Tailoring Recommendations to Groups of Users: A Graph Walk-based Approach

Heung-Nam Kim
University of Ottawa
800 King Edward, Ottawa, ON
K1N 6N5, Canada
hnkim@mcrlab.uottawa.ca

Majdi Rawashdeh
University of Ottawa
800 King Edward, Ottawa, ON
K1N 6N5, Canada
majdi@mcrlab.uottawa.ca

Abdulmotaleb El Saddik
University of Ottawa
800 King Edward, Ottawa, ON
K1N 6N5, Canada
elsaddik@uottawa.ca

ABSTRACT
With the rapid popularity of smart devices, users are easily and conveniently accessing rich multimedia content. Consequentially, the increasing need for recommender services, from both individual users and groups of users, has arisen. In this paper, we present a graph-based approach to a recommender system that can make recommendations most notably to groups of users. From rating information, we first model a signed graph that contains both positive and negative links between users and items. On this graph we examine two distinct random walks to separately quantify the degree to which a group of users would like or dislike items. We then employ a differential ranking approach for tailoring recommendations to the group. Our empirical evaluations on the MovieLens dataset demonstrate that the proposed group recommendation method performs better than existing alternatives. We also demonstrate the feasibility of Folkommender for smartphones.

Author Keywords
Social Recommender system; Group recommendation; Random walk with restarts

ACM Classification Keywords
H.5.3 **[Information interfaces and presentation]**: Group and Organization Interfaces

INTRODUCTION

With the current popularity of smart devices, such as smartphones and Smart TVs, users are experiencing a new convenient way of accessing rich multimedia content. For instance, Smart TVs, which offers more advanced capabilities in connectivity and experience through an Internet connection, enable users to view a limitless variety of content on their TV screen in addition to digital broadcast content. Consequentially, recommender techniques are becoming increasingly important, because users are also experiencing increasing difficulty with determining the most desirable content. While recommender systems have proliferated over the years, the large majority of the systems have been designed to recommend items (e.g., movies, videos, music, or TV programs) to individual users, depending on their personal interests [1]. It is, however, frequently the case that a group of people, such as friends, families, and colleagues, see multimedia content together [16]. Therefore, it would be a natural direction to develop recommender systems that make recommendations to groups of users; this is often referred to as group recommender systems [10]. More specifically, a recommender system may need to recommend different video and TV content to a viewer depending on whether they plan to view the content with their boyfriend/girlfriend or with their parents.

As a matter of fact, more and more research effort has been dedicated to group recommender systems, aiming at identifying items that satisfy individual preferences of all group members as much as possible. However, most existing studies have concentrated primarily on two aggregation approaches: (i) aggregating individual preferences to create a single group profile and then generating a recommendation list for this single group profile (referred to as the profile aggregation approach) [12, 14, 15, 21], and (ii) generating individual recommendation lists independently and then aggregating those lists to produce a single group recommendation list (referred to as the ranking aggregation approach) [2, 3, 8, 16]. Regardless of which approach is employed, both approaches have relied much on typical individual recommender techniques, such as collaborative and content-based filtering [1], in order to complete the entire group recommendation process. In addition, group recommendation paradigms that are based merely on aggregating either individual profiles or individual recommendations may often result in unsatisfactory recommendations to some of group members despite its simplicity. Therefore, the transition from individual recommendations to group recommendations still requires much more research effort, as the need for smart recommender services, from both individual users and groups of users, is expected to increase exponentially.

In this paper, we introduce Folkommender (Folk + Recommender)—a new recommender system based on a graph-based ranking method—that is designed to recommend multimedia content most notably to groups of

IUI'13, March 19–22, 2013, Santa Monica, CA, USA.

people, rather than to individual users; we henceforth use the term items to refer to multimedia content. Folkommender explores not only a positive aspect but also a negative aspect in a way that can achieve a general consensus among individual members about recommended items. From a graph viewpoint, the fundamental premises underlying our recommendation strategy are that (i) compelling items potentially suited to a certain group would be in close proximity to their past preferred items and (ii) inappropriate items for a certain group would be more closer to the items that its members have disliked in the past. To discover these two separate sets of items, we build two types of weighted user-item graphs derived from users' past preferences. We also incorporate social aspects of collaborative filtering in the graphs by assuming that (i) each member in a group is likely to prefer/hate an item that other similar users have previously liked/disliked and (ii) each member is likely to like/dislike an item similar to the items that they have previously liked/disliked. Using these graphs, we analyze two distinct random walks expected to visit particular items more frequently than other items: (i) a positive random walk that speculates about how much a group of users would like items and (ii) a negative random walk that speculates about how much a group of users would dislike items. Finally, we employ a differential approach, which considers both positive and negative rankings, for tailoring recommendations to a specific group.

RELATED WORK

We begin with the review of prior work on group recommender systems in this section. Since TV content is commonly enjoyed by groups of people, a growing number of studies have focused on the field of TV content/program recommendations for multiple viewers. In this domain, the profile aggregation approach has widely been used. Masthoff [14] studied a variety of aggregation strategies, including the Average model (that maximizes average satisfaction of individual members), the Least-Misery model (that maximizes satisfaction of misery), and the Most-Pleasure model (that takes the maximum of individual members), for building a group model derived from individual preferences. The author empirically assessed the effectiveness of different aggregation strategies for modeling a group of viewers watching TV together. In another study, Senot et al. [18] also investigated different group profiles obtained via various aggregation strategies for building accurate profiles according to groups' characteristics. Through experiments with a real dataset of TV viewings, the authors found that the Average model facilitated more accurate profiles for groups. Zhiwen et al. [21] proposed a TV program recommender system for multiple viewers, called TV4M. In the TV4M, each user profile was represented as a vector consisting of feature-weight pairs. For a group of users, the TV4M created a common group profile, also representable by a vector, by merging all members' profiles based on the total distance minimization. By using the merged profile, content-based filtering was exploited to recommend TV programs to the group of people. Sotelo et al. [19] presented a similar service, proposing an extension of AVATAR to establish a group recommender system in TV applications. This system adopted two different recommendation strategies depending on homogeneous groups and heterogeneous groups. For the homogeneous groups, it utilized the profile aggregation approach, whereas the ranking aggregation approach was employed for the heterogeneous groups.

