The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

SAGA: A Submodular Greedy Algorithm for Group Recommendation

Shameem A Puthiya Parambath
QCRI, HBKU, Doha, Qatar
spparambath@hbku.edu.qa

Nishant Vijayakumar
Apptopia Inc., Boston, USA
nishant.vijayakumar@gmail.com

Sanjay Chawla
QCRI, HBKU, Doha, Qatar
schawla@hbku.edu.qa

Abstract

In this paper, we propose a unified framework and an algorithm for the problem of group recommendation where a fixed number of items or alternatives can be recommended to a group of users. The problem of group recommendation arises naturally in many real world contexts, and is closely related to the budgeted social choice problem studied in economics. We frame the group recommendation problem as choosing a subgraph with the largest group consensus score in a completely connected graph defined over the item affinity matrix. We propose a fast greedy algorithm with strong theoretical guarantees, and show that the proposed algorithm compares favorably to the state-of-the-art group recommendation algorithms according to commonly used relevance and coverage performance measures on benchmark dataset.

1 Introduction

Recommender systems can be regarded as a particular instance of a personalized information retrieval system where the explicitly or implicitly learned user profiles/features serve the purpose of a 'query' in the traditional IR system. Recommender systems provide effective information exploration by proposing a set of relevant items/alternatives to users. Many efficient and scalable algorithms exist for personalized recommender systems, carefully tailored for many desirable properties of the recommendation set like relevance, diversity, coverage etc (Ekstrand, Riedl, and Konstan 2011; Steck 2011; Puthiya Parambath, Usunier, and Grandvalet 2016). In contrast to personalized recommender systems, a group recommender (GR) system has to provide recommendations for a set of users jointly, returning a set of items in consensus with diverse user preferences. The user interests might be often conflicting in nature. A number of real world applications can be formulated within the framework of choosing a subset of items from a given item set to appeal to a group of users. For example, group discount vendors have to choose a set of items that best appeals to the set of diverse users in a given city; similarly flight operators have to choose a set of movies to play in a plane's entertainment system. The problem is also closely related to the budgeted social choice problem studied in economics (Lu and Boutilier 2011; Skowron, Faliszewski, and Jerome Lang 2015).

In recommender systems, items are assumed to have intrinsic preference scores for each user, which represent the level of affinity of the user to the item. We assume the preference scores to be on an ordinal scale, typically in the range of 1-5, with higher values indicating stronger preference. A simple and straightforward approach to group recommendation is to aggregate the preference scores i.e combine the preference scores of individual group members for an item to form a joint ranking of the items and select the top-**k** items (Masthoff 2011; Baltrunas, Makcinskas, and Ricci 2010). Most commonly used aggregation functions are *(i) average misery*; recommending items with highest average preference scores among the group members, *(ii) least misery*; recommending items that maximize the minimum preference scores among users and *(iii) plurarility*; recommending items that maximize the number of "highest preference scores" (Masthoff 2011; Amer-Yahia et al. 2009). But such score aggregation strategies completely disregard the user disagreement or the individual user satisfaction for individual items as discussed in Amer-Yahia et al. (2009). To get a more balanced recommendation, it is important to consider the disagreements between users of the group for the same item. Thus the problem can be reformulated as a bi-objective optimization problem where one objective corresponds to the agreement (relevance) of the item to the users and the other objective corresponds to the disagreement between the users for the item. The general procedure to solve a multi-objective optimization problem is to take the convex combination of the objectives and perform homotopy continuation, also referred as scalarization or weighted-sum approach (Ehrgott and Gandibleux 2002). Similar to aggregation strategies, scalarization based algorithms assume that preference scores for the test items are available, but in practice one has to run a baseline recommender system as a first step to generate the preference scores.

