higher than 3 (ignoring members whose rating for this item is unknown) and the group prediction for the item is higher than 3. False negative (FN) for an item is when all the member ratings for that item are higher than 3 (ignoring members whose rating for this item is unknown) and the

group prediction for the item is lower than 3. False positive (FP) for an item is when some member ratings for that item are lower than 3 (ignoring members whose rating for this item is unknown) and group prediction for the item is higher than 3. F measure is shown in Equation 21. F ranges from 0 to 1, and similar to nDCG, the higher the F obtained, the more accurate is the group rating prediction.

$$F = 2 \; \frac{TP}{(TP+FN)} \frac{TP}{(TP+FP)} \tag{21}$$

4.4 Experiment Design

To measure the improvement of our MC-GR approach, we implement several successful and popular group recommendation approaches as baselines. Below are the labels and descriptions we use to denote each of these baselines.

LM: the group profile is generated using the least misery strategy and basic user-based CF is used to generate group recommendations. The group ratings in the group profile are calculated according to Equation 22.

$$rating_{G,i} = Min(rating_{u,i}) \tag{22}$$

AVG: the group profile is generated using the average strategy and basic user-based CF is used to generate group recommendations. The group ratings in the group profile are calculated according to Equation 23.

$$rating_{G,i} = \frac{1}{|G|}\sum_{u \in G} rating_{u,i} \tag{23}$$

AM: the group profile is generated using the average without misery strategy. This method aims to find a compromise between LM and AVG. A threshold is used to filter out items that will cause disappointment for members who have ratings lower than a predefined threshold. In our experiment, this threshold is set to 2. The group ratings in the group profile are calculated according to Equation 24. After building the group profile, basic user-based CF is used to generate group recommendations.

$$rating_{G,i} = \frac{1}{|G|}\sum_{u \in G}\{rating_{u,i} \,|\, \forall rating_{u,i} > 2\} \qquad (24)$$

MCS: the group profile is generated using weighted individual preference, and weights are computed by MCS for each member. Group recommendations are predicted using global average rating.

MC-GR (or called MCS-MLA): the group profile is generated using weighted individual preference, and weights are computed by MCS for each member. Group recommendations are predicted using the MLA model. The parameter for identifying neighbor items is set to 0.2.

For each specific group size, 1000 groups are randomly generated, and the average metrics over 1000 groups give the final result. For instance, for a 10-member group, we randomly select 50% of data as the training data and the rest as the test data. Ten members are randomly selected from users in the test set to form the group, because we want to avoid a situation in which the selected member's ratings are all in the training set and cannot be measured over the test data. We calculate the metrics and repeat this process 1000 times to obtain the average metrics.

