Decision Support Systems 152 (2022) 113663

Contents lists available at ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

Aggregating user preferences in group recommender systems: A crowdsourcing approach

Firat Ismailoglu

Department of Computer Engineering, Artificial Intelligence and Data Science Research Center, Sivas Cumhuriyet University, 58140 Sivas, Turkey

ARTICLE INFO

Keywords:
Recommender systems
Crowdsourcing
Ordinal classification
Bayesian learning
Expectation maximization

ABSTRACT

We present that group recommendations are similar to crowdsourcing, where the responses of different crowd workers are aggregated in the absence of ground truth. With this in mind, we mimic the use of the EM algorithm as in crowdsourcing to aggregate the preferences of group members to estimate group ratings and the expertise levels the group members. Moreover, for the first time in the literature, we cast the problem of estimating group rating as an ordinal classification problem relying on the natural ordering between the ratings, which allows us to define the expertise levels of the members in terms of sensitivity and specificity. In fact, we impose priors on the sensitivity and the specificity scores corresponding to the members, taking a Bayesian approach. We validate the effectiveness of the proposed aggregation method using the CAMRa2011 dataset, which consists of small and established groups, and the MovieLens dataset, which consists of large and random groups.

1. Introduction

A knowledge-based Decision Support Systems called Recommender Systems (RSs) support users in identifying the most relevant items and services in e-commerce. When this involves groups of users, rather than individual users, a special type of RSs called Group Recommender Systems (GRSs) are usually considered [1,23]. GRSs are able to maximize the overall satisfaction of the group by compromising between the preferences of the group members, whereas traditional RSs focus on satisfying individual users, which may be infeasible when the users have conflicting preferences.

In a variety of scenarios, items are consumed by a group of users, so a group-level decision must be made. This has led to GRSs finding numerous applications. *Tourism.* Since travel is usually a group activity, one of the most active application areas is tourism for GRSs. For example, TravelWithFriends recommends travel ideas to a group of friends who want to travel together based on the friends' travel interests, which are defined by 19 different travel categories such as budget travel, LGBT, and nature [11]. *TV-Movie.* GroupReM is a successful GRS for movie recommendations to groups [27]. *Music.* [24] recommends music to a group of people exercising at a gym at a given time. Different from [11] and [27], the groups in [24] consist of people who do not necessarily know each other. GRSs can perform well for such groups as well.

GRSs have several other advantages over traditional personalized recommendations. First, GRSs can mitigate the *cold-start problem*, i.e., the problem of not being able to provide reliable recommendations to users who have recently been added to the system [2]. Even if nothing is known about a user's preferences, GRSs can still provide her with appropriate recommendations as long as her group information is known. In fact, she can receive recommendations made by a GRS for the group to which she belongs. In some cases, creating recommendations for each individual user in a system can be costly. For example, if a company with thousands of customers decides to print flyers to recommend its products, printing a different flyer for each of its customers would result in a massive cost [6]. Here, it would make sense to first group the customers with similar interests and then use a GRS to determine the content of a flyer for each group.

There are two approaches to making group recommendations with GRSs [23]. The first approach consists in merging individual recommendations to create the group recommendations, which is also known as result aggregation. Clearly, in this way, the RS employed produces recommendations to each group member independently, which prevents the occurrence of group profiles. As a result, we lack insight into the actual tastes of the groups [9]. The second approach, on the other hand, considers a group as a single user, aggregating the preferences of group members for each item, which leads to the creation of group profiles. Here, the way the way the aggregation is performed plays the crucial role; as it is the key to extract the group profiles. Hence the literature has witnessed a growing number of aggregation methods of various kinds [35,32,38]. In the following, we briefly review the research on the

E-mail address: fismailoglu@cumhuriyet.edu.tr.

https://doi.org/10.1016/j.dss.2021.113663
Received 19 January 2021; Received in revised form 17 August 2021; Accepted 18 August 2021
Available online 24 August 2021

Table 1
Some significant similarities between crowdsourcing and GRSs.

Crowdsourcing	Group Recommender Systems
Unknown ground truth for the labels	Unknown ground truth for the group preferences
Annotator provides her own opinion	Group member provides her own opinion
Variation between the expertise levels of the annotators	Variation between the expertise levels of the members
Aggregating the opinions on the labels	Aggregating the opinions on the group ratings
Common to deal with ordered labels	Common to deal with ordered ratings

aggregation methods before presenting the aggregation method proposed in this paper.

Existing aggregation methods either employ simple arithmetic operations, such as summation, averaging, or involve more advanced ideas. The aggregation methods that fall into the latter category often comes in three flavors: social relationships, negotiation, and hybridization. The methods in the first group rely on the social relationships between group members to construct the group profile [3,17]. Another trend in GRSs is to view the aggregation task as a negotiation process [4,35]. The basic idea here is to represent each member by an autonomous agent; and then let the agents negotiate with each other with aim of maximizing their individual utility. In practice, this approach usually leads to a complex objective function with a number of parameters that need to be carefully tuned. Hybridized methods, on the other hand, aim to combine multiple aggregation methods to take advantage of each method simultaneously [32,38]. Nevertheless, which aggregation methods should be preferred and in what proportion they are used to maintain a harmony between them remains unclear.

In contrast to the aggregation methods summarized above that are used in GRSs; in this paper, we take an entirely new approach presenting the similarities between group recommendation and crowdsourcing. This holds the promise of creating a new way of aggregating user preferences in GRSs from the perspective of crowdsourcing, which can be briefly defined as the practice of distributing a problem that can be difficult to solve with computers to networked people (a.k.a. annotators, crowd workers) [25]. In fact, we draw several analogies between the task of aggregating labels submitted by different annotators in crowdsourcing, and that of aggregating user preferences in GRSs, summarized in Table 1. Nevertheless, we admit a noticeable difference between crowdsourcing and GRSs. That is, the amount of interaction between the annotators in crowdsourcing is far less than the amount of interaction between group members in GRSs.

In crowdsourcing, a principled and effective method of aggregating the labels obtained from different annotators is to first estimate the annotators' expertise levels, and then incorporate this information into the aggregation [25,28]. For this purpose, the famous EM algorithm is usually employed [10]. Specifically, the aggregated labels are updated in the E-step based on the current expertise levels defined by means of sensitivity–specificity scores, and then in the M-step, these expertise levels are updated based on the current aggregated labels [28]. In this paper, we propose to adapt this idea to GRSs. Thus, as in crowdsourcing, we estimate the expertise levels of group members making use of the current estimate of group ratings in the E-step, and in the M-step, we refine the estimates according to the group ratings taking into account the current expertise levels of the members.

Also, another novelty that we introduce is to approach the problem of aggregating members' preferences in GRSs as an ordinal classification problem. In this sense, we follow the common practice of ordinal classification, where we end up with $r-1$ binary classification problems for r unique ratings as explained in [15]. This gives us the opportunity to define sensitivity and specificity for the group members when we asses their level of expertise. What is more, we take a Bayesian approach imposing priors on the sensitivity and specificity of the group members. This allows us to incorporate any prior information about the members, such as demographic characteristics of the members and the trust networks in the groups, in the process of estimating their expertise levels.

1.1. The objectives and the achievements of the research

The objectives of this research include the following:

1. to estimate the expertise levels of group members and to leverage this information in aggregating their preferences,
2. to be able to incorporate any background knowledge about group members into the estimation of group preferences,
3. to assess the similarity between the task of aggregating user preferences in GRSs and that of aggregating labels in crowdsourcing.

To achieve the objectives stated above, we took the following actions in order:

1. We employed the EM algorithm which allows us to refine estimates of group preferences and that of expertise levels of group members concurrently.
2. We took a Bayesian approach imposing priors on the sensitivity and specificity scores of group members, which forms a basis for determining group preferences. The Bayesian approach allows us to incorporate prior assumptions about the members, such as how reliable they are.
3. We were able to successfully adopt the use of the EM algorithm in crowdsourcing to GRSs, which supports the claim that the task of label aggregation in crowdsourcing and that of preference aggregation show parallelism.