Inspired by the recent work on personalized diverse recommendations (Puthiya Parambath, Usunier, and Grandvalet 2016), in this paper we propose a general framework for algorithms for group recommendations. We propose a generic objective function by framing the problem as a relevant group interest coverage in an affinity graph defined over the item features. The crux of our approach is that the

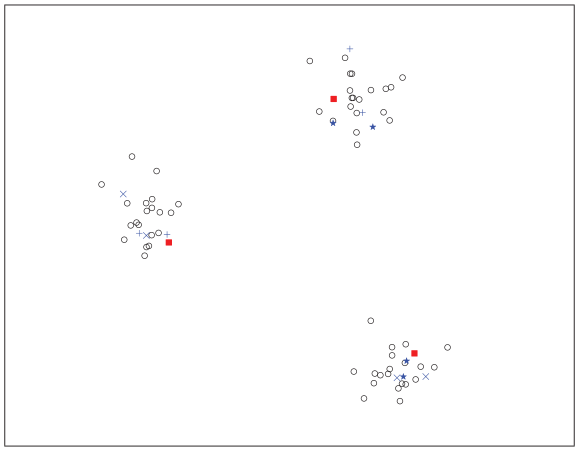

Figure 1: Graphical intuition of the proposed approach

item-item and user-user interactions are captured in a single objective. Unlike score aggregation strategies, we do not require the preference scores of the test items to be generated beforehand. We give two instances of the objective where the GR problem reduces to a submodular function maximization problem.The 'diminishing return' property of submodular functions allows us to trade-off the redundancy and relevance, thus making the recommended set containing relevant items covering a wider spectrum of group interest. An intuitive graphical representation of the underlining idea behind our approach is given in Figure 1. The figure contains preferences of a group of three users spanning three item classes, users being represented by the blue '+','*' and '$\times$' symbols. None of the three users have common item classes, but the group covers all the item classes. Hence the optimal recommendation strategy is to recommend relevant items from the three item clusters. The red square is the recommendation made by our proposed algorithm, whereas the baseline algorithms recommendations covered only two item clusters (See Section 4 for details).

The remainder of this paper is structured as follows. We give a brief overview of group recommendation algorithms in Section 2. Section 3 describes our framework and relationship to other group recommendation strategies . We carry out large scale experiments on benchmark data and report the results in Section 4. We conclude the paper in Section 5.

2 Related Work

Most of the earlier work on group recommendation is based on a model where the individual group member attributes are aggregated to generate the group recommendation (Masthoff 2011). Based on this model, state of the art algorithms can be further classified as profile aggregation based and score aggregation based. In the first approach, individual user profiles are aggregated to create a group profile. The group profile can be seen as a single artificial user capturing the group dynamics, and state of the art personalized recommendation algorithm can be used to produce group recommendations for this artificial user. In the second approach also, the group recommendation is produced in two steps. In the first step, preference scores for each individual group member for the test items are generated by running a collaborative filtering algorithm. The second step involves running a score aggregation algorithm on the preference scores obtained in the first step to generate the group recommendations. In one of the earliest work on group recommendation, O'Connor et al. (2001) proposed an algorithm where the individual user ratings are generated using nearest neighbor models for collaborative filtering, and employing score aggregation strategies like *least misery* and *average misery* to obtain group recommendations. Similarly, in (Kim et al. 2010), the authors proposed an algorithm by aggregating the individual user profiles to create a group profile and using neighborhood models to generate recommendations for the newly created group profile. This algorithm comes under the first approach explained above. Masthoff (2011) gives an overview and use cases of commonly employed scoring strategies in recommender systems. However, both the *score aggregation* and *profile aggregation* approaches fail to attribute for the individual user disagreements for different items (Amer-Yahia et al. 2009).

Amer-Yahia et al. (2009) advocated that a better group recommendation strategy can be devised by considering the disagreements between the individual users in the group for the same item. They proposed an algorithm where the recommendation set is obtained by solving a discrete optimization problem. The objective function is a linear combination of an agreement term and a disagreement term. It is an instantiation of the scalarization algorithm for the multi-objective optimization problem. The proposed algorithm requires one to run a baseline personalized recommender system to generate preference scores for every user. Moreover, many variants of the above discussed scalarization based group recommendation schemes are proposed in domain specific settings and we direct the reader to the survey papers by Jameson and Smyth (2007) and Boratto and Carta (2010), and the references therein. One of the difficulties in developing group recommendation algorithms is the lack of proper evaluation strategies. In Baltrunas, Makcinskas, and Ricci (2010) , the authors propose a methodology for offline evaluation of group recommendation algorithms. Recent works make use of expensive online evaluation strategies (Quijano-Sánchez, Díaz-Agudo, and Recio-García 2014; Boratto, Carta, and Fenu 2016; Liu et al. 2016; Khoshkangini, Pini, and Rossi 2016)