4.5 Results and Discussion

Figure 4 shows the nDCG results obtained by LM, AVG, AM and our approach. As shown in Figure 4, it is clear that our approach, whether local average rating is used or not, consistently outperforms the baseline approaches. On the ML100K dataset, the LM, AVG and AM approaches are close when group size is relatively small, and AM is the best approach when group size increases. MCS is 2.4% better than LM when group size is 5. When group size is 30, MCS is 2.5% better than AVG, 4% better than LM, and 4.8% better than AM. Our MC-GR results on various sized groups show that local average rating significantly improves performance. Our MC-GR is 3.2% better than AVG and about 5% better than LM and AM when group size is 5. When group size is 30, our MC-GR is 3.5% better than LM, 5.7% better than AM, and 7.6% better than AM.
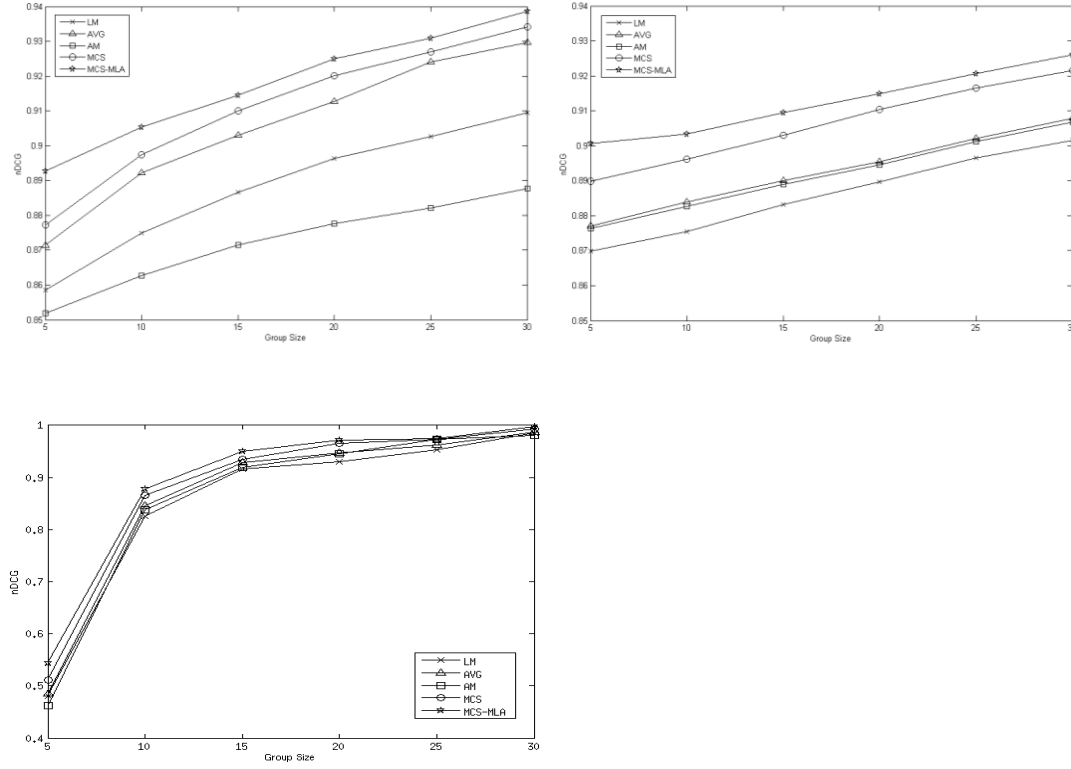
Figure 4. nDCG scores computed for the alternative implementations of our MCS and MC-GR (MCS-MLA) approaches and of the collaborative filtering approach based on LM, AVG and AM aggregation strategies on 1000 groups of each size.

On the ML1M dataset, by contrast, MCS is 1.4% better than AVG when group size is 5. When group size is 30, MCS is 1.5% better than AVG, 1.6% better than AM, and 2.2% better than LM. Our MC-GR results on various sized groups show that local average rating significantly improves performance. Our MC-GR is 2.7% better than AVG and AM, and 3.5% better than LM when group size is 5. When group size is 30, MCS-MLA is 2% better than AVG, 2.1% better than AM and 2.7% better than LM.

Even on a sparser dataset, ML1M, our MCS and our MC-GR approaches clearly make better recommendations, and the approach using our MC-GR outperforms the approach using MCS only. An interesting fact is that for the AVG strategy on ML1M, performance decreases when the group size becomes large. A reasonable explanation for this is that when there is insufficient information, it is difficult to find a fair solution for all the members.