We organize the remainder of this paper as follows. In Section 2, we survey the work related to GRSs. In Section 3, we formulate the problem of the aggregation. In Section 4, we describe the proposed aggregation method in detail. In Section 5, we present the experiments conducted to compare the proposed method with the baseline aggregation methods. Finally, in Section 6, we conclude this study and discuss how to extend the scope of the proposed method.

2. Related works

We divide this section into four parts. In the first part, we introduce the four types of groups considered in GRSs to define what we mean by the notion of groups throughout the paper. Next, we give an insight into crowdsourcing as the proposed method has its roots in this area. We reserve the third part to discuss the state-of-the art aggregation methods; and we conclude the section with the current research on GRSs.

2.1. The types of groups in GRSs

In general, the groups in GRSs emerge in four different kinds [5,38]. In the following, we summarize them briefly.

- **Established group**: consists of members who share long-term common interests, such as family, group of colleagues. A notable study of the established groups in GRSs is a challenge called Context-Aware Movie Recommendation (CAMRa2011) [29]. In this competition over 45 teams competed to recommend the most appropriate movies to households.
- **Occasional group**: a group of people that come together for a common aim at a particular moment; such as running clubs. In comparison with the established groups, occasional groups are more likely to be encountered in GRSs, as the mere presence of short-term common interests is sufficient to define such groups. A prominent study addressing the occasional groups in GRSs is TravelWithFriends [11], which recommends travel destinations to people traveling

together. A more recent and interesting study is [40], which suggests upcoming conferences to a group of authors of a scientific paper.

- **Random group**: a group of people in a common environment at a particular moment only by chance; such as people shopping in a particular super market at a particular time. For example, MusicFX [24] recommends music to a group of people doing exercises in a gym, which can be considered as random group.
- **Automatically identified group**: is not a pre-defined group, unlike the other three, but is detected automatically based on members' preferences, often using a clustering algorithm (e.g. the k-means algorithm). Not being pre-defined, [3] considers these groups as online groups, while referring to the other groups as offline groups. Also remember that one of the main motivations for producing group recommendation is to avoid the high computational cost that personalized recommendation would lead to. Hence, in the absence of pre-defined groups, it is popular to create online groups, i.e., automatically identified groups, to benefit from group recommendation [37].

2.2. Crowdsourcing

Crowdsourcing can be defined as a web-based business practice in which a complex task is outsourced to a crowd of web workers (i.e., crowdworkers) to leverage collective intelligence [13,7]. In doing so, crowdsourcing obtains a solution from all workers for the problem at hand. From a machine learning perspective, such solutions are referred to as labels, which is why we use the term label throughout this paper. Once the labels are obtained, they must be aggregated to obtain the final label. However, it is clear that the workers often have different levels of expertise, which makes it challenging to perform the aggregation. Even worse, some workers may be just spammers or adversaries. Hence a crowdsourcing system should be able to assess the expertise levels of workers and incorporate this information into the aggregation process [16]. In this sense, a simple approach consists of first asking workers a small number of trap questions (i.e., questions whose answers are already known) and then eliminating workers based on their answers to these questions. A more advanced approach consists of a sequence of rounds, where in each round, the probabilities of the possible labels for each question and the expertise levels of the workers are updated concurrently, often using the EM algorithm [19].

The success of crowdsourcing lies in the use of collective intelligence, which leads to cooperation, teamwork, and consensus [7]. Each worker brings fresh insights to problems; in fact they cross-fertilise one another, thus the final solution is often superior to the one devised by lone individuals [16]. Additionally, crowdsourcing can be used as a means for *human computation*, which aims to use human efforts to solve problems that computers alone cannot solve. In this way, one can combine the computational power of computers with the perceptual and cognitive abilities of humans for non-routine tasks such as understanding texts, gathering information about the physical world, etc.[21]. Another merit of crowdsourcing is that it enables problem solving based on the divide and conquer principle, i.e. a given complex problem is broken down into micro-problems that are potentially easier to tackle.

The advantages of crowdsourcing described above have led to a number of applications in a variety of fields, including smartphones, public health, logistics, and agriculture. Nevertheless, Wikipedia is usually considered as the most representative application of crowdsourcing, where Internet users are intrinsically motivated to contribute to the creation of articles, leading to continuous improvement of the platform. For more applications of crowdsourcing, please refer to [18]. Finally, with the dramatic increase in the use of social media and the other new media technologies, more and more crowdworkers are participating in the problem solving, which makes crowdsourcing increasingly popular.

2.3. The aggregation methods used in GRSs

As emphasized in the introduction, the aggregation process is at the heart of GRSs, as it is the key to elicit the actual preferences of a group. This has motivated various researches to develop aggregation methods of different types [32,3]. As in [30], we classify these methods in two groups: the *baseline aggregation methods* and *the advanced aggregation methods*. The methods that fall into the latter group have been developed more recently and consider more complex ideas, while the methods in the former are somewhat old-fashioned and simple, but it does not necessarily mean that their performance is inferior.

The baseline aggregation methods can be divided into three subgroups: consensus-based (democratic), majority-based, and borderline. Briefly, *the consensus-based methods* incorporate the ratings of all group members to produce the group rating using simple arithmetic operations. Additive Utilitarian (AU) simply sums the ratings that an item has received, whereas Multiplicative (Mul) [8] prefers to multiply the ratings of all members. Average (Avg) [8] considers the arithmetic mean of the ratings given to an item. *The majority-based methods* are in favor of the most popular items [30]. For example, when estimating the group preference for an item, Approval Voting (AV) counts the number of group members who gave the item a high rating, e.g. above 3 on a five-star scale. Finally, *the borderline methods* select a single rating from the ratings that an item received from group members and then consider it as the group rating for that item. We include three aggregation methods in this category: Least Misery (LM), Most Pleasure (MP), and Most Respected Person (MRP) [23]. LM considers the lowest rating an item received as the group rating; whereas MP selects the highest rating. In contrast, MRP identifies the most respected/influential person within a group, and considers his or her rating as the group rating, which raises the question of how to select the most respected person [30].

There are also some advanced aggregation methods in the literature that involve relatively sophisticated ideas. We divide the methods in this group into three subgroups: social relationship-based, negotiation-based, and hybridized methods. *The relationship-based methods*, as the name implies, emphasize the importance of social relationships among group members in constructing the group profile. Specifically, [17] asks members to take the TKI test to measure members' assertiveness and cooperativeness. Using these two measures, a personality score is estimated for each member [36]. The original user–item matrix is updated using a linear combination of the personality values and the social relationships. Another relationship-based method IBGR [3] calculates the influence of members on each other. In doing so, it introduces similarity and trust metrics and determines a leader for each group that achieves the highest level of trust and similarity. Finally, the members and the leaders are weighted according to their level of influence.

The negotiation-based methods view group recommendation as a negotiation process in which each group member wants the group preferences to be as close as possible to his or her own preferences [35]. Here, each member is represented by an agent that interacts with the other members during the aggregation. This approach also shares the advantages of the automated negotiation, such as consistency, strategy, and lack of emotion. [35] introduces a method called TruGRC. To guide the negotiation process, TruGRC creates a virtual coordinator whose goal is to balance members' preferences with those of the group. In fact, the virtual coordinator can adjust the members' preferences with the goal of maximizing the overall benefit of the group. However, in practice, the negotiation process leads to a complex objective function with a large number of parameters need to be tuned. Finally, *the hybridized methods* combine several aggregation methods to obtain a hybrid method [8,32,38]. Here, the main motivation is to take the advantages of the basic aggregation methods while creating a synergy between them. Upward Leveling (UL) [32] combines Avg and AV; whereas Agreement without Uncertainty (AwU) [38] combines AU and AV.

Table 2
The notation used in this paper.

$\mathcal{U}$	Set of users	$\mathcal{I}$	Set of items
$\mathcal{G}$	Set of groups	G_g^i	Subset of users in G_g that rated item i
R	Rating matrix	R_{bin}^j	jth binary rating matrix
m	Number of users	n	Number of items
$r_{u,i}$	Rating of user u for item i	$r_{G_g,i}$	Rating of group G_g for item i
μ_i	Probability of group rating being 1 for item i	p	Prevalence of the positive rating
α_u	Sensitivity of user u	β_u	Specificity of user u
μ_i^j	μ_i for jth binary rating matrix	α_u^1	First param. of the Beta dist. for α_u

2.4. Current research on Group Recommender Systems

We conclude this chapter with a review on current research on GRSs. As in the past, most of the current research on GRSs focuses on aggregation methods. In addition to aggregation methods that rely only on user preferences to determine group tastes, approaches that consider group dynamics and behaviour are also becoming popular. Nevertheless, they require contextual information about the items and the users, which introduces additional costs [12,22].