Group recommender systems have also been studied in other application domains. McCarthy and Anagnost [15] developed a group preference arbitration system, called MusicFX, that determines music suited to a group of people working out together in a fitness center. O'Connor et al. [16] proposed PolyLens to recommend movies to small groups of users and discussed several design issues for group recommender systems. The PolyLens first predicted individuals' preference ratings for items by using user-based collaborative filtering and thus merged individual recommended lists based on the Least-Misery model. Berkovsky and Freyne [5] studied food recipe recommendations based on collaborative filtering. These authors evaluated the profile aggregation approach in comparison with the ranking aggregation approach, and identified that the former was superior to the latter in recipe recommendations. When aggregating individual preferences, this study considered several weighted models by assuming that some members of a group would have more influence than others. Amer-Yahia el al. [2] presented a consensus model aiming at maximizing group relevance, as well as minimizing group disagreement. For capturing disagreements between members of a group, the authors suggested two models, Average Pair-wise disagreements and Disagreement Variance, whereas the group relevance was computed by using the Average model or the Least-Misery model. This study demonstrated that incorporating disagreements between group members into a consensus function impacted positively on group recommendation. In another study, Gartrell et al. [8] proposed a new consensus function that considered three factors: a social factor, an expertise factor, and a dissimilarity factor. Based on the quantified social factor, different functions were flexibly used for aggregating individual recommendations. Baltrunas et al. [3] analyzed the effectiveness of group recommendations achieved by different rank aggregation techniques. Their experimental results demonstrated that the recommendation quality for homogenous groups did not tend to be affected by the group size. The authors also observed that the more similar users in a group, the more satisfied they are with recommendations. More recently, Kim et al. [12] introduced Commenders designed to recommend books for online communities. The Commenders exploited two filtering phases to improve both the recommendation effectiveness and the satisfaction of individual members. In the first phase, for a given group, the Commenders aggregated features of individual

members' profiles, and then generated candidate books that other similar groups frequently purchased. During the second phase, these candidate books were further filtered out by measuring how each candidate book was relevant to each individual member.

Although the above-mentioned studies may present promising possibilities for group recommendations, those studies show several main differences with our study. Compared with prior work that used the ranking aggregation method, our work does not require computing individual members' ranking scores separately and, thus, does not require merging them for items. Instead, the proposed approach can estimate unified ranking scores of group members on items at the same time, by analyzing link-structures that represent users' past rating behaviors. On the other hand, the principal difference compared to the profile aggregation approach is that our approach does not create a single merged profile for a group. Rather, our ranking method speculates as to how close items are to every individual member and their rated items. More importantly, in our study we distinguish a group's positive preferences from its negative preferences in ways that may facilitate a better grasp of group preferences and thus minimize every individual user's dissatisfaction with recommended items.

DEFINITIONS AND NOTATIONS

For better understanding, we introduce the basic notations used in this paper and formalize the group recommendation problem. Throughout this paper, we use boldface, upper-case letters, such as **R**, to denote matrices; the corresponding italicized, upper-case letters with two subscript indices represent entries in the matrices, such as R_{ui}. We always use boldface lower-case letters, such as **r**, to denote vectors; **r**(i) refers to the ith element (or entry) of the vector **r**. Moreover, we use the notations m and n to denote the total number of unique users and items in a system, respectively.

The users' preference indicator exploited in this study is either a *5-point Likert scale* or *a binary scale* (e.g., like/dislike, thumbs up/down). For a set of m users $U = \{u_1, u_2,...,u_m\}$ and a set of n items $I = \{i_1, i_2,..., i_n\}$, we represent explicit past opinions (ratings) of whether or not the users have preferred the items by an $m \times n$ user-item matrix **R**; an entry R_{ui} is r such that $r \in \{1,\ldots, 5\}$ or $r \in \{1, -1\}$ if the uth user has rated the ith item, and is unknown otherwise. We denote by $\varnothing$ the unknown opinion. The group recommendation problem of our study is formally defined as follows: Given a user-item matrix **R** and a group of users $X \subseteq U$, the group recommendation is to identify an ordered set I_X of items that will be of interest to every individual member of group X such that $| I_X | \leq K$, $\forall u \in X$, $i \in I_X$: $R_{ui} = \varnothing$.

A GRAPH WALK-BASED APPROACH TO GROUP RECOMMENDATIONS

In this section, we present a graph-based approach to ranking and recommending items that are tailored to a specific group. In our work, each user and item is represented as a node in a graph. From users' ratings, we derive two distinct link structures between users and items: a positive link structure and a negative link structure. To take advantage of collaborative social wisdom, we also incorporate neighborhood-based links into each link structure. On such link structures (graphs), we perform two separate random walks that periodically restart from specific nodes associated with a given group of users. On the basis of the random walks, we finally identify a set of ranked items that the group would like the most. In the remainder of this section, we elaborate how to model graphs from users' ratings, how to analyze two distinct random walks in regard to a given group, and how to rank (recommend) items for the group.

Building Relationship Graphs

Although the rating scale is fixed in the Likert-scale model, different users would have different rating behaviors for items. In our study, if a user's rating for a certain item is greater than that user's average rating $\breve{R}_u$, we call this rating a positive opinion for that item, and a negative opinion otherwise. In this context, we further split the user-item matrix **R** into two matrices: a positive matrix and a negative matrix:

- User-item positive matrix $\mathbf{R}^+$ whose entry represents the degree to which the corresponding row user has *liked* the corresponding column item, defined as follows:

$$R_{ui}^{+} = \begin{cases} R_{ui} - \breve{R}_u, & \text{if } R_{ui} > \breve{R}_u \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

- User-item negative matrix $\mathbf{R}^-$ whose entry represents the degree to which the corresponding row user has *disliked* the corresponding column item, defined as follows:

$$R_{ui}^{-} = \begin{cases} \breve{R}_u - R_{ui}, & \text{if } R_{ui} < \breve{R}_u \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

Note that we remove the rows consisting entirely of zeros from $\mathbf{R}^+$ and $\mathbf{R}^-$, implying that the corresponding users have no positive (or negative) opinions for any items. Analogously, we discard the columns of $\mathbf{R}^+$ and $\mathbf{R}^-$ consisting entirely of zeros, implying that the corresponding items have previously not received any positive (or negative) opinions. Accordingly, the dimensions of the rows and columns of both matrices are equal to or less than m and n, respectively.