On the Jester dataset, our approaches are better than AVG, AM and LM. When group size is 5, our MCS is 5% and MC-GR is 9 % better than AVG, while LM is close to AVG and AM is worse than AVG. When group size increases, which means the group profile covers more items and unknown ratings become less, nDCG results are close to 1 and the performance of MCS and MV-GR decreases. MCS is 0.9% better than AVG and MC-GR is 1.1% better than AVG when group size is 30.
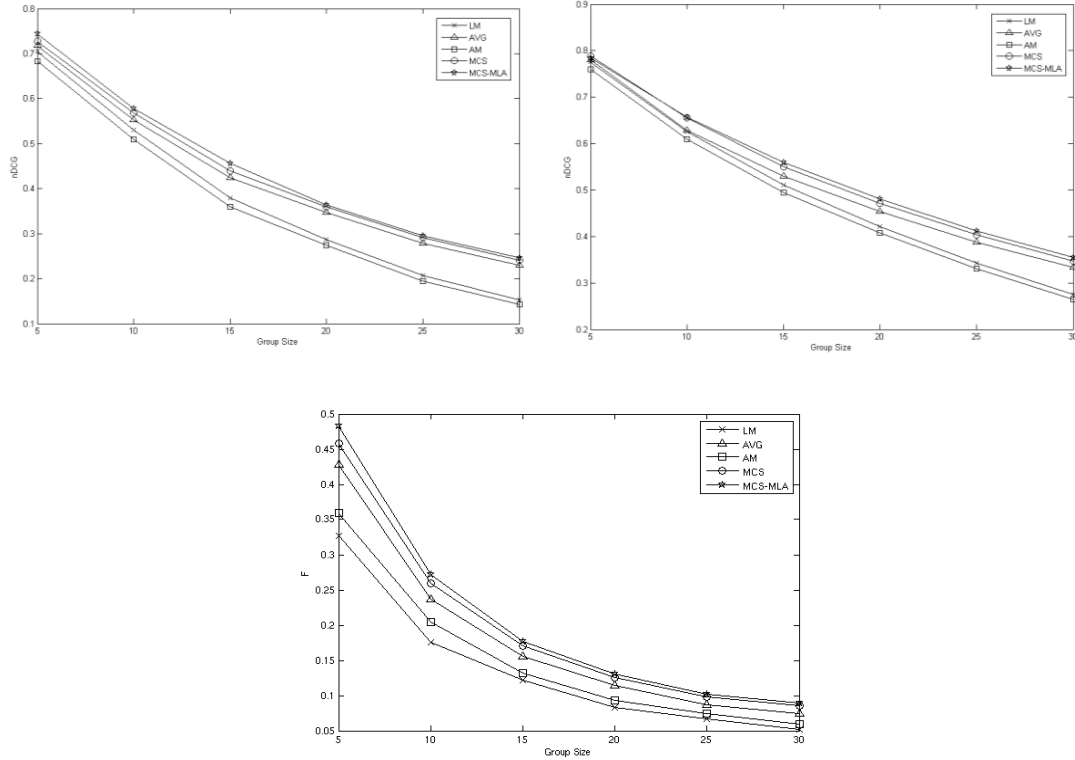

Figure 5. F scores computed for the alternative implementations of our MCS and MC-GR (MCS-MLA) approaches and of the collaborative filtering approach based on LM, AVG and AM aggregation strategies on 1000 groups of each size.

Figure 5 shows the F results obtained by LM, AVG, AM and our approach. As shown in Figure 5, AVG is the best approach when the group size is very small, i.e. 5. This is mainly because the performance would be better if the approach could correctly predict the majority opinions. When the group size becomes larger, our approach decreases more slowly than LM, AVG and AM. On the ML100K dataset with a group size of 30, MCS is 8% better than AVG, which is the best approach in LM, AVG and AM, and on the ML1M dataset, MCS is 6% better than AVG, which is also the best

approach in LM, AVG and AM. On the Jester dataset with a group size of 30, MCS is 10% better than AVG, which is also the best approach in LM, AVG and AM.

When drawing comparisons, it can be seen that nDCG consistently increases in most cases when group size becomes larger, and F consistently decreases for all approaches. The reason for this is that nDCG measures the relative rank differences for recommendations, and F measures the errors between predictions and real values.

Because a threshold is used in the MLA model to estimate the target item related average rating, we also performed several experiments to examine the sensitivity of performance with this threshold, in which we varied the value of the threshold to generate the predicted rating. The nDCG and F results of the proposal were also compared with others.