In some recent publications, the distribution of ratings that an item has received from the members of a group is taken into account in determining the degree to which the group members agree with that item [32,38]. This is done to reduce the likelihood that items with a high deviation will be included in the recommendation list, which in turn will result in as many members of a group as possible being satisfied.

Another burgeoning trend in GRSs is to incorporate the group identification task into the group recommendation, so that the resulting groups benefit maximally from the group recommendations [31]. Finally, the use of GRSs goes beyond the traditional application domains. [26] is interested in retail and recommends products for groups of offline stores. Unlike the other applications of GRSs, the study in [26] also considers the quantity and repeated recommendation of products, whereas in the traditional setting, each item is recommended only once.

3. Problem formulation

In a recommender system, we are given a set of users $\mathcal{U}$ with m users, and a set of items $\mathcal{I}$ with n items. As the users rate the items, a user – item rating matrix $R \in \mathbb{R}^{m\times n}$ with entries $r_{u,i}$ is constructed, where $r_{u,i}$ denotes the rating of user u for item i. Also, it is typical that a vast majority of the entries of R is missing, as the users provide ratings for just a tiny fraction of the items available. We shall denote such entries with $\perp$.

In a group recommendation scenario, $\mathcal{U}$ is partitioned into (often) disjoint groups $\mathcal{G} = \{G_1, \ldots, G_k\}$, with $G_l \cap G_s = \varnothing$ ($l \neq s$ and $l, s \in \{1, \ldots, k\}$) and $\bigcup_{g=1}^{k} G_g = \mathcal{U}$. Here, the goal is to recommend items for the groups other than for the individual users. To achieve this, two main approaches exist. One is to aggregate individual recommendations and the other is to aggregate users' ratings to attain group profiles. In the present study, we follow the latter. To denote the rating of a group G_g for an item i, we use notation $r_{G_g,i}$. Also, to denote the subset of users in G_g that provide a rating for item i, we shall use G_g^i. When it comes to estimate $r_{G_g,i}$, the literature offers a wide range of aggregation methods, please refer to the related literature review in 2.3. Below, we explain how we estimate $r_{G_g,i}$ using the proposed method. Finally, we provide Table 2 to summarize the notation used throughout in this paper.

4. EM-based aggregation method

4.1. From a multi-scale ratings matrix to a set of binary rating matrices

In a recommender system, users provide either binary ratings (i.e., like or dislike) or ratings in a larger range indicating to what extent they like an item. For example, in 5-star scale, $r_{u,i}$ ranges over the elements of $\{1, \ldots, 5\}$. In this paper, we shall refer to the latter case as multi-scale rating and deal with it, as is more common in GRSs. In fact, we approach

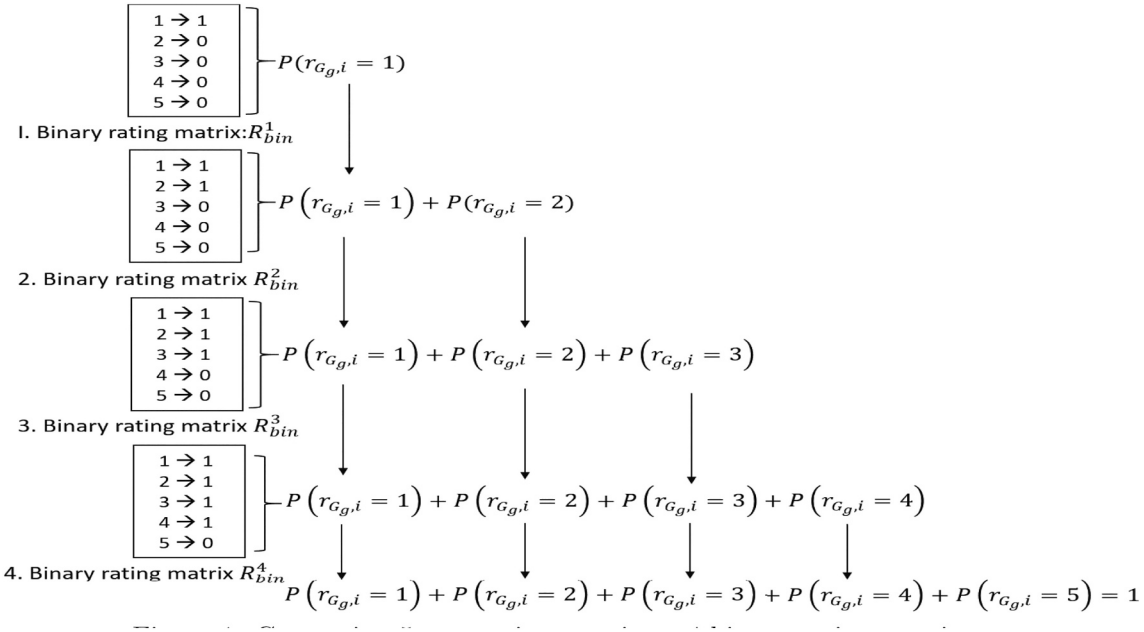


Fig. 1. Converting 5-star rating matrix to 4 binary rating matrices.

the problem of aggregating multi-scale ratings of an item as an *ordinal classification* problem, which constitutes one of the main contributions of this paper. We argue that considering one of r ($r > 2$) unique ratings as the group rating for an item is analogous to assign one of r ordered classes to a test instance; although in case of GRSs, we lack a set of features for the items [15,28].

Following the common practice in ordinal classification, where a $r-$class classification problem is transformed into $(r-1)$ binary classification problems, we obtain $(r-1)$ user – item matrices that consist of binary ratings, for a given user – item matrix that has r unique ratings. We denote the resulting binary matrices as R^j_{bin} ($j \in \{1,\ldots,r-1\}$). The first resulting user – item matrix (R^1_{bin}) is utilized to estimate the probability that (aggregated) group rating equals to 1. To do so, the ratings above 1 are replaced with 0, whereas those equal to 1 remain unchanged. In the second resulting matrix, the ratings above 2 are replaced with 0, while the ratings of 1 or 2 are replaced with 1. Using the second matrix (R^2_{bin}), we can estimate the probability that group rating equals to 1 or 2. Subtracting the probability obtained from the first matrix, one can readily estimate the probability of 2 being group rating for the item. The remaining probabilities (e.g. probability of 3, 4, etc.) can be obtained in a similar manner. Fig. 1 illustrates this idea for 5-star scale.

When it comes to estimate the probability of positive class in each resulting user–item matrix that has binary ratings, we employ an EM-based algorithm as in crowdsourcing [39]. [28] proposed an EM-based approach for crowdsourcing, where at each iteration the level of expertise of workers are updated relying on the answer of the group at that iteration; and then the answer of the group is refined relying on the expertise levels of the workers that have recently updated. In the present work, we propose to adopt this approach for GRSs, so that we can reflect the expertise levels of group members when aggregating their preferences. In what follows, we explain the proposed idea in detail.

4.2. A maximum likelihood model

Given a binary user–item matrix $R_{bin} \in \{0,1\}^{|G_g|\times n}$ for a group G_g, we aim to estimate the probability that the group rating is equal to 1 for each item. We shall denote such probability as

$$\mu_i = P(r_{G_g,i} = 1), \tag{1}$$

$i \in \{1,\ldots,n\}$. These probabilities form the set $\boldsymbol{\mu} = \{\mu_i\}_1^n$. In addition, we also aim to estimate the sensitivity and the specificity of each user in the group. The sensitivity for user $u \in G_g$ is calculated as the probability that she likes an item thus provides a rating of 1, given that the group likes the item:

$$\alpha_u = P(r_{u,i} = 1 | r_{G_g,i} = 1). \tag{2}$$

Likewise, the specificity for user $u \in G_g$ indicates the probability that she dislikes an item, thus rates as 0, when the group dislikes the item:

$$\beta_u = P(r_{u,i} = 0 | r_{G_g,i} = 0). \tag{3}$$

The sensitivity and the specificity values of the users in G_g form the sets $\boldsymbol{\alpha} = \{\alpha_u\}_1^{|G_g|}$ and $\boldsymbol{\beta} = \{\beta_u\}_1^{|G_g|}$ respectively. Clearly, the users with high sensitivity and high specificity tend to reflect the true tastes of the group, and hence should be prioritized during the aggregation.