Alternatively, in a situation where the binary rating scale (i.e., like/dislike) is employed for the preference indicator, $\mathbf{R}^+$ and $\mathbf{R}^-$ can be easily defined as follows:

- User-item positive matrix $\mathbf{R}^+$, defined as follows:

$$R^{+}_{ui} = \begin{cases} 1, & \text{if } R_{ui} = 1 \\ 0, & \text{otherwise.} \end{cases} \quad (3)$$

- User-item negative matrix $\mathbf{R}^{-}$, defined as follows:

$$R^{-}_{ui} = \begin{cases} 1, & \text{if } R_{ui} = -1 \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

Using $\mathbf{R}^{+}$ and $\mathbf{R}^{-}$, we build a bipartite signed, undirected graph $G_{BI} = (U \cup I, E^{+} \cup E^{-})$, whose nodes can be partitioned into two disjoint sets U and I such that each edge in E^{+} and E^{-} connects a node in U and a node in I. Weights on edges correspond to nonzero entries of $\mathbf{R}^{+}$ and $\mathbf{R}^{-}$.

Since, by definition, no two nodes within the same set are adjacent in the bipartite graph G_{BI}, there are no explicit links between pairs of two users or pairs of two items. In the following subsections, we additionally define two neighborhood graphs in which edges connects pairs of two nodes within the same set.

User-user neighborhood graph

In our study, for a given user, the user neighborhood is defined as the set of users who have exhibited preferred items similar to those of that user; this set is often called *k nearest neighbors* [6]. To identify such a neighborhood set, we employ the cosine similarity measure. For computing the similarity between two users, each user is treated as a *row* vector in the n dimensional item-space of the user-item matrix $\mathbf{R}$. Formally, the similarity between two users u and v is computed as

$$W_{uv} = \frac{\sum_{a \in I} R_{ua} \times R_{va}}{\sqrt{\sum_{a \in I} R^{2}_{ua}} \sqrt{\sum_{a \in I} R^{2}_{va}}} \quad (5)$$

Finally, for m users, the similarity of users can be represented as an $m \times m$ user-user similarity matrix $\mathbf{W}$. Since we consider the k nearest neighbors of each user, we set W_{uv} to the similarity value between a pair of two users u and v if the corresponding similarity value is greater than the k highest similarity value in the vth column of $\mathbf{W}$, and 0 otherwise. In other words, each column in the matrix $\mathbf{W}$ contains at most k nonzero values indicating the similarity values between the corresponding column user and the k most similar users. Note that all entries on the main diagonal in the matrix equal 0. We denote this resultant matrix as $\mathbf{W}_k$ where k is the number of similar users. By utilizing the matrix $\mathbf{W}_k^{\mathrm{T}}$ as an adjacency matrix, we define a user neighborhood graph $G_U = (U, E_U)$ where E_U contains a directed edge $e(u,v)$ iff user v is user u's neighborhood such that $W_{vu} > 0$.

Item-item neighborhood graph

For a given item, the item neighborhood is defined as the set of items to which users have assigned ratings similar to what they do for that item; this set is often called *k most similar items* [7]. In the case of item-item similarities, each item is regarded as a *column* vector in the m dimensional user-space of $\mathbf{R}$. Therefore, for two vectors corresponding to the ith and jth columns of $\mathbf{R}$, the similarity between two items i and j is given by

$$V_{ij} = \frac{\sum_{a \in U} R_{ai} \times R_{aj}}{\sqrt{\sum_{a \in U} R^{2}_{ai}} \sqrt{\sum_{a \in U} R^{2}_{aj}}} \quad (6)$$

For n items, the similarity between every pair of two items can be represented as an $n \times n$ item-item similarity matrix $\mathbf{V}$. We also discard any unnecessary values for each item in a manner similar to the user-user matrix $\mathbf{W}_k$. We denote the resultant matrix by $\mathbf{V}_k$, where each column stores k most similar items to the column's corresponding item; in the matrix $\mathbf{V}_k$, V_{ij} is set to the similarity value between two items i and j, if the corresponding similarity value is greater than the k highest similarity value in the jth column of $\mathbf{V}_k$ and 0 otherwise. Additionally, all entries on the main diagonal in the matrix also equal to 0. In an item viewpoint, a directed neighborhood graph is defined as $G_I = (I, E_I)$ where E_I contains a directed edge $e(i, j)$ iff item j is k most similar items to item i such that $V_{ji} > 0$.

Analyzing Random Graph Walks

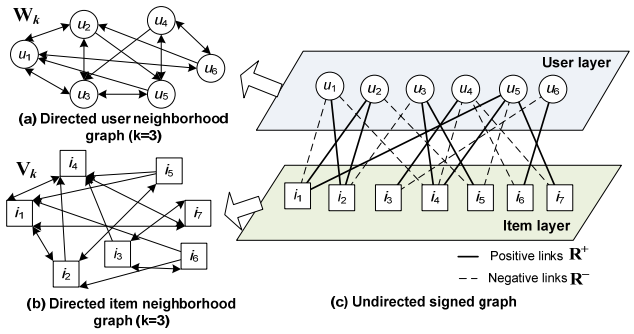

Figure 1. Two-layer signed graph model.

Figure 1 illustrates a two-layer signed graph. The basic intuition behind our group recommendation approach from a random walk viewpoint is that an item that is likely to be suitable for a particular group of users would tend to be highly reachable not only from the users of that group, but also from items that they have previously liked via paths of (strong) positive links. On the other hand, if a certain item tends to be highly visited by a random walk via paths of the group's negative links, the group of users would dislike such an item. With this aspect in mind, on our signed graph we perform two distinct random walks expected to visit particular items more frequently than other items: a *positive random walk* and a *negative random walk*. The former can only cross from one layer to another layer through positive links, whereas the latter can do so through negative links. Additionally, both walks can move to another node within the same layer via available (user or item) neighborhood links of a current node.