Figure 6 and Figure 7 show the results of using MCS alone, and 0.2, 0.3 and 0.4 are employed in the MCS-MLA model. From Figure 6, we can clearly observe that the F results are not greatly affected when different parameters are used in the MLA model. From Figure 7, we can see that our MC-GR approaches outperform MCS and there are no big differences when the parameter is not strictly set. We also notice that the different parameters affect the performance considerably when the group size is relatively small. This demonstrates that for random groups, local average rating tends to be an average rating when the group size is large.
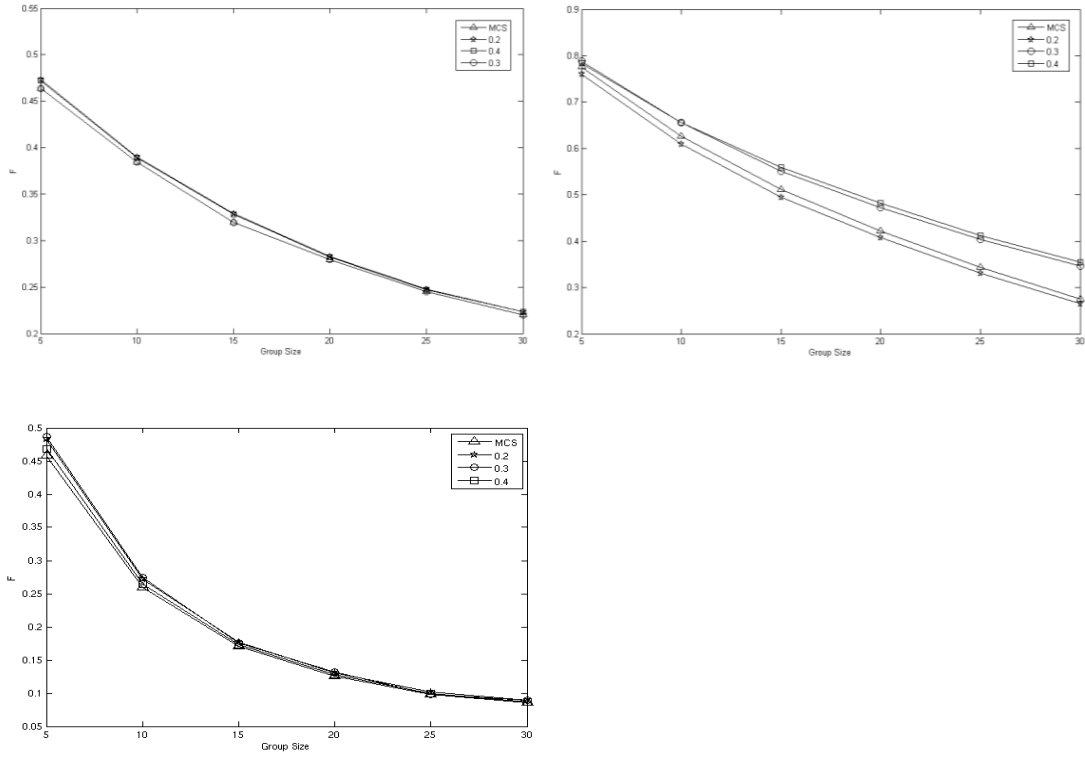
Figure 6. F results when using the MCS model alone and when combining the MLA model using 0.2, 0.3 and 0.4 thresholds to produce local average ratings.
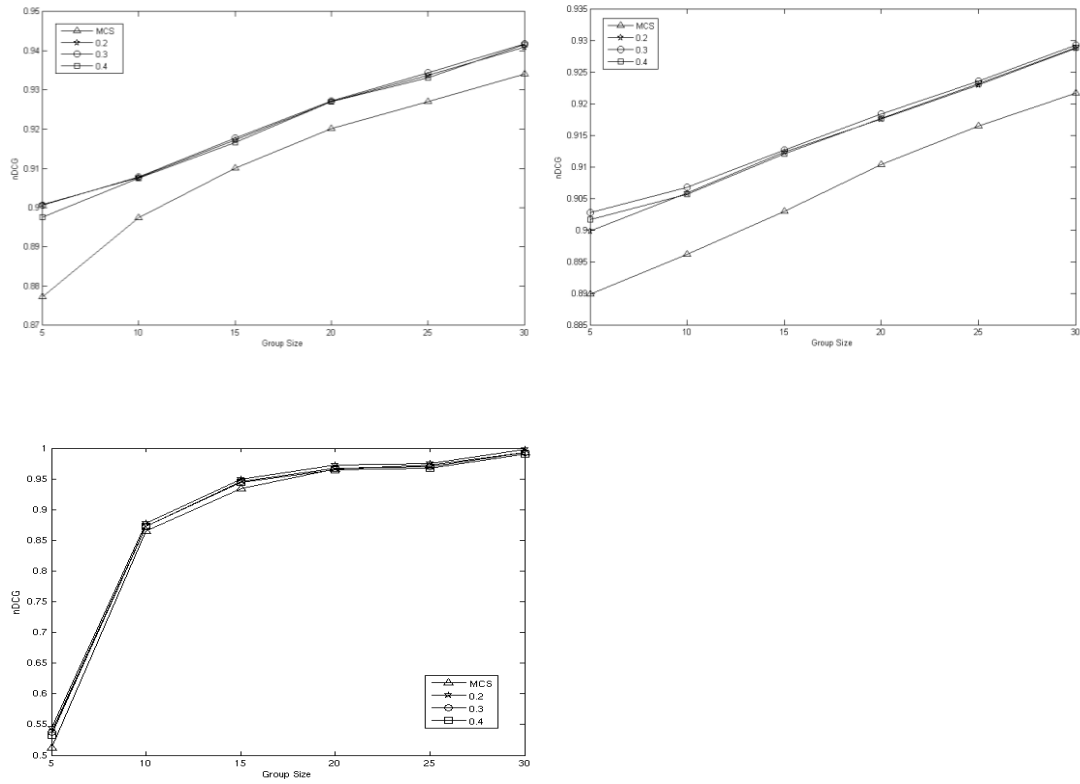