Under the assumptions that the items are rated independently by group members; and that the members are independent of each other when rating the items, then the likelihood of the parameters $\boldsymbol{\mu}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ given the user–item matrix R is as follows:

$$\begin{aligned}
P(R|\boldsymbol{\mu},\boldsymbol{\alpha},\boldsymbol{\beta}) &= \prod_{i\in\mathcal{I}}\prod_{u\in G_g^i} P(r_{u,i}|\mu_i,\alpha_u,\beta_u) \\
&= \prod_{i\in\mathcal{I}}\prod_{u\in G_g^i} P(r_{u,i}|r_{G_g,i}=1,\alpha_u)P(r_{G_g,i}=1)+P(r_{u,i}|r_{G_g,i}=0,\beta_u)P(r_{G_g,i}=0) \\
&= \prod_{i\in\mathcal{I}}\prod_{u\in G_g^i} P(r_{u,i}|r_{G_g,i}=1,\alpha_u)(\mu_i)+P(r_{u,i}|r_{G_g,i}=0,\beta_u)(1-\mu_i) \\
&= \prod_{i\in\mathcal{I}}\prod_{u\in G_g^i} (\alpha_u)^{r_{u,i}}(1-\alpha_u)^{1-r_{u,i}}(\mu_i)+(\beta_u)^{1-r_{u,i}}(1-\beta_u)^{r_{u,i}}(1-\mu_i) \\
&= \prod_{i\in\mathcal{I}}(\mu_i)\prod_{u\in G_g^i}(\alpha_u)^{r_{u,i}}(1-\alpha_u)^{1-r_{u,i}}+(1-\mu_i)\prod_{u\in G_g^i}(\beta_u)^{1-r_{u,i}}(1-\beta_u)^{r_{u,i}}
\end{aligned} \tag{4}$$

To estimate the set of parameters $\{\boldsymbol{\mu},\boldsymbol{\alpha},\boldsymbol{\beta}\}$ that maximize the log of (4), i.e., the maximum-likelihood estimate, we employ the well-established Expectation Maximization (EM) algorithm [10]. In the following subsection, we detail the application of the EM algorithm for the above likelihood function.

4.3. The use of the EM algorithm for the group recommendation systems

For each item, we attempt to estimate the probability that group giving a positive rating by relying on the performances of the users in terms of their sensitivity and specificity scores. However, both the actual probabilities and the sensitivity–specificity scores are unknown to us. Following [28], we refine the probability of positive rating for each item based on the given sensitivity and specificity scores of users and also the given prevalence of positive ratings in the E step; and then in the M-step we update the sensitivity and specificity scores given the probabilities of positive ratings.

Moreover, it is very likely that some users in the group have influence on the other members, who are referred to as opinion leaders [3] or influencers [34]. On the other hand, some users in a group are just spammers who randomly post reviews. If such information about the members is available in advance, this information should be considered in the aggregation. To account for this, we take a Bayesian approach imposing priors on the sensitivity and specificity scores. More specifically, we use the beta distribution for the sensitivity and specificity as they indicate the probability of a binary event. As such, α_u ($u \in G_g$) is distributed according to the beta distribution parametrized by α_u^1 and α_u^2, i.e., $\alpha_u \sim \text{Beta}(\alpha_u^1, \alpha_u^2)$. Similarly, we assume that β_u ($u \in G_g$) follow the beta distribution with the parameters β_u^1 and β_u^2, i.e., $\beta_u \sim \text{Beta}(\beta_u^1, \beta_u^2)$. Altogether, we explain the steps of the proposed EM algorithm to be used for the GRSs as follows:

1. For each item, initialize μ_i with its maximum likelihood estimator: $\mu_i = \frac{1}{|G_g^i|}\sum_{u\in G_g^i} r_{u,i}$.
2. Given the current estimate of μ_i, estimate the sensitivity and the specificity scores of the users as well as the prevalence of the positive rating p as follows:

$$\begin{aligned}
\alpha_u &= \frac{\alpha_u^1 - 1 + \sum_{i\in\mathcal{I}_u}\mu_i r_{u,i}}{\alpha_u^1 + \alpha_u^2 - 2 + \sum_{i\in\mathcal{I}_u}\mu_i}, \\
\beta_u &= \frac{\beta_u^1 - 1 + \sum_{i\in\mathcal{I}_u}(1-\mu_i)(1-r_{u,i})}{\beta_u^1 + \beta_u^2 - 2 + \sum_{i\in\mathcal{I}_u}(1-\mu_i)}, \\
p &= \frac{\sum_{i=1}^n \mu_i}{n},
\end{aligned} \tag{5}$$

   where $\mathcal{I}_u$ denotes the set of items rated by user u.
3. Given $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and p update the probability of positive rating, μ_i, as:

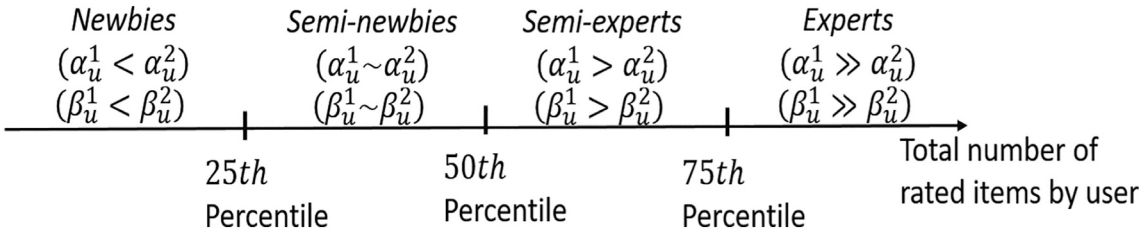


Fig. 2. Setting the hyperparamaters for the sensitivity and for the specificity based on the number of ratings provided.

$$\mu_i = \frac{p \cdot \left(\prod_{u \in G_g^i} (\alpha_u)^{r_{u,i}} (1-\alpha_u)^{1-r_{u,i}}\right)}{p \cdot \left(\prod_{u \in G_g^i} (\alpha_u)^{r_{u,i}} (1-\alpha_u)^{1-r_{u,i}}\right) + (1-p) \cdot \left(\prod_{u \in G_g^i} (\beta_u)^{1-r_{u,i}} (1-\beta_u)^{r_{u,i}}\right)}. \tag{6}$$

4. Repeat steps 2 and 3 until $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, p and μ_i converge.

We note that one can also impose a prior on the prevalence p. For example, assuming that p has the Beta distribution with parameters, p_1 and p_2, then p can be updated with the following formula:

$$p = \frac{p_1 - 1 \sum_{i=1}^{n} \mu_i}{p_1 + p_2 - 2 + n}.$$

Considering a prior for the prevalence of the positive rating could be useful, if the user-rating matrix of interest is binary by its nature. However, in our case, the binary classes that emerge in the binary user–item matrices are somewhat arbitrary, thus no prior information is available about the positivity or the negativity of the ratings.

4.3.1. Adjusting the hyperparameters for the sensitivity and for the specificity

Apparently, setting α_u^1 as a higher value with respect to α_u^2 leads to a high α_u, i.e., a high sensitivity score for user u. In much the same way, if one wishes to have a high β_u for user u, then she needs to consider a relatively larger β_u^1 as compared to β_u^2. This, in turn, allows us to be biased towards some particular users in a group, such as influencers, spammers, or adversaries, in determining the group opinion for an item. In some group recommendation scenarios, information about demographic features of the users, or about social networks in the groups is already available [34]. If this is the case, one can readily tune the hyperparameters of α_u and β_u making use of such information. However, in the present study, to be more general, we assume to have a user–item matrix only. Still, it may be possible to weight the users, without relying on any external information.