For the two random walks, we distinguish between positive and negative links, and separately construct two graphs: one

graph only contains positive links between two different layers and another contains negative ones between them. The adjacency matrices $\mathbf{P}'$ and $\mathbf{N}'$ respectively corresponding to these graphs are represented by

$$\mathbf{P}' = \begin{pmatrix} \mathbf{W}_k & \mathbf{R}^+ \\ (\mathbf{R}^+)^{\mathrm{T}} & \mathbf{V}_k \end{pmatrix} \quad \mathbf{N}' = \begin{pmatrix} \mathbf{W}_k & \mathbf{R}^- \\ (\mathbf{R}^-)^{\mathrm{T}} & \mathbf{V}_k \end{pmatrix} \tag{7}$$

in which each entry's value (weigh) can be viewed as the link strength between a pair of nodes. We then normalize matrices $\mathbf{P}'$ and $\mathbf{N}'$ in order to obtain column-stochastic matrices $\mathbf{P}$ and $\mathbf{N}$ which are transition matrices of two Markov chains corresponding to two distinct random walks, respectively. Since transition probabilities are proportional to link-strengths, a random walker is more likely to traverse links of high strength.

Our study also incorporates the restart probability $d \in (0, 1)$ as in (Personalized) PageRank. That is, with probability d, a random walker jumps back (teleports) to a random node among a set of specific nodes (e.g., favorite items) and thus restarts from those nodes at each step [10]; a generally-accepted value for d is 0.15. For a given group, we compute two stationary distribution vectors that result from the positive and negative random walks using our transition matrices. Formally, the stationary distribution vectors of those random walks with restarts are defined as the solutions of the following equations:

$$\mathbf{r}^+ = (1-d)\mathbf{P}\mathbf{r}^+ + d\mathbf{v}^+ \tag{8}$$

$$\mathbf{r}^- = (1-d)\mathbf{N}\mathbf{r}^- + d\mathbf{v}^- \tag{9}$$

such that $\| \mathbf{r}^+ \|_1 = 1$ and $\| \mathbf{r}^- \|_1 = 1$. $\mathbf{v}^+$ and $\mathbf{v}^-$ are probability vectors that allow non-uniform probabilities for teleporting to particular nodes.

In our study, different teleportation vectors $\mathbf{v}^+$ and $\mathbf{v}^-$ are built depending on a given group. Let $X = \{u_1, u_2, ..., u_{|X|}\}$ denote a group of users for recommendations. For group X, we denote by $I^+(X)$ and $I^-(X)$ the set of items *positively* and *negatively* rated by at least one of group X's members, respectively. Then, the non-normalized teleportation vectors, with regard to the ath entry, are respectively defined as:

$$\overline{\mathbf{v}}^+(a) = \begin{cases} 1 & \text{if } a \in X, \\ U^+(a) & \text{if } a \in I^+(X), \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

$$\overline{\mathbf{v}}^-(a) = \begin{cases} 1 & \text{if } a \in X, \\ U^-(a) & \text{if } a \in I^-(X), \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

where $U^+(a)$ is the number of group X's members who have *positively* rated item a in common and $U^-(a)$ is that of common members who have *negatively* rated item a; we normalize the teleportation vectors so that all entries sum to 1, i.e., $\| \mathbf{v}^+ \|_1 = 1$ and $\| \mathbf{v}^- \|_1 = 1$. The salient concept behind the positive teleportation vector, which corresponds to attaching different importance to nodes, is that at every step, with some probability, a random walker teleports to members of the current group or items that the members have previously liked. In the case of a negative teleportation vector, a random walker teleports to items that group members have disliked in the past. Since $\mathbf{v}^+$ and $\mathbf{v}^-$ are non-uniform distribution vectors, a probability of a node to which a random walker teleports could differ from that of another node. The more the users liked an item, the higher the probability of teleporting to the item. Alternatively, we may be able to employ values in user-item positive and negative matrices $\mathbf{R}^+$ and $\mathbf{R}^-$ for setting items' entries of $\mathbf{v}^+$ and $\mathbf{v}^-$, respectively. We however give group members the same weights (i.e., ones) so that the random walks are not biased towards a specific member.

The resultant distributions contained in vector $\mathbf{r}^+$ correspond to the long-term probabilities of the positive random walk expected to visit specific nodes when the random walker periodically restarts from one of the nodes in X and $I^+(X)$, i.e., non-zero entries in $\mathbf{v}^+$, where the particular restart node is chosen non-uniformly at random. Accordingly, this vector $\mathbf{r}^+$ can be used for ranking nodes (particularly items) in regard to a given group X from a positive perspective, reflecting the degree to which items are in close proximity to the group. We regard items having high reachable probabilities as desirable items for group X. We name this resultant vector $\mathbf{r}^+$ a *positive ranking vector*. As opposed to the positive random walk, the negative random walk traverses negative links between users and items and thus is much more likely to visit items that a group of users would dislike. Therefore, the higher the probability an item has, the less its importance (ranking) for the group. We refer to this resultant vector $\mathbf{r}^-$ as a *negative ranking vector*.

Group Recommendations

Since individual members of a certain group may have different preferences of the same items, an effective method for capturing a consensus among members about items is critically impo

rtant to group recommender systems. When generating a recommendation list tailored to a group of people, we should take some questions into consideration [13]: Is it better to recommend items with which one person is as satisfied as possible? Or to recommend items with which no one is too dissatisfied? To minimize each individual member's dissatisfaction with recommended items, we employ a *differential* ranking approach that subtracts negative ranking scores from positive ranking scores. In other words, we consider not only how group users are likely to like an item, but also how they are likely to dislike that item when deciding on a final recommendation list. In this regard, we further compute the final ranking vector $\mathbf{r}$ as follows:

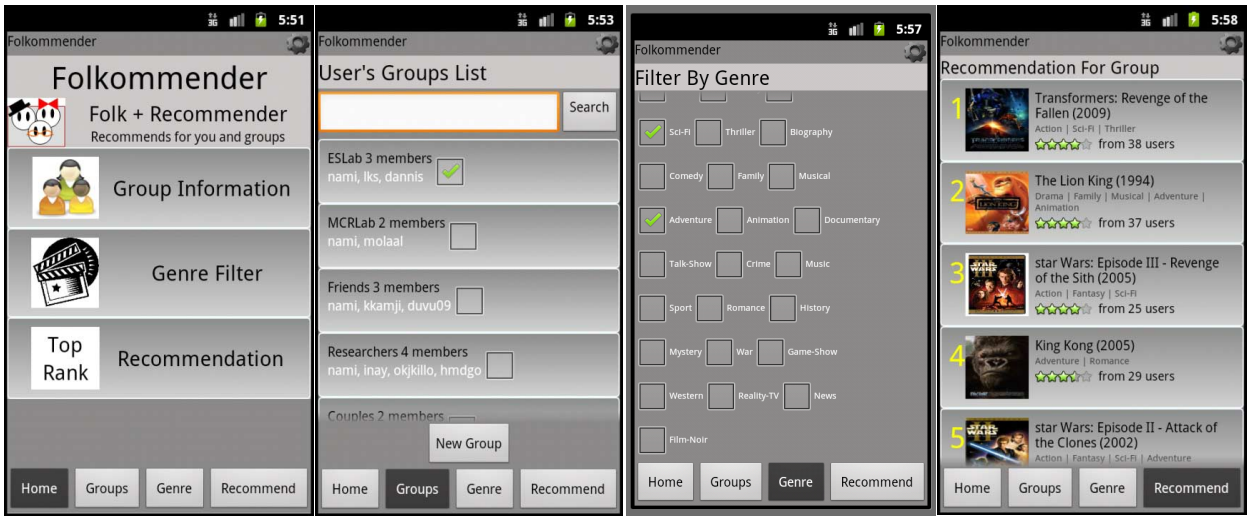

Figure 2. The Folkommender App screenshots, showing a main page, a group selection, genre filtering, and a top-ranked recommendation list.

$$\mathbf{r} = \alpha \mathbf{r}^{+} - (1-\alpha)\mathbf{r}^{-} \qquad (12)$$

where $\alpha \in (0, 1)$ is a non-negative constant for adjusting the relative weighting between the positive ranking and the negative ranking. Note that for any given group X, the resultant ranking vector $\mathbf{r}$ contains a group view of preferences not only for users, but also for items. Since we aim to recommend items for a group of users, we ignore entries related to users. We call a value of an entry $i \in I$ in $\mathbf{r}$, $\mathbf{r}(i)$, a *group ranking score* of item i. The higher the ranking score of a particular item, the more desirable that item is for recommendations. Once $\mathbf{r}$ is estimated with respect to group X, we identify a set of K ordered items having the highest ranking scores and subsequently recommend those items to the group. It should be emphasized that our approach can be also employed for individual recommendations by assuming that a certain group consists only of a single user, i.e., $|X| = 1$. In this context, $\mathbf{r}$ will contain ranking scores solely for that user, resulting in "personalized recommendations" instead of group recommendations.

SMARTPHONE APPLICATION

Using our graph-based recommendation model, we have implemented a web-based Folkommender prototype that supports movie recommendations not only to groups, but also to individuals. Folkommender also has been designed to be served with smartphones. Figure 2 shows snapshots of a Folkommender application (App) working on an Android smartphone. The Folkommender App mainly consists of three steps to generate recommendations. The first step is to form a group to which Folkommender makes recommendations. Folkommender allows users to form groups in two ways: selecting their existing group list or creating a new group comprised of people on Folkommender. Individual users are also able to create themselves as a group in order to receive recommendations based exclusively on their personal interests. As one of the optional functions, users can choose genres to filter out movies according to the desired genres. If the genres are not selected, all genres are considered. In the final step, Folkommender displays the recommended movies according to their ranking scores for the selected group.

Beyond this smartphone App, Folkommender can be extended to a smart TV App—a web-based software program that works on a digital TV connected to the Internet—so that it can be directly accessible through TV screens. Since our App is currently running on an Android platform, it can be easily extensible to Android-based smart TVs, such as Google TV. As part of our future work, we plan to expand the Folkommender App to be optimized for smart TVs.

DATASET AND EVALUATION METHODOLOGY

Evaluating group recommender systems is generally harder than evaluating personalized recommender systems due to the absence of publicly available datasets containing both individual preferences and group preferences for items (e.g., numerical ratings, like/dislike). For the offline experiment, we used the MovieLens 100k dataset, consisting of 100,000 ratings (a 1-to-5 rating scale) from 943 users on 1682 movies. Since this dataset does not have group information, we artificially formed a number of groups for evaluation purposes.

Group Formation

When generating groups, we took account of two factors, group *size* and group *cohesiveness*, as per the suggestion in [2, 3]. We first selected items rated by at least 50 users. Subsequently, for each of those items, we identified users who have given the item a perfect rating of 5 in common and generated three different types of groups—that is, a *homogeneous* (similar) group, a *heterogeneous* (dissimilar) group, and a *random* group—from those users, with a

predefined group size. Given an item *i*, for example, if six users, u_1, … u_6, have rated it in common with 5 points, we generated three groups, e.g., $X_1 = \{u_1, u_2\}$, $X_2 = \{u_3, u_4\}$, and $X_3 = \{u_5, u_6\}$, depending on the constraints of similarities among inner members of the groups. To define the degree of how similar/dissimilar group members are, we examined the distribution of cosine similarities between every pair of users in the MovieLens dataset. As a result, the average similarity was 0.178, with a standard deviation of 0.111. Accordingly, we formed a homogeneous group that was comprised of users who have similarities (between them) higher than 0.3, whereas a heterogeneous group consisted of users having similarities less than 0.17. For a random group, we randomly generated its members without any restriction on their similarities. Moreover, when forming groups, we considered four group sizes: 2, 3, 4, and 8. Table 1 displays the overall characteristics of the generated groups for our experiments.

Group type:	Similar	Dissimilar	Random
# groups	588	489	588
# unique users	578	565	644
Average per group			
Size	4.02	2.96	4.07
Similarity	0.423	0.143	0.305
Shared items	97.9	14.4	58.3

Table 1. Group characteristics used in our experiments.