Figure 7 nDCG results when using the MCS model alone and when combining the MLA model using 0.2, 0.3 and 0.4 thresholds to produce local average ratings.

5. GroTo: a Group Recommender System for Web-based Tourist Group

Group tourism (GroTo) is a web-based group recommender system that aims to provide personalized recommendation activities for web-based tourist groups in Australia. In this system, the activities are classified and labeled in advance. There are six categories of tourism activity in GroTo: Nature, Sports, Arts, Aboriginal, Attractions and Social. Each category contains detailed activities for users to rate. For example, going to the beach, or visiting state parks and farms, can be rated by users in the Nature category.

The GroTo system has three components: a system interface, a recommender engine and a data server, as shown in Figure 8.

The system interface collects information from users who can actively specify their preferences for various tourist activities via web-based interfaces provided by the system. Users' context information can also be passively collected from mobile devices. Note that the preferences and historical visiting information are transformed into structural data, e.g. XML, in the user data collector module. Users' data are passed to the recommender engine for further processing.

The recommender engine parses structural the information of users, and user preferences are transformed into rating vectors in the user data server module. Additionally, every historical location that can be found and labeled in our system is transformed into ratings in the user data server module. Negative feedback is not transformed into ratings but will be used as criteria for pre-selection. The activities filter generates available activities by excluding all the activities that clearly do not appeal to members. The user contribution server models a group profile to describe overall group preferences using our MCS model. The group profile and negative list are given to the recommender server to filter appropriate activities to recommend to the group.

The data server is responsible for recording data from the system and individual information, preferences and feedback. It is important to point out that, except for individual

information and feedback, the group can be stored as a case for future recommendation. A group can be identified by its members' reason for getting together, such as holiday, conference, business or education. Recommendations can be precisely made to future groups by using a group filter that can find similar cases in the database.
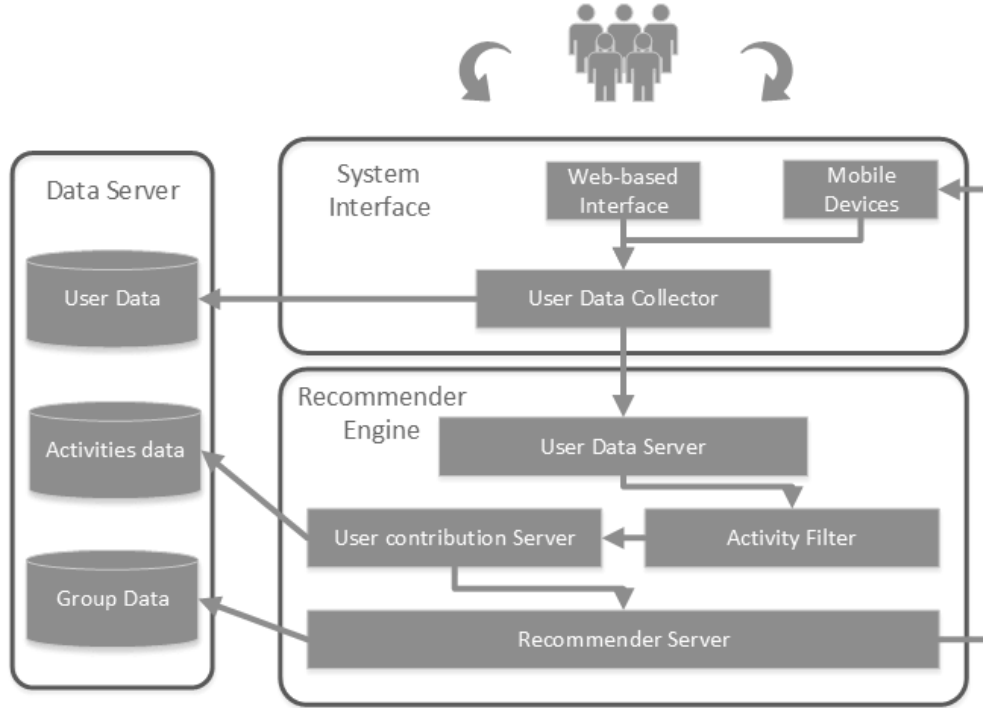