We claim that the level of expertise of a user is positively correlated with the number of ratings she provided. Following this, we rank the users based on the total number of items they rated in ascending order, and then divide them into four quarters. Those in the first quartile are *newbies* that rated relatively a few items, which is to say that they are the least experienced users. For such users, we propose to assign α_u^1 and α_u^2 so that α_u^1 is lower than α_u^2. We consider those in the second quartile as semi-newbies and set α_u^1 and α_u^2 so that α_u^1 is equal or less than α_u^2. For those in the third quartile, we consider them as semi-experts and assign a larger α_u^1 as compared to α_u^2. Finally, we treat those in the fourth quartile as the experts, as the number of rated items is the highest for such users. This leads us to set a much larger value for α_u^1, as compared to α_u^2. We note that we use the same reasoning in tuning β_u^1 and β_u^2. Fig. 2 summarizes how we tune the hypermarameters based on the number of the rated items.

Table 3
A user-item matrix and the binary rating matrices derived from it.

	Orj. rating mat.(R)		R^1_{bin}		R^2_{bin}		R^3_{bin}		R^4_{bin}	
	i_1	i_2	i_1	i_2	i_1	i_2	i_1	i_2	i_1	i_2
u_1	5	5	0	0	0	0	0	0	0	0
u_2	5	4	0	0	0	0	0	0	0	1
u_3	1	2	1	0	1	1	1	1	1	1
u_4	3	2	0	0	0	1	1	1	1	1
u_5	4	1	0	1	0	1	0	1	1	1
u_6	2	3	0	0	1	0	1	1	1	1
u_7	4	3	0	0	0	0	0	1	1	1

We name the proposed aggregation method based on the keywords: EM, Crowdsourcing and Ordinal Classification. We thus name it as ECOagg, and give Algorithm 1 that shows the pseudocode associated with ECOagg.

Finally, we provide a toy example to illustrate how ECOagg works. Suppose that we have seven users and two items; and that the users provide ratings to the items from 1 to 5 exclusively, as shown in the leftmost part of Table 3. Here, five unique ratings lead to having four binary rating matrices, i.e., $R^j_{\text{bin}}, (j \in \{1, \ldots, 4\})$. Without assuming any prior knowledge of the users, thus setting the values of $\alpha_u^1, \alpha_u^2, \beta_u^1$ and β_u^2 equal to 1 for all users, ECOagg calculates the probability of positive ratings for the first item, i.e., $\mu_1^1, \mu_1^2, \mu_1^3$ and μ_1^4, using the resulting binary matrices as 0.142, 0.171, 0.296 and 0.714, respectively, after four iterations. Using these values, we estimate the probabilities of the group ratings of the first item as

$$P(r_{G_g,1} = 1) = \mu_1^1 = 0.142,$$
$$P(r_{G_g,1} = 2) = \mu_1^2 - \mu_1^1 = 0.171 - 0.142 = 0.029,$$
$$P(r_{G_g,1} = 3) = \mu_1^3 - \mu_1^2 = 0.296 - 0.171 = 0.125,$$
$$P(r_{G_g,1} = 4) = \mu_1^4 - \mu_1^3 = 0.714 - 0.296 = 0.418,$$
$$P(r_{G_g,1} = 5) = 1 - \mu_1^4 = 1 - 0.714 = 0.286.$$

Hence the winning rating is 4, which serves as the group rating for the first item. Running ECOagg for the second item yields the values for μ_2^1, μ_2^2, μ_2^3 and μ_2^4 as 0.142, 0.577, 0.824 and 0.911, respectively. The probabilities of the group ratings of second item are, $P(r_{G_g,2} = 1) = 0.142$, $P(r_{G_g,2} = 2) = 0.435, P(r_{G_g,2} = 3) = 0.247, P(r_{G_g,2} = 4) = 0.087$ and $P(r_{G_g,2} = 5) = 0.089$. The winning rating for the second item is, thus, 2.

Algorithm 1. ECOagg: An EM-Based Aggregation Method for Group Recommender System

Algorithm 1 ECOagg: An EM-Based Aggregation Method for Group Recommender System

Input: User–item rating matrix of $R \in \{\perp, 1, 2, ..., c\}^{m \times n}$ for group G_g.
Output: Aggregated group rating for each item: $\hat{r}_{G_g,i}, (i \in \{1, ..., n\})$.
Define (c-1) binary user–item matrices:
for j in $\{1, 2, \cdots, c-1\}$ **do**
  if $r_{u,i} > j$ **then** ▷
    Assign the (u, i) entry of R^j_{bin} as 0.
  else
    Assign the (u, i) entry of R^j_{bin} as 1.
  end if
end for
For each R^j_{bin} estimate the probability of positive rating for each item
for j in $\{1, 2, ..., c-1\}$ **do**
  for i in $\{1, 2, ..., n\}$ **do**
    Calculate μ^j_i as shown in (6) ▷ *Relying on the estimated sensitivity and specificity scores of the users.*
  end for
end for
for i in $\{1, 2, ..., n\}$ **do**
  $\hat{r}_{G_g,i} = argmax\{\mu^1_i, (\mu^2_i - \mu^1_i), ..., (\mu^{c-1}_i - \mu^{c-2}_i - \cdots - \mu^1_i), (1 - \mu^{c-1}_i)\}$ ▷ *rating with the highest probab.*
end for
return $\hat{r}_{G_g,i}, (i \in \{1, ..., n\})$.

Table 4
The statistics of CAMRa2011 and MovieLens datasets.

Dataset	User	Item	Rating	Sparsity
CAMRa2011	602	7710	116,344	97.5%
MovieLens	943	1682	100,000	93.7%

5. Experiments

5.1. The datasets used

As mentioned in the literature review, the groups considered in GRSs can be of four types: established, occasional, random and automatically-identified. Due to the page limitation, we only address the established groups and the random groups in the experiments in this paper. For the established groups, we employ CAMRa2011 dataset, which was published as part of a competition on Context-aware Movie Recommendation at RecSys2011 [29]. This dataset contains movie ratings of households from 290 families with families consisting of 2.08 members on average. We use the famous MovieLens dataset[1] to form the random groups. For both datasets, the ratings take an integer value from 1 to 5, which indicates how much a user likes a movie. Table 4 shows the statistics of these two datasets.

5.2. The benchmark aggregation methods

Here we present the aggregation methods that serve as the basis for comparing the performance of ECOagg. In the literature review, we highlighted that the baseline aggregation methods in GRS come in three flavors: consensus, majority and borderline. We select one aggregation method from each for comparison. Concretely, we choose *Average*(*AVG*) from the consensus-based methods due to its high popularity, *Aproval Voting*(*AV*) from the majority based methods due to its high accuracy and *Least Misery*(*LM*) from the borderline methods since LM ensures that all members are satisfied.

Recall that in Section 2.3, we classified the advanced aggregation methods in GRSs into three categories: social relationship-based, negotiation-based, and hybridized. In addition to the baseline methods explained above, we also consider three advanced aggregation methods for the experiments. We select Instance-Based Group Recommendation (IBGR) [3] from the social relationships-based methods; because unlike the other aggregation methods in this category, IBGR does not rely on any external information to construct the social/trust networks in groups, and therefore can work for the datasets we are interested in. Also, among the negotiation-based methods we choose Trust-Aware Group Recommendation (TruGRC) [35] because it has a virtual coordinator who can adjust members' preferences with the purpose of maximizing the overall utility of the group. Finally, we select Upward Leveling (UL) [32] as a hybridized method, since it combines two state-of-the-art aggregation methods, i.e., *AVG* and *AV*, along with standard deviation.

TruGRC and UL have some parameters that need to be tuned. Specifically, for TruGRC, we set the learning rate for the gradient descent algorithm η to 0.001, the dimension of the latent vectors d to 10 corresponding to the user and items, and two regularization parameters λ and λ_α to 0.01 and 1, respectively; since these parameters are reported as the optimal parameters for TruGRC [35]. Nevertheless, we ran TruGRC with different settings for the datasets of interested; but did not observe any significant difference in terms of the evaluation metrics. As for UL, the authors report that UL performs best when the weights corresponding *AVG*, *AV*, and SD are set to 0.4, 0.2 and 0.4, respectively [32]. We thus use the same weights in all experiments.