Evaluation Protocol

To evaluate the performance of group recommendations, for each formed group, we withheld members' ratings on an item that all members have rated with 5 points and used this item as the test data for the group (i.e., one relevant item per group). The remaining rating data was employed as the training set for making group recommendations. Since every member in a group has greatly liked a test item in common, this item should appear earlier than other items (that every member has not yet rated) in a ranked list made by a group recommendation algorithm. Therefore, the goal of this test is to identify the relative place of the hidden relevant item within the total order of available items sorted by their ranking scores. To measure the recommendation quality with this dataset, we employed the Hit Rate (HR) at top-*K* [7], which judges how often a list of recommendations for a given group contains a test item that all members of the group have actually liked. Since recommender algorithms focus mainly on recommending high-quality items with higher ranks, we set the number of recommended items to 10 (i.e., top-10 recommendation).

Comparison algorithms

For comparison purposes, we experimented with the *Average* ranking aggregation model, which maximizes average satisfaction of individual members. Given a group *X*, the Average model computes a group score of a given item *i* as follows [2]:

$$\mathrm{avg}_X(i) = \frac{1}{|X|}\sum_{u \in X} \mathrm{score}(u,i) \quad (13)$$

in which score(u, i) is a predicted preference score of user u for item i. For predicting individual scores, we exploited two straightforward Collaborative Filtering (CF) algorithms, a user-based CF [6] and an item-based CF [17], which have been widely adopted in group recommender algorithms [2, 3, 16, 19]. In addition to CF approaches, we also tested a variant of PageRank for a recommender system, called *ItemRank* [9]. In the experiment section, we denoted by AUCF, AICF, and AIRank the Average model with UCF, ICF, and ItemRank, respectively.

EXPERIMENTS AND RESULTS

This section compares the performance of our recommender method against that of the baseline methods. As noted in a number of previous studies, the size of the neighborhood k could influence the recommendation quality of neighborhood-based algorithms. Therefore, we carried out experiments by varying k from 10 to 30 with an increment of 10, and chose the value of k that yielded a higher rank for each group on each method. One point that is worth noting is that, in the MovieLens evaluation dataset, any further increases, beyond k = 30, either barely improved the performance or rather affected the degradation of the performance with regard to HR. For the differential ranking approach of Folkommender, defined in Equation (12), we also varied parameter α from 0.5 to 0.9 for adjusting the relative weighting between the positive and negative ranking vectors. As a result, we observed that higher values of α ($0.7 \le \alpha \le 0.9$) tended to lead to higher recommendation ranks. As a consequence, we reported the results obtained using two versions of Folkommender, one as the non-differential approach of Folkommender in which α was set to 1 (denoted as PosFolkR) and the other as the differential approach of Folkommender in which α was set to 0.8 (denoted as DifferFolkR).

AUCF	AICF	AIRank	PosFolkR	DifferFolkR
16.2%	15.9%	20.4%	22.1%	23.5%

Table 2. Hit-rate within the top-10 recommendations for all test groups.

We first measured HR values obtained using the previously mentioned five methods for all test groups (1665 groups in total) without considering their cohesiveness and size. Since users practically consider only the top part of a recommended list, this measure is useful to glance briefly the algorithms which on average tend to put hidden items in the very top positions. A summary of the principal results is shown in Table 2. As our study focuses on group recommendations, we intuitively expected that two main factors—group size and group cohesiveness—would affect the overall performance of each method. We therefore pay

full attention to the influence of these two factors in more detail throughout the subsequent sections.

Evaluation of group cohesiveness

This section investigates the impact of group cohesiveness on the recommendation performance. In service of this objective, we calculated HR values depending respectively on homogeneous, heterogeneous, and random groups without consideration of group size.

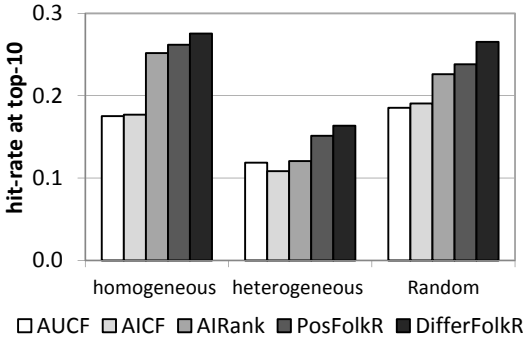

Figure 3. Hit-rate at top-10 recommendations with respect to different types of groups.

Figure 3 depicts the results for the three types of groups. As shown in the graph, it is clear that our Folkommender approaches, PosFolkR and DifferFolkR, performed significantly better than the baseline approaches on all occasions. For instance, for the homogeneous groups, the differential Folkommender strategy improved HR by 10% over AUCF, by 9.9% over AICF, and by 2.4% over AIRank, respectively. The results of the random groups did not qualitatively differ from those of the homogenous groups. Shifting attention to the results of the heterogeneous groups, we can also clearly see the superiority of the Folkommender methods. We also found that the recommendation performance trended downward as the overall similarity of group members decreased; that is, a significant HR disparity was clearly apparent between the homogeneous (or random) groups and the heterogeneous groups. This might be partially because the heterogeneous groups on our evaluation dataset were composed of a relatively small number of users compared with the homogeneous and random groups. This would be true in a real-world situation, as largely heterogeneous groups would be formed only rarely in practice. This aspect will be explored in more detail in the next section.

As mentioned earlier, a homogeneous group is comprised of users who have "similar" tastes, whereas a heterogeneous group is comprised of "dissimilar" users. To gain deeper insight into the impact of this cohesiveness on group recommendations, we further selected two different classes of groups according to the constraint of group members' similarity: (i) the average similarity of members is less than or equal to 0.1 (81 groups in total) and (ii) that of members is greater than or equal to 0.5 (141 groups in total). We then examined the HR values only for those groups. As shown in Table 3, the Folkommender approaches continued to surpass the baselines for both highly inner similar and dissimilar groups. A more interesting situation that we focus on is the highly heterogeneous groups; in this situation, DifferFolkR outputted the best results, implying that combining positive ranking scores with negative ones did help to improve the recommendation rankings. PosFolkR, however, achieved comparable results for the highly homogeneous groups.