In economics, group recommendation is studied as a special case of the budgeted social choice problem (Lu and Boutilier 2011; Skowron, Faliszewski, and Jerome Lang 2015). The budgeted consensus recommendation is often considered as a trade-off between pure personalized recommendation and pure consensus recommendation. In group recommendation, the budget is specified as a function of the recommendation size, keeping the other cost components constant. Most of the work on budgeted social choice theory assumes that user preferences for the items are strict, i.e. every user derives benefit from at most one item that has the highest preference score (utility score), and preference scores are often derived using positional scoring rules like Borda count. Lu and Boutilier (2011) proposed a greedy algorithm using knapsack heuristics, but it worked very poorly on group recommendation tasks with positional scoring rules. Skowron, Faliszewski, and Jerome Lang (2015) extended the theoretical framework for limited choice models with positional scoring rules using the ordered weighted average (OWA) operators.

Unlike the above mentioned approaches, our approach proceeds in a single step by carrying out preference estimation implicitly and thus avoids running a baseline personalized recommender system to generate the *unseen* test preference scores. The trade-off between the agreement and the disagreement components is dealt with by the exact definition of the *user saturation function*, *item saturation function* and the affinity functions, instead of explicitly trading-off of both the terms. Our work is inspired by Puthiya Parambath, Usunier, and Grandvalet (2016) where the authors propose a personalized diverse recommendation algorithm. Nonetheless, our work differs from theirs significantly. We introduce group consensus score as a function of the *user saturation function* and the *item saturation function*. The exact notion of *user saturation function* is motivated by the *item saturation function* defined in Puthiya Parambath, Usunier, and Grandvalet (2016), to capture the coverage of an item observed by a user. Our proposed algorithm is more general and the algorithm proposed in Puthiya Parambath, Usunier, and Grandvalet (2016) is a special case of our approach in that when the group consists of a single user, the *item saturation function* is a concave function and the *user saturation function* is the identity function. By defining the *user coverage function* to be submodular, we aim to *(i)* recommend items which are relevant to the majority of the group and *(ii)* discourage the recommendations to be centered around a few user.

3 Group Recommendation

We are given a set $\mathcal{X}$ of n items and a set $\mathcal{U}$ of m users. The set of users forms a set of groups $\mathbb{G}$. The group dynamics can evolve over time, and a user may or may not be a member of a group. Also, a user can be part of just a single group or multiple groups. We also assume the existence of an affinity function over the item space $h : \mathcal{X} \times \mathcal{X} \to \Re_+$. Similarly, we assume the existence of an affinity function over the user space $c : \mathcal{U} \times \mathcal{U} \to \Re_+$. We assume that the affinity functions h and c are non-negative. There is no assumption regarding the symmetricity or transitivity of the affinity functions. Each user has a preference score (utility score or relevance score) for each of the item denoted using an ordinal number, typically in the range 1-5 with higher value indicating stronger preference. A user can have same preference score for different items.

Given the item set $\mathcal{X}$ and the corresponding item-item affinity matrix $\mathbf{W}$, we can view the pair $(\mathcal{X}, \mathbf{W})$ as a complete graph where the edge between nodes $i, j \in \mathcal{X}$ is weighted according to the value $\mathbf{W}_{ij}$. Given a group $\mathcal{G} \in \mathbb{G}$, the user-user affinity matrix $\mathbf{A}$ and the observed preference scores for the set of users in the group $\mathcal{G}$ for the item set $\mathcal{I} \subset \mathcal{X}$, the item set $\mathcal{I}$ defines a subgraph of the complete graph $(\mathcal{X}, \mathbf{W})$. We can define the group recommendation task as retrieving a subset of $\mathbf{k}$ items from the set $\mathcal{Z} = \mathcal{X} \setminus \mathcal{I}$ such that it *(i) covers* most of the observed items and has high preference scores in the set $\mathcal{I}$ and *(ii) covers* a larger spectrum of users in the group $\mathcal{G}$. We introduce group consensus score to capture the above two aspects of the group recommendation. Formally, we define the group consensus score for a set of items $\mathcal{S} \subseteq \mathcal{Z}$ with cardinality $\mathbf{k}$ with respect to the set $\mathcal{I}$ for the given group $\mathcal{G}$ as,