[1] https://www.grouplens.org

Table 5
The nDCG results for the CAMRa2011 dataset.

Aggregation method	top-*N*			
	1	3	5	10
AV	0.839	0.842	0.847	0.855
AVG	0.887	0.901	0.905	0.902
LM	0.883	0.895	0.899	0.9062
UL	0.901	0.904	0.906	**0.913**
TruGRC	0.889	0.895	0.888	0.883
IBGR	0.887	0.903	0.904	0.903
ECOagg	**0.911**	**0.911**	0.901	0.905

Statistically significant results are given in bold

5.3. Evaluation metrics and evaluation methodology

To evaluate the performance of the aggregation methods we employ two metrics: the normalized Discounted Cumulative Gain (*nDCG*) [5], and a fairness metric called m-proportionality (*m-prop*) [33]. The former is used to assess the extent to which members are satisfied with the group recommendations at the individual level, and the latter is used to assess the satisfaction of the entire group In the following, these two metrics are explained in detail.

To compute the *nDCG* score for a user, one needs a recommendation list of N items, denoted $\{i_1, i_2, \ldots, i_N\}$, and the user's actual ratings for these items. At this point, we note that we estimate the actual ratings that are missing using the SVD++ algorithm as it provides high accuracy by taking into account the user and item biases in the prediction, which is not the case for the other matrix factorization methods, such as SVD [20]. Before computing *nDCG*, we first compute the DCG for user u as follows:

$$\mathrm{DCG}_u = r_{u,i_1} + \sum_{l=2}^{N} \frac{r_{u,i_l}}{log_2 l}.$$

To normalize the DCG score, we divide it by the maximum DCG score which can be achieved if the recommendation list of N items consists of the user's top N favorites. Such a cDCG score is usually referred to as the ideal DCG, i.e., $IDCG_u$. The normalized DCG score is then computed as

$$\mathrm{nDCG}_u = \frac{\mathrm{DCG}_u}{\mathrm{IDCG}_u}. \tag{7}$$

Note that the *nDCG* metric reflects not only the actual ratings of the recommended items, but also the order of the recommended items. Concretely, the items at the top of the recommendation list play a more important role in the *nDCG* metric. Finally, to obtain the *nDCG* score of a group, we simply average the *nDCG* scores of the group members.

In GRSs, it is also important to gratify group members as evenly as possible. To determine the extent to which this is achieved, a metric called *m-prop* is usually considered [14]. Concretely, *m-prop* measures how fair the recommendation list is for the entire group. This is estimated for the group G_g by the following formula:

$$m - \mathrm{prop}_{G_g} = \frac{\sum_{u \in G_g} [\![\sum_{l=1}^{N} [\![r_{u,i_l} \geq t]\!] \geq m]\!]}{|G_g|}, \tag{8}$$

where t is a threshold. If the actual rating of the user u for the recommended item i_l exceeds the threshold t, we conclude that the user gratifies the recommended item. Thus, the metric $m - prop$ thus can be read as the rate of users in a group gratifying at least m of N recommended items.

To compare the performances of the aggregation methods evaluated by means of the *nDCG* and the $m - prop$, we use the 5-fold cross-validation technique in the following way. We randomly divide the set of items into five groups of equal size. At each iteration, we use one set of the items as a test set, and estimate the group ratings of the items in it. Having done that, we recommend the top-N items that achieved the highest group ratings to the group. Based on the recommended items we calculate the *nDCG* and the $m - prop$ scores per group. Finally, we average these scores to estimate the performance of the aggregation method at that iteration.

Table 6
The nDCG results for the MovieLens dataset.

top-*N*	Aggregation method	Number of Groups (k)					
		4	8	16	32	64	128
	AV	0.775	0.782	0.783	0.783	0.783	0.789
	AVG	0.663	0.681	0.691	0.718	0.735	0.771
	LM	0.686	0.677	0.689	0.714	0.745	0.771
1	UL	0.662	0.671	0.708	0.749	0.781	0.807
	TruGRC	0.789	0.788	0.786	0.792	**0.799**	0.803
	IBGR	0.752	0.759	0.752	0.748	0.774	0.801
	ECOagg	**0.804**	0.799	**0.799**	0.794	0.791	0.791
	AV	0.771	0.773	0.777	0.779	0.781	0.788
	AVG	0.697	0.698	0.702	0.728	0.749	0.778
	LM	0.693	0.698	0.696	0.723	0.751	0.778
3	UL	0.684	0.687	0.709	0.747	0.772	**0.811**
	TruGRC	0.774	0.777	0.781	**0.785**	0.792	0.802
	IBGR	0.753	0.765	0.761	0.761	0.779	0.801
	ECOagg	**0.792**	**0.791**	0.781	0.777	0.777	0.786
	AV	0.781	0.779	0.781	0.784	0.787	0.791
	AVG	0.704	0.708	0.711	0.736	0.756	0.791
	LM	0.701	0.704	0.706	0.732	0.758	0.785
5	UL	0.692	0.697	0.712	0.746	0.773	0.805
	TruGRC	0.781	0.781	**0.787**	0.789	0.794	0.807
	IBGR	0.763	0.774	0.769	0.771	0.784	0.803
	ECOagg	**0.789**	0.781	0.779	0.788	0.793	0.801
	AV	0.788	0.781	0.795	0.796	0.798	0.802
	AVG	0.731	0.736	0.735	0.754	0.776	0.807
	LM	0.715	0.721	0.723	0.746	0.772	0.801
10	UL	0.711	0.715	0.726	0.756	0.782	0.803
	TruGRC	0.793	0.782	0.796	0.801	0.804	**0.814**
	IBGR	0.774	0.782	0.783	0.783	0.795	0.802
	ECOagg	**0.806**	**0.798**	0.798	0.804	0.804	0.805

Statistically significant results are given in bold

5.4. Results and discussion

5.4.1. The nDCG results

We give the *nDCG* scores of the aggregation methods for the CAMRa2011 dataset in 5 and for the MovieLens dataset in Table 6. Furthermore, in both tables, relying on the one-tailed t-tests, we highlight the best result if it is statistically more significant than the second best at a 95% confidence level under the same setting.

At first sight, it is clear that the choice of the aggregation method in a GRS is a non-trivial task, especially when dealing with large groups, which corresponds to small k values. In fact, we observe a 15% difference between the highest and lowest *nDCG* scores when the groups in the MovieLens dataset are crowded. Also recall that the groups in this dataset are formed randomly, and therefore have a heterogeneous structure. This, in turn, makes it difficult to aggregate the different preferences of the members. On the other hand, as mentioned earlier, the groups in the CAMRa2011 are established and much smaller; that is,

Table 7
The *m-prop* results for the CAMRa2011 dataset ($N = 5$).

Aggregation method	m				
	1	2	3	4	5
AV	0.981	0.961	0.864	0.687	0.511
AVG	0.979	0.941	0.871	0.755	0.545
LM	0.976	0.928	0.859	0.741	0.531
UL	0.984	0.951	0.885	0.785	0.583
TruGRC	0.982	0.962	0.897	0.785	0.557
IBGR	0.982	0.958	0.885	**0.827**	0.587
ECOagg	0.984	0.961	**0.906**	0.819	0.589

Statistically significant results are given in bold

the groups are much more homogeneous compared to those in the MoviLens dataset. Thus, the performances of the aggregation methods show a similar pattern for the CAMRa2011 dataset. Here, the difference between the highest achieved *nDCG* score and the lowest ranges from 1% to 8%.

Examining the effect of the number of recommended items, i.e., the length of the recommendation list (N), we find that the performances of the aggregation methods do not necessarily vary. Nevertheless, we observe a slight decrease (around %2–%3) in these performances as the number of the recommended items gets larger. This may be due to the sparsity of the datasets. The more items are recommended, the more we rely on the *estimated* actual ratings to calculate the *nDCG* scores.