To sum up, the performance of group recommendations depended somewhat on a state of cohesion inherent in groups. The more similar users in a group, the more accurately the algorithms identified suitable items for the group. Compared to the typical average methods, the differential Folkommender ranking strategy performs well in general. Incorporating negative ranking scores in creating a final ranking list proves to be effective for random or heterogeneous groups, but is not quite useful for the highly inner similar groups compared with the non-differential strategy.

	Group members' similarity	
	≤ 0.1	**≥ 0.5**
AUCF	13.4 %	14.2 %
AICF	4.9 %	21.3 %
AIRank	14.6 %	43.3 %
PosFolkR	17.1 %	47.5 %
DifferFolkR	22.0 %	48.9 %

Table 3. Hit-rate at top-10 with respect to highly heterogeneous groups and highly homogeneous groups.

Evaluation of group size

In this experiment, we investigated how sensitive each of the algorithms was regarding the number of users contained in groups. In addition to group cohesiveness, group size would constitute another significant factor that could affect the recommendation performance. To explore this impact, we measured HR values associated with different group size within homogeneous, heterogeneous, and random groups, respectively. Figure 4 depicts the results, showing how each method was affected by the size of groups. The number of groups per size is also plotted in the top of each bar chart. As shown, our Folkommender offers more benefits to small groups rather than to large groups. The results demonstrated that the Folkommender approaches were considerably superior to the CF approaches in the cases when group sizes were relatively small (i.e., group size ≤ 4). When the group size was 8 (i.e., large groups), AICF performed best particularly for homogeneous and random groups, but our differential approach still performed best for heterogeneous groups. Interestingly, AICF yielded uneven performance and seemed highly sensitive to the group size; in the large group case, AICF outputted the best results for homogeneous and random

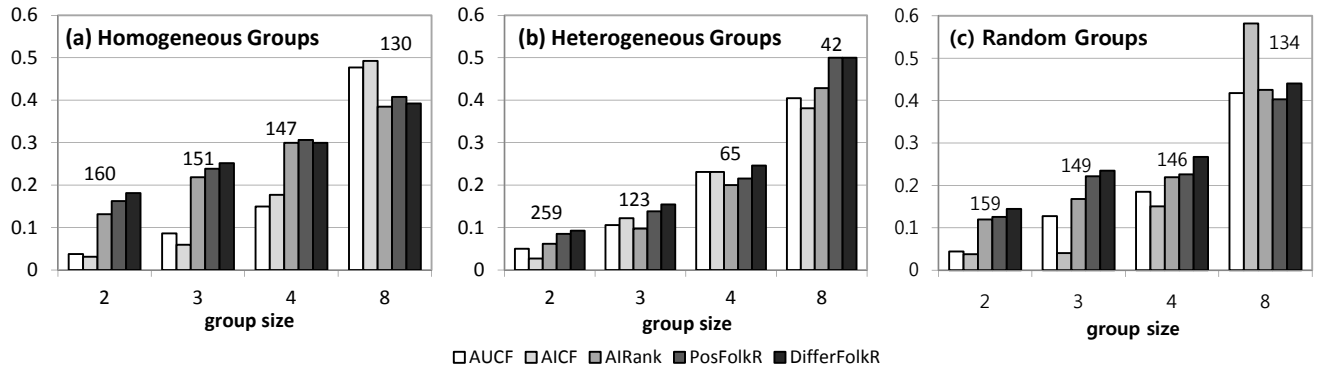

Figure 4. Comparisons of the hit-rate at top-10 according to different group size.

groups, but in the small group cases it outputted the worst results.

Our results also showed that the performance of all the methods in terms of HR appeared profoundly group size-dependent, regardless of whether or not group members were similar to one another. Surprisingly, it was determined that the HR values did tend to improve as the group size increased. For example, in the case of the homogenous groups consisting of two members, AUCF, AICF, AIRank, PosFolkR, and DifferFolkR demonstrated HR values of 0.038, 0.031, 0.131, 0.163, and 0.181, respectively; those values remarkably increased up to 0.477, 0.495, 0.384, 0.408 and 0.392, respectively, when the group size was 8. A similar upward trend was observed in the heterogeneous and random groups as well. These results might be caused by the fact that we judged how well each algorithm positioned a test item—that every member has previously rated with 5 points—at a higher rank for a given group. In this scenario, we found that numerous members contributed more to hitting the test item within their recommended list.

DISCUSSIONS AND OPEN ISSUES

There are several open issues and interesting tasks which could be performed so as to extend our work.

Since our graph-based recommendation method does not require any additional information about users or items aside from users' rating information, it could be potentially applicable to various recommendation contexts—for example, TV programs, social media, movies, music, and travel packages. Nevertheless, similar to most collaborative filtering-based methods, our method could suffer from rating sparsity-related problems. One problem that Folkommender could encounter is "cold start" cases wherein little or no information is known about users or items [1]. For example, an item newly added to the system cannot be recommended until sufficient users give the item their ratings. Analogously, cold start items that only a few users have rated would be positioned very rarely in the top part of a recommended list. Furthermore, when calculating group ranking scores, our method could be biased somewhat toward active members having a large number of ratings compared to other members. Consequently, the recommendation result could neglect opinions of cold start members who have previously rated very few items. These limitations raise some future theoretical and empirical research for more advanced recommender methods.

As shown by our experimental results, our method seemed not to be effective for large, homogeneous groups, particularly compared with AICF. In this work, we assumed that members in the same group have equal weight as we aimed to identify items that could satisfy all members as much as possible. In some cases, however, certain members would have more influence than others and thus be good leaders in their group recommendations. Since members in a homogeneous group have similar tastes in viewing multimedia content, influential members, particularly within "large" groups, may lead to good recommendations for their members. In teleportation vectors $\mathbf{v}^+$ and $\mathbf{v}^-$, we assigned the same values to entries of all members belonging to a specific group. However, it is possible to assign different values depending on the influence/importance particular users have in a group. We intend to investigate this impact on our recommendation performance in the future.