Figure 8. Architecture of our tourism recommender system GroTo.

We give an example in which only a selection of activities in GroTo is considered. A group is formed by six members who each nominate their preferences via the system interface. Their inputs are shown in Table 1. According to our proposal, their weightings are [0. 1216, 0. 1622, 0. 2162, 0.1622, 0.1622, 0. 1757]. According to Equation 13, we can obtain the group profile, $[r_{g,beach} = 3, r_{g,national/state\,parks} = 1.62, r_{g,farms} = 1.34, r_{g,diving} = 2.23, r_{g,snow\,sports} = 1.96, r_{g,cycling} = 1.71]$.

Table 1. Ratings of group members on the activities Nature and Sport: each row represents a member.

Nature					Sports					
beach	national / state parks	whale / dolphin watching	botanic gardens	farms	fishing	diving	surfing	snow sports	golf	cycling
5				3		4				2
	4			4						3
4						5		2		
4	1					3				5
	5			2				4		
5						1		5		1

The known ratings are shown in Table 2. We use UBC to predict the unknown group ratings for whale/dolphin watching, botanic gardens, fishing, surfing and golf, with the results 2.31, 1.73, 1.89, 2.56 and 1 respectively. If we recommend the three best activities to the group, they are surfing, whale/dolphin watching and fishing. Since the GroTo system's interface is under final development and testing, we only show the results in these tables. A detailed report on the GroTo system will be presented in another paper.

The results of this example show that our proposal can accurately aggregate individual preferences and produce appropriate recommendations for group.

Table 2. Observed ratings of non-member users for the activities Nature and Sport

UserID	beach	national / state parks	whale / dolphin watching	botanic gardens	farms	fishing	diving	surfing	snow sports	golf	cycling
1	5		4			2	5	4	2		
2		3		4							
3						5	4			5	
4											
5		4		4				3			4
6	2		1								
7	4	3				4				1	
8		5		4							
9							5				3
10	4		4			1					

6. Conclusion and Further Study

In this paper, we have proposed a new group recommendation approach for modeling group profiles by considering all member contributions to the group's activities. We have also proposed a MCS model to measure the contribution of each group member in which, by partitioning the item space, we can analyze members' opinions using the SNMF technique. In addition, the MLA model has been proposed to alleviate the fat tail problem by adaptively calculating the average rating related to the target item when predicting unknown group ratings. Using these two models, we can handle a high level of compromise in the group profile and exclude unnecessary information when generating predictions of user preferences.

The experiments were set up on two popular public datasets, and we have compared our approach with three popular approaches in the field of group recommendation. The results show the high effectiveness of our MCS-MLA approach.

This study not only has theoretical significance but also potentially has high practical application. Many online services, such as movie or tourism recommendation sites and other websites, could adopt our approach.

Our future study will include the extension of the proposed approach to select representative samplings instead of random samplings when sub-space differences are taken into consideration. A possible future improvement is to mathematically define a function to describe the degree of contribution divergence, and to incorporate alternative models when the function has a higher value.

References

[1]   J. Lu, Q. Shambour, Y. Xu, Q. Lin, and G. Zhang, "BizSeeker: a hybrid semantic recommendation system for personalized government-to-business e-services," *Internet Research,* vol. 20, pp. 342-365, 2010.

[2]   D. Wu, G. Zhang, and J. Lu, "A fuzzy preference tree-based recommender system for personalized business-to-business e-services," *IEEE Transactions on Fuzzy Systems,* 2014.

[3]   Q. Shambour and J. Lu, "A trust-semantic fusion-based recommendation approach for e-business applications," *Decision Support Systems,* vol. 54, pp. 768-780, 2012.