When comparing the performances of the aggregation methods tested on the CAMRa2011 dataset, ECOagg and UL achieve the highest *nDCG* scores in almost all settings. Considering only these two methods, ECOagg is superior to UL when the number of recommended items is small. A baseline aggregation method AV follows these two methods. Also, the performance of IBGR, an advanced method, is comparable to AV.

As for the MovieLens dataset, where the groups are larger and random, we reach somewhat different conclusions. Again, the proposed aggregation method ECOagg scores the highest, especially for large groups; however its most serious competitor is TruGRC for this dataset. Also, AV shows a considerable performance, suggesting that selecting items that have received a high rating from the vast majority of a group may be an appropriate approach to determine the group preferences in certain circumstances.

Table 8
The *m-prop* results for the MovieLens dataset.

m	Aggregation method	Number of Groups (k)					
		4	8	16	32	64	128
1	AV	0.962	0.958	0.953	0.953	0.958	0.972
	AVG	0.824	0.791	0.827	0.888	0.921	0.951
	LM	0.763	0.781	0.831	0.877	0.916	0.951
	UL	0.751	0.776	0.843	0.885	0.933	0.961
	TruGRC	0.966	0.958	**0.976**	0.966	0.963	0.979
	IBGR	0.951	0.946	0.957	0.951	0.951	0.965
	ECOagg	**0.974**	**0.971**	0.964	0.966	0.969	0.971
2	AV	0.868	0.868	**0.871**	0.861	0.868	0.893
	AVG	0.671	0.591	0.619	0.691	0.781	0.841
	LM	0.549	0.574	0.626	0.704	0.773	0.847
	UL	0.551	0.571	0.636	0.724	0.809	0.873
	TruGRC	0.818	0.828	0.834	**0.885**	0.888	0.894
	IBGR	0.813	0.824	0.847	0.832	0.859	0.881
	ECOagg	**0.888**	**0.889**	0.861	0.861	0.888	0.892
3	AV	0.737	0.738	0.741	0.738	0.731	0.731
	AVG	0.451	0.395	0.414	0.487	0.577	0.675
	LM	0.415	0.381	0.429	0.493	0.574	0.675
	UL	0.386	0.387	0.447	0.512	0.614	0.729
	TruGRC	0.737	0.743	0.748	**0.759**	**0.758**	0.733
	IBGR	0.618	0.628	0.674	0.649	0.693	0.733
	ECOagg	**0.747**	0.744	0.746	0.745	0.738	0.731
4	AV	0.488	0.486	0.482	0.484	0.494	0.464
	AVG	0.277	0.233	0.259	0.276	0.347	0.456
	LM	0.271	0.228	0.257	0.282	0.351	0.448
	UL	0.234	0.241	0.285	0.298	0.399	0.501
	TruGRC	0.486	0.501	0.461	**0.501**	0.502	**0.512**
	IBGR	0.377	0.376	0.424	0.425	0.466	0.498
	ECOagg	**0.515**	0.505	**0.498**	0.487	0.488	0.491
5	AV	0.181	0.175	**0.189**	0.189	0.181	0.183
	AVG	0.148	0.108	0.108	0.111	0.149	0.219
	LM	0.122	0.107	0.112	0.111	0.141	0.211
	UL	0.119	0.106	0.126	0.118	0.174	0.244
	TruGRC	0.178	0.173	0.181	0.183	**0.194**	0.243
	IBGR	0.158	0.143	0.165	0.185	0.188	**0.274**
	ECOagg	**0.234**	**0.191**	0.182	0.184	0.181	0.225

Statistically significant results are given in bold

5.4.2. The m-prop results

As mentioned earlier, another indicator for evaluating the success of a GRS is its ability to maximize the number of group members satisfied with the recommended items. The metric called *m-prop* considers a member satisfied if she has provided a high rating (in our case a rating of 3.5 on a 5-star scale) to m out of N recommended items. The proportion of such members in the group yields the m-prop score. We fix the number of recommended items as 5 and let m vary from 1 to 5 in the experiments. Table 7 and Table 8 show the *m-prop* scores from the experiments conducted on the CAMRa2011 and Movielens datasets respectively.

Clearly it is somewhat trivial to achieve high *m-prop* scores when the groups are small and m is set to a small value; because if this is the case a group member is considered satisfied even if she is happy with a tiny fraction of the recommended items; and such members can easily dominate the small groups. For this reason, the aggregation methods show similar performance for the CAMRa2011 dataset when m is small. However, it becomes more difficult when m becomes large. In fact, the difference between the highest and lowest scores is about 10%, obtained by ECOagg and AV, respectively, when m is set to the maximum (i.e., 5).

Looking at the *m-prop* scores from the MovieLens dataset, where the groups are much larger and more heterogeneous, it is evident that the choice of aggregation method plays a more critical role, as the m scores in this case vary over a wide range. This suggests that in the presence of large and heterogeneous groups, one has to be more careful when choosing the aggregation method if one wants to satisfy the largest possible number of users. See the columns of Table 8 corresponding to small k values. For example, if m is three and k is four, and UL is employed for the aggregation process, then 38% of users are satisfied with three of the five recommended items, while this percentage is about 74% if ECOagg is preferred instead.

Based on the *m-prop* scores in Table 8, we claim that ECOagg's main competitor is TruGRC. ECOagg outperforms TruGRC when groups are larger, although TruGRC tends to achieve the highest *m-prop* score when the groups are relatively small. AV, a baseline method, follows bothof these sophisticated aggregation methods, in contrast its poor performance in CAMRa2011 dataset. This suggests that at the phase of the aggregation, featuring highly rated items only may be an appropriate approach to satisfy maximum number of group members in some circumstances. Nevertheless, this approach can be criticized for not recommending novel and interesting items to the groups, as it biased towards the popular items.

6. Conclusions

We divide this section into four parts to provide a structural framework for drawing conclusions about the successes and the limitations of the proposed aggregation method.

6.1. Theoretical contributions

In this paper, we propose a novel aggregation method for Group Recommender Systems (GRSs), called ECOagg, inspired by crowdsourcing, making the following theoretical contributions. First, we present the similarities between the task of aggregating labels coming from different crowd workers and that of aggregating preferences of group members to estimate group preferences. Second, we adapt the EM algorithm as used for aggregating labels in crowdsourcing to aggregate the members' preferences. Third, we consider the problem of estimating group rating as an ordinal classification problem, as there exists a natural ordering between the ratings; and fourth, we take a Bayesian approach imposing priors on the expertise levels of group members.

6.2. Practical contributions

The theoretical contributions summarized above lead to the

following practical contributions. Uncovering the similarities between crowdsourcing and GRSs promises to provide a new way to aggregate user preferences in GRSs using the established and justified methods of crowdsourcing. Also, thanks to the EM algorithm, which plays the key role in ECOagg, we improve the estimates of group preferences and the expertise levels of the group members concurrently.

Approaching the problem of estimating group rating from the perspective of ordinal classification opens up the possibility to borrow a wide range of established approaches available in ordinal classification. Following one of the most common approaches, we obtain $r-1$ binary classification problems for r ($r>2$) unique ratings. As a result, we are able to measure the expertise levels of the group members in terms of sensitivity and specificity. Finally, thanks to the Bayesian approach we take, we can incorporate any background information about the members, such as how reliable they are.

6.3. Limitations

We test the effectiveness of ECOagg on two datasets:CAMRa2011, where the groups are nuclear families, thus small and established; and MovieLens, where the groups are much larger and randomly constructed. Due to the page limitation, we restrict ourselves to these two types of groups, while there are two other types of groups in GRSs, namely occasional and automatically identified.

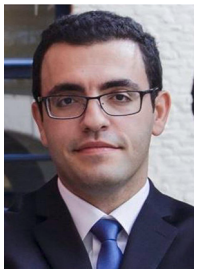
As emphasized before, ECOagg can incorporate any kind of background knowledge about group members due to its Bayesian nature. Nevertheless, to show the effectiveness of ECOagg without relying on external information, we only used the given datasets during the experiments in the this paper.

6.4. Future research

Future work will primarily focus on integrating some auxiliary information about users, such as trust networks, and members' contribution scores, into ECOagg, and try to cover the other types of groups in GRSs. Finally, as our method shows the similarities between crowdsourcing and GRSs, different aggregation methods can be adopted from crowdsourcing to be employed for GRSs in the future.

References

[1] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions, IEEE Trans Knowl. Data Eng. 17 (6) (2005) 734–749.
[2] C.C. Aggarwal, Recommender systems: The Textbook, vol. 1, Springer, 2016.
[3] R. Barzegar Nozari, H. Koohi, A novel group recommender system based on members' influence and leader impact, Knowl. Based Syst. 205 (2020) 106296.
[4] P. Bekkerman, S. Kraus, F. Ricci, Applying cooperative negotiation methodology to group recommendation problem, in: Proceedings of Workshop on Recommender Systems in 17th European Conference on Artificial Intelligence (ECAI 2006), 2006, pp. 72–75.
[5] L. Boratto, S. Carta, Modeling the preferences of a group of users detected by clustering: a group recommendation case-study, in: Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS14). WIMS'14. Association for Computing Machinery, New York, NY, USA, 2014.
[6] L. Boratto, S. Carta, G. Fenu, Investigating the role of the rating prediction task in granularity-based group recommender systems and big data scenarios, Inf. Sci. 378 (2017) 424–443.
[7] D.C. Brabham, Crowdsourcing, 1st ed., Mit Press, 2013.
[8] I.A. Christensen, S. Schiaffino, Entertainment recommender systems for group of users, Expert Syst. Appl. 38 (11) (2011) 14127–14135.
[9] S. Dara, C.R. Chowdary, C. Kumar, A survey on group recommender systems, J. Intell. Inf. Syst. 54 (2) (2020) 271–295.
[10] A.P. Dawid, A.M. Skene, Maximum likelihood estimation of observer error-rates using the EM algorithm, J. R. Stat. Soc. Ser. C: Appl. Stat.) 28 (1) (1979) 20–28.
[11] T. De Pessemier, J. Dhondt, K. Vanhecke, L. Martens, Travelwithfriends: a hybrid group recommender system for travel destinations, in: Workshop on Tourism Recommender Systems (touRS15), in Conjunction With the 9th ACM Conference on Recommender Systems (recsys 2015)., 2015, pp. 51–60.
[12] A. Delic, J. Neidhardt, T.N. Nguyen, F. Ricci, Research methods for group recommender system, in: RecTour@ RecSys, 2016, pp. 30–37.
[13] E. Estellés-Arolas, F. González-Ladrón-de Guevara, Towards an integrated crowdsourcing definition, J. Inf. Sci. 38 (2) (2012) 189–200.
[14] A. Felfernig, L. Boratto, M. Stettinger, M. Tkalčič, Evaluating Group Recommender Systems, Springer International Publishing, Cham, 2018, pp. 59–71.
[15] E. Frank, M. Hall, A simple approach to ordinal classification, in: European Conference on Machine Learning, Springer, 2001, pp. 145–156.
[16] H. Gimpel, V. Graf-Drasch, R.J. Laubacher, M. Wöhl, Facilitating like Darwin: supporting cross-fertilisation in crowdsourcing, Decis. Supp. Syst. 132 (2020) 113282.
[17] J. Guo, Y. Zhu, A. Li, Q. Wang, W. Han, A social influence approach for group user modeling in group recommendation systems, IEEE Intell. Syst. 31 (5) (2016) 40–48.
[18] M. Hossain, I. Kauranen, Crowdsourcing: a comprehensive literature review, Strateg. Outsourcing Int. J. (2015).
[19] N.Q.V. Hung, N.T. Tam, L.N. Tran, K. Aberer, An evaluation of aggregation techniques in crowdsourcing, in: International Conference on Web Information Systems Engineering (WISE), Springer, 2013, pp. 1–15.
[20] Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model, in: Proceedings of the 14th ACM SIGKDD International conference on KDD, 2008, pp. 426–434.
[21] M. Lease, O. Alonso, Crowdsourcing and human computation, introduction. Encyclopedia of Social Network Analysis and Mining, 2014, pp. 305–315.
[22] Y. Li, R. Wang, G. Nan, D. Li, M. Li, A personalized paper recommendation method considering diverse user preferences, Decis. Supp. Syst. 146 (2021) 113546.
[23] J. Masthoff, Group Recommender Systems: Aggregation, Satisfaction and Group Attributes, Springer US, Boston, MA, 2015, pp. 743–776.
[24] J.E. McCarthy, T.D. Anagnost, Musicfx: an arbiter of group preferences for computer supported collaborative workouts, in: Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work. CSCW'00. Association for Computing Machinery, New York, NY, USA, 2000, p. 348.
[25] F.R.A. Neto, C.A. Santos, Understanding crowdsourcing projects: a systematic review of tendencies, workflow, and quality management, Inf. Process. Manage. 54 (4) (2018) 490–506.
[26] J. Park, K. Nam, Group recommender system for store product placement, Data Mining Knowl. Discov. 33 (1) (2019) 204–229.
[27] M.S. Pera, Y.-K. Ng, A group recommender for movies based on content similarity and popularity, Inf. Process. Manage. 49 (3) (2013) 673–687.
[28] V.C. Raykar, S. Yu, L.H. Zhao, G.H. Valadez, C. Florin, L. Bogoni, L. Moy, Learning from crowds, J. Mach. Learn. Res. 11 (4) (2010).
[29] A. Said, S. Berkovsky, E.W. De Luca, J. Hermanns, Challenge on context-aware movie recommendation: Camra2011, in: Proceedings of the Fifth ACM Conference on Recommender Systems, 2011, pp. 385–386.
[30] C. Senot, D. Kostadinov, M. Bouzid, J. Picault, A. Aghasaryan, C. Bernier, Analysis of strategies for building group profiles, in: International Conference on User Modeling, Adaptation, and Personalization, Springer, 2010, pp. 40–51.
[31] Y.-D. Seo, Y.-G. Kim, E. Lee, H. Kim, Group recommender system based on genre preference focusing on reducing the clustering cost, Expert Syst. Appl. (2021) 115396.
[32] Y.-D. Seo, Y.-G. Kim, E. Lee, K.-S. Seol, D.-K. Baik, An enhanced aggregation method considering deviations for a group recommendation, Expert Syst. Appl. 93 (2018) 299–312.
[33] D. Serbos, S. Qi, N. Mamoulis, E. Pitoura, P. Tsaparas, Fairness in package-to-group recommendations, in: Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 371–379.
[34] W. Wang, G. Zhang, J. Lu, Member contribution-based group recommender system, Decis. Supp. Syst. 87 (2016) 80–93.
[35] X. Wang, Y. Liu, J. Lu, F. Xiong, G. Zhang, TRUGRC: trust-aware group recommendation with virtual coordinators, Future Gener. Comput. Syst. 94 (2019) 224–236.
[36] E. Yalcin, A. Bilge, A personality-based aggregation technique for group recommendation, Eskişehir Tech. Univ. J. Sci. Technol. A: Appl. Sci. Eng. 21 (4) (2020) 486–498.
[37] E. Yalcin, A. Bilge, Novel automatic group identification approaches for group recommendation, Expert Syst. Appl. 174 (2021) 114709.
[38] E. Yalcin, F. Ismailoglu, A. Bilge, An entropy empowered hybridized aggregation technique for group recommender systems, Expert Syst. Appl. 166 (2021) 114111.
[39] M.-C. Yuen, I. King, K.-S. Leung, A survey of crowdsourcing systems, in: 2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing, IEEE, 2011, pp. 766–773.
[40] A. Zawali, I. Boukhris, A group recommender system for academic venue personalization, in: International Conference on Intelligent Systems Design and Applications, Springer, 2018, pp. 597–606.



Firat Ismailoglu was born in Mersin, Turkey. He graduated from Selcuk University, Turkey, with a BSc in Mathematics in 2007. He obtained an MSc in Knowledge Discovery and Data Mining from the University of East Anglia, UK, in 2010. In 2016, he completed his PhD in machine learning at the Department of Data Science and Knowledge Engineering, Maastricht University, the Netherlands. After that, he did a postdoc at the Department of Mathematics and Computer Science at the Eindhoven University of Technology, the Netherlands. As of September 2017, he has been working as an Assistant Professor at the Department of Computer Engineering, Sivas Cumhuriyet University, Turkey.