In our experiments, we tested the recommendation performance in a situation where groups are somewhat stable. However, we also need to concentrate on the dynamic nature of groups, which can turn into a flux over time. It is often the case that groups are temporally formed by people who have a common aim at a particular moment. Simply, we can compute the positive and negative ranking vectors, $\mathbf{r}^+$ and $\mathbf{r}^-$, using the power iteration method. In practice, however, it would not be possible to compute these ranking vectors naively with fixed iterations online in response to a temporal group's request, as those tasks are time-consuming processes. Thus, as with most recommender systems, we need pre-computation steps which can be accomplished offline. Fortunately, remarkable efforts have been made to facilitate the computation of PageRank-like algorithms [4, 20]. In particular, Jeh and Widom [11] proved a linear relationship between

teleportation vectors and their corresponding personalized PageRank vectors. This linear relationship allows (personalized) PageRank scores to be obtained using a linear combination of *basis vectors* that are practically computed offline. Inspired by this study, at recommendation time, our positive and negative ranking vectors can be obtained from pre-computed basis vectors associated with a specific group; this computation approach could support fast recommendations online.

Our experiments showed that our group recommendation approach is fruitful in ranking hidden items. In the MovieLens dataset, however, the number of relevant items to a group in the test set may be a small fraction of the number of entire relevant items that are actually of interest to that group. This is because it is impossible that we collect the entire ground truth data about which items are of interest to each group. Accordingly, in our test scenario, we were not able to evaluate the appropriateness of many of the recommended items for a group as there is no preference information of that group for those items in the dataset. Accordingly, more in-depth studies with users are required to further validate the effectiveness of our method.

CONCLUSIONS

An effective service providing group recommendation is of undisputed value to users. In this paper, we developed a recommender system, which we call Folkommender, for recommending items to groups of users. Specifically, we model a two-layer signed graph that reflects both users' positive and negative opinions on items. We then exploit this graph for developing a graph-based differential ranking approach on group recommendation. This ranking approach examines random walks with restarts on two different link-structures derived from users' ratings in order to consider group ranking scores not only from a positive perspective but also from a negative perspective. Our experiments on the MovieLens dataset show that Folkommender performs better than existing alternatives most notably for small groups regardless of whether or not members are similar to each other. Additionally, our differential ranking strategy can be successfully leveraged to improve recommendation performance especially for heterogeneous/random groups.

REFERENCES

1. Adomavicius, G., Tuzhilin, A. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. and Data Eng. 17*, 6 (2005), 734–749.
2. Amer-Yahia, S., Roy, S.B., Chawlat, A., Das, G., Yu, C. Group recommendation: semantics and efficiency. *Proc. VLDB Endow. 2,* 1 (2009) 754–765.
3. Baltrunas, L., Makcinskas, T., Ricci, F. Group recommendations with rank aggregation and collaborative filtering. *Proc. RecSys 2010*, ACM Press (2010), 119–126.
4. Berkhin, P. A survey on PageRank computing. *Internet Mathematics 2*, 1 (2005), 73–120
5. Berkovsky, S., Freyne, J. Group-based recipe recommendations: analysis of data aggregation strategies. *Proc. RecSys 2010*, ACM Press (2010), 111–118.
6. Breese, J.S., Heckerman, D., Kadie, C. Empirical analysis of predictive algorithms for collaborative filtering, *Proc. UAI-98*, Morgan Kaufmann (1998), 43–52.
7. Deshpande, M., Karypis, G. Item-based top-n recommendation algorithms. *ACM Trans. Information Systems 22*, 1 (2004), 143–177.
8. Gartrell, M., Xing, X., Lv, Q., Beach, A., Han, R., Mishra, S., Seada, K. Enhancing group recommendation by incorporating social relationship interactions. *Proc. GROUP 2010*, ACM Press (2010), 97–106.
9. Gori, M., Pucci, A. ItemRank: a random-walk based scoring algorithm for recommender engines. *Proc. IJCAI 2007*, AAAI Press (2007), 2766–2771.
10. Jameson, A., Smyth, B. Recommendation to groups. In The Adaptive Web, Lecture Notes in Computer Science 4321, Springer (2007), 596–627.
11. Jeh, G., Widom, J. Scaling personalized Web search. *Proc. WWW 2003*, ACM Press (2003), 271–279
12. Kim, H.K., Oh, H.Y., Gu, J.C., Kim, J.K.: Commenders: a recommendation procedure for online book communities. *Electronic Commerce Research and Applications 10*, 5 (2011), 501–509.
13. Konstan J. A., Riedl, J.: Recommender systems: from algorithms to user experience, *User Modeling and User-Adapted Interaction 22*, 1-2 (2012), 101–123.
14. Masthoff, J. Group modeling: selecting a sequence of television items to suit a group of viewers. *User Modeling and User-Adapted Interaction 14*, 1 (2004), 37–85.
15. McCarthy, J.F., Anagnost, T.D. MUSICFX: an arbiter of group preferences for computer supported collaborative workouts. *Proc. CSCW 1998*, ACM Press (1998), pp. 363–372.
16. O'Connor, M., Cosley, D., Konstan, J.A., Riedl, J. PolyLens: a recommender system for groups of users. *Proc. ECSCW 2001*, (2001), 199–218.
17. Sarwar, B., Karypis, G., Konstan, J., Riedl, J. Item-based collaborative filtering recommendation algorithms. *Proc. WWW 2001*, ACM Press (2001), 285-295.
18. Senot, C., Kostadinov, D., Bouzid, M., Picault, J., Aghasaryan, A., Bernier, C.: Analysis of strategies for building group profiles. *Proc. UMAP 2010*, Springer (2010), 40–51.
19. Sotelo, R., Blanco-Fernandez, Y., Lopez-Nores, M., Gil-Solla, A., Pazos-arias, J. TV program recommendation for groups based on multidimensional TV-Anytime classifications. *IEEE Trans. Consumer Electronics 55*, 1 (2009), 248–256.
20. Tong, T., Faloutsos, C., Pan, J.-Y. Fast random walk with restart and its applications. *Proc. ICDM 2006*, IEEE Computer Society (2006), 613 – 622.
21. Yu, Z., Zhou, X., Hao, Y., Gu, J. TV program recommendation for multiple viewers based on user profile merging. *User Modeling and User-Adapted Interaction 16*, 1 (2006), 63–82.