$$\mathbf{Gscore}(\mathcal{G}, \mathcal{I}, \mathcal{S}) = \sum_{u \in \mathcal{G}} g_u\Big(\sum_{l \in \mathcal{G} \setminus \{u\}} \mathbf{A}_{ul} \sum_{i \in \mathcal{I}_u} r_u^i f\big(\sum_{j \in \mathcal{S}} \mathbf{W}_{ij} \big)\Big) \quad (1)$$

where $g_u : \Re_+ \to \Re$ is the *user saturation function*, $f : \Re_+ \to \Re_+$ is the non-negative *item saturation function*, $\mathcal{I}_u$ is the set of observed items in $\mathcal{I}$ for the user u, and r_u^i is the observed preference score for the item i by user u. The rationale behind the objective function is to find the items similar to the observed relevant items which appeals for the entire group, as demonstrated by higher value for $\mathbf{Gscore}$. By choosing proper *item saturation* and *user saturation* functions, we aim to produce a consensus set of recommendations. Given a fixed *user saturation function* g_u and *item saturation function* f, a set of $\mathbf{k}$ items for the group $\mathcal{G}$ with the highest group consensus score can be obtained by solving the optimization problem

$$\underset{\substack{\mathcal{S} \subseteq \mathcal{Z} \\ |\mathcal{S}| \leq \mathbf{k}}}{\text{argmax}}\, \mathbf{Gscore}(\mathcal{G}, \mathcal{I}, \mathcal{S}) \ . \quad (2)$$

Before further discussion into the design details of the *saturation functions* g_u and f, we describe the generality of the objective function (2). It can be shown that many well-known group recommendation strategies are special cases of the objective function (2).

Special Cases

For different formulations of the *user saturation function* g_u, the optimization problem in (2) corresponds to *score aggregation* strategies currently employed in group recommendation tasks.

Identity item saturation function Here, we see that by fixing the *item saturation function* (f) to be the identity function i.e. $f(x) = x$ and varying the *user saturation function* g_u gives rise to many score aggregation strategies. We will also assume that $\mathbf{A}$ is the indicator matrix for the given group $\mathcal{G}$ i.e. $\mathbf{A}_{ij} = 1$ if i and j are in group $\mathcal{G}$, zero otherwise.

Least Misery: If g_u is set as the constant function $\min$ i.e. $g_u(x) = \min(x), \forall u \in \mathcal{G}$, then the optimization problem corresponds to selecting $\mathbf{k}$ items which maximize the lowest preference scores on the test set, among all members in the group. This is equivalent to the *least misery* aggregation strategy (Masthoff 2011).

Most Pleasure: Similarly, by setting g_u to the $\max$ function i.e. $g_u(x) = \max(x), \forall u \in \mathcal{G}$, the optimization problem in (2) becomes the *most pleasure* scoring strategy (Masthoff 2011). $\mathbf{k}$ items that maximize the highest preference scores for the items in the test set, among all members in the group are recommended.

Average Misery: If g_u is set as the constant function $g_u(x) = \frac{1}{|\mathcal{G}|}, \forall \mathcal{G} \in \mathbb{G}$, then the optimization problem reduces to the average misery approach, where the items that maximize the average preference scores among all the members in the group are recommended (Masthoff 2011).

Plurality: If g_u is the indicator function such that it takes the value one when the preference score is the highest (ties are broken arbitrary) and zero otherwise, then the optimization problem corresponds to the plurality scheme (Masthoff 2011).

Ordered Weighed Average Operators: If the inner term $\sum_{i \in \mathcal{I}_u} r_u^i f\big(\sum_{j \in \mathcal{S}} \mathbf{W}_{ij}\big)$ is assumed to be in the sorted order, and for a general $g_u : \Re_+ \to \Re_+$, the group consensus function corresponds exactly to the ordered weighted average (OWA) operators studied by Skowron, Faliszewski, and Jerome Lang (2015). In this case, many different functional formulations of g_u will result in different group recommendation strategies like $\mathbf{k}$-best OWA, Hurwicz OWA etc (See Skowron, Faliszewski, and Jerome Lang (2015) for details).

Identity user saturation function Here, we fix the *user saturation function* (g_u) to be the identity function i.e. $g_u(x) = x$ and vary the *item saturation function* f.