[4]   T. Sharon, H. Lieberman, and T. Selker, "A zero-input interface for leveraging group experience in web browsing," in *The 8th International Conference on Intelligent User Interfaces*, Miami, Florida, USA, 2003, pp. 290-292.

[5]   J. K. Kim, H. K. Kim, H. Y. Oh, and Y. U. Ryu, "A group recommendation system for online communities," *International Journal of Information Management,* vol. 30, pp. 212-219, 2010.

[6]   Z. Yu, X. Zhou, Y. Hao, and J. Gu, "TV program recommendation for multiple viewers based on user profile merging," *User Modeling and User-Adapted Interaction,* vol. 16, pp. 63-82, 2006.

[7]   J. A. Recio-Garcia, G. Jimenez-Diaz, A. A. Sanchez-Ruiz, and B. Diaz-Agudo, "Personality aware recommendations to groups," in *Proceedings of the 3rd ACM Conference on Recommender Systems*, New York, New York, USA, 2009, pp. 325-328.

[8]   J. F. McCarthy and T. D. Anagnost, "MusicFX: an arbiter of group preferences for computer supported collaborative workouts," in *The 1998 ACM Conference on Computer Supported Cooperative Work*, Seattle, Washington, USA, 1998, pp. 363-372.

[9]   L. Ardissono, A. Goy, G. Petrone, M. Segnan, and P. Torasso, "Intrigue: personalized recommendation of tourist attractions for desktop and hand held devices," *Applied Artificial Intelligence,* vol. 17, pp. 687-714, 2003.

[10]   A. Jameson, "More than the sum of its members: challenges for group recommender systems," in *The Working Conference on Advanced Visual Interfaces*, Gallipoli, Italy, 2004, pp. 48-54.

[11]   S. Basu Roy, S. Amer-Yahia, A. Chawla, G. Das, and C. Yu, "Space efficiency in group recommendation," *The VLDB Journal,* vol. 19, pp. 877-900, 2010/12/01 2010.

[12]   X. Wang, L. Sun, Z. Wang, and D. Meng, "Group Recommendation Using External Followee for Social TV," in *Proceedings of the 2012 IEEE International Conference on Multimedia and Expo (ICME)*, 2012, pp. 37-42.

[13]   W. Zhang, J. Wang, and W. Feng, "Combining latent factor model with location features for event-based group recommendation," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Chicago, Illinois, USA, 2013, pp. 910-918.

[14]   A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decision Support Systems,* vol. 43, pp. 618-644, 2007.

[15]   X. Liu, Y. Tian, M. Ye, and W.-C. Lee, "Exploring personal impact for group recommendation," in *The 21st ACM International Conference on Information and Knowledge Management*, Maui, Hawaii, USA, 2012, pp. 674-683.

[16]   P. Resnick and H. R. Varian, "Recommender systems," *Communications of the ACM,* vol. 40, pp. 56-58, 1997.

[17]   G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering,* vol. 17, pp. 734-749, 2005.

[18]   B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *The 10th International Conference on World Wide Web*, Hong Kong, China, 2001, pp. 285-295.

[19]   J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *The 4th Conference on Uncertainty in Artificial Intelligence*, 1998, pp. 43-52.

[20]   J. L. Herlocker, "Understanding and improving automated collaborative filtering systems," PhD thesis, University of Minnesota, 2000.

[21]   J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang, "Recommender system application developments: a survey," *Decision Support Systems,* vol. 74, pp. 12-32, 2015.

[22]   L. Baltrunas, T. Makcinskas, and F. Ricci, "Group recommendations with rank aggregation and collaborative filtering," in *Proceeding of the 4th ACM Conference on Recommender Systems*, Barcelona, Spain, 2010, pp. 119-126.

[23]   S. Berkovsky and J. Freyne, "Group-based recipe recommendations: analysis of data aggregation strategies," in *Proceedings of the 4th ACM Conference on Recommender Systems*, Barcelona, Spain, 2010, pp. 111-118.

[24]   J. Masthoff, "Group recommender systems: combining individual models," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds., Springer: New York, 2011, pp. 677-702.

[25]   J. Gorla, N. Lathia, S. Robertson, and J. Wang, "Probabilistic group recommendation via information matching," in *The 22nd International Conference on World Wide Web*, Rio de Janeiro, Brazil, 2013, pp. 495-504.