Diverse Personalized Recommendation: In case of single user groups ($\mathbf{A}$ will be the identity matrix), when the *item saturation function* is concave, the objective function reduces to the one studied by Puthiya Parambath, Usunier, and Grandvalet (2016) for diverse personalized recommendations. For other functional forms of f, we get different personalized recommendation schemes (Puthiya Parambath, Usunier, and Grandvalet 2016).

Time Evolving Group Dynamics: The individual user preferences are dynamic and it can evolve over time. User's interest for activities like shopping, watching movies etc, can fade or strengthen over time. It is important to adjust the parameters of the algorithm to reflect the evolving group dynamics by lowering the significance of the "fading" user and highering the significance of the "strengthening" user. One way to achieve this is by re-weighting the individual user contribution in a group when recommending new items. This can be done by defining the *user saturation function* g_u as $g_u(x) = \frac{x}{t_u}$, where t_u is the "*transportation time*" of user u in the given group. One can define the "*transportation time*" as a function of user's activity time in the group where a frequent loyal user will have low *transportation time* whereas occasional user will have high *transportation time*.

Design of Saturation Function

Ideally, we want the recommended set to include relevant yet items that appeals to a larger spectrum of users in the group. We design *saturation functions* such that the $\mathbf{Gscore}$ has the "diminishing return" property. In our settings, the "diminishing return" property ensures that once a relevant item with respect to one (or few) user(s) is added, the marginal gain of adding another relevant item for the same user(s) is less than adding a relevant item with respect to the remaining users. In particular, the "diminishing return" property of the *item saturation function* helps to select relevant yet diverse items, whereas the "diminishing return" property of the *user saturation function* encourages user coverage. The "diminishing return" is the characterizing property of submodular functions, and we choose the *saturation functions* to be such that group consensus score is a submodular function.

We discuss two settings. In the first settings, we propose a simple extension of the algorithm proposed in Puthiya Parambath, Usunier, and Grandvalet (2016) by choosing g_u to be the identity function.

Lemma 1. *For any concave function f and the identity function g_u, $\mathbf{Gscore}$ is a submodular function (with respect to inclusion in $\mathcal{S}$). If f is monotonic then $\mathbf{Gscore}$ is also monotonic.*

The proof of Lemma 1 can be found in Bach (2013). In the second setting, we choose g_u to be a concave function.

Lemma 2. *For any monotone concave function f and any concave funtion g_u, $\mathbf{Gscore}$ is a submodular function (with respect to inclusion in $\mathcal{S}$). If g_u is monotonic then $Gscore$ is a monotonic submodular function.*

Proof. By Lemma1, $\sum_{l \in \mathcal{G} \setminus \{u\}} \mathbf{A}_{ul} \sum_{i \in \mathcal{I}_u} r_u^i f\big(\sum_{j \in \mathcal{S}} \mathbf{W}_{ij}\big)$ is a submodular function of $\mathcal{S}$. Let us denote this term as $z(\mathcal{S})$. It is sufficient to prove that $g(z(\mathcal{S}))$ (we omit the subscript for convenience) is a submodular function. Due to the concavity of g and monotonicity of z (due to the monotonicity of f), $\forall \mathcal{S}_1, \mathcal{S}_2$ such that $\mathcal{S}_1 \subseteq \mathcal{S}_2 \subseteq \mathcal{Z}$ and $\forall e \in \mathcal{Z}$ such that $e \notin \mathcal{S}_2$,

$$g(z(\mathcal{S}_1 \cup e)) - g(z(\mathcal{S}_1)) \geq g(z(\mathcal{S}_2 \cup e)) - g(z(\mathcal{S}_2))$$

which completes the proof. □

Accelerated Greedy Algorithm

For any monotone concave function f and any concave funtion g_u, the optimization problem (2) becomes a submodular maximization problem. The problem is NP-hard and no algoithms exist to solve it exactly (Nemhauser, Wolsey, and Fisher 1978). The general strategy to solve the submodular maximization problem is based on the greedy heuristic as given by Nemhauser, Wolsey, and Fisher (1978). The algorithm iteratively selects an element from the ground set such that it gives the maximum value for the incremental update value $\mathbf{Gscore}(\mathcal{G}, \mathcal{I}, \mathcal{S} \cup \{e\}) - \mathbf{Gscore}(\mathcal{G}, \mathcal{I}, \mathcal{S})$ at each iteration (the ties are broken arbitrarily). In case of monotonic submodular functions with cardinality constraints, Nemhauser, Wolsey