[26]   Y.-L. Chen and L.-C. Cheng, "An approach to group ranking decisions in a dynamic environment," *Decision Support Systems,* vol. 48, pp. 622-634, 2010.

[27]   M. O'Connor, D. Cosley, J. Konstan, and J. Riedl, "PolyLens: a recommender system for groups of users," in *The 7th European Conference on Computer Supported Cooperative Work*, Bonn, Germany, 2001, pp. 199-218.

[28]   L. Quijano-Sánchez, D. Bridge, B. Díaz-Agudo, and J. Recio-García, "A case-based solution to the cold-start problem in group recommenders," in *The 20th International Conference on Case-Based Reasoning*, Lyon, France, 2012, pp. 342-356.

[29]   M. Ye, X. Liu, and W.-C. Lee, "Exploring social influence for recommendation: a generative model approach," in *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Portland, Oregon, USA, 2012, pp. 671-680.

[30]   M. Gartrell, X. Xing, Q. Lv, A. Beach, R. Han, S. Mishra*, et al.*, "Enhancing group recommendation by incorporating social relationship interactions," presented at the 16th ACM International Conference on Supporting Group Work, Sanibel Island, Florida, USA, 2010.

[31]   L. Quijano-Sanchez, J. A. Recio-Garcia, B. Diaz-Agudo, and G. Jimenez-Diaz, "Social factors in group recommender systems," *ACM Transactions on Intelligent Systems and Technology (TIST),* vol. 4, pp. 1-30, 2013.

[32]   S. Amer-Yahia, S. B. Roy, A. Chawlat, G. Das, and C. Yu, "Group recommendation: semantics and efficiency," *Proceedings of the VLDB Endowment,* vol. 2, pp. 754-765, 2009.

[33]   D. Liu, X.-S. Hua, L. Yang, M. Wang, and H.-J. Zhang, "Tag ranking," in *Proceedings of the 18th International Conference on World Wide Web*, Madrid, Spain, 2009, pp. 351-360.

[34]   L. Ardissono, A. Goy, G. Petrone, and M. Segnan, "A multi-agent infrastructure for developing personalized web-based systems," *ACM Transactions on Internet Technology (TOIT),* vol. 5, pp. 47-69, 2005.

[35]   E. Vildjiounaite, V. Kyllönen, T. Hannula, and P. Alahuhta, "Unobtrusive dynamic modelling of TV programme preferences in a Finnish household," *Multimedia Systems,* vol. 15, pp. 143-157, 2009.

[36]   I. Garcia and L. Sebastia, "A negotiation framework for heterogeneous group recommendation," *Expert Systems with Applications,* vol. 41, pp. 1245-1261, 2014.

[37]   K. McCarthy, M. Salamó, L. Coyle, L. McGinty, B. Smyth, and P. Nixon, "CATS: a synchronous approach to collaborative group recommendation," in *The 19th International Florida Artificial Intelligence Research Society Conference (FLAIRS)*, Melbourne Beach, Florida, USA, 2006, pp. 86-91.

[38]   X. Hu, X. Meng, and L. Wang, "SVD-based group recommendation approaches: an experimental study of Moviepilot," in *Proceedings of the 2nd Challenge on Context-Aware Movie Recommendation*, Chicago, Illinois, USA, 2011, pp. 23-28.

[39]   Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer,* vol. 42, pp. 30-37, 2009.

[40]   J. D. M. Rennie and N. Srebro, "Fast maximum margin matrix factorization for collaborative prediction," in *Proceedings of the 22nd International Conference on Machine Learning*, Bonn, Germany, 2005, pp. 713-719.

[41]   T. Zhou, W. Bian, and D. Tao, "Divide-and-conquer anchoring for near-separable nonnegative matrix factorization and completion in high dimensions," in *Proceedings of the 13th IEEE International Conference on Data Mining*, Dallas, Texas, USA, 2013, pp. 917-926.

[42]   W. Wang, J. Lu, and G. Zhang, "A new similarity measure-based collaborative filtering approach for recommender systems," in *Foundations of Intelligent Systems*. vol. 277, Z. Wen and T. Li, Eds., Springer: Berlin Heidelberg, 2014, pp. 443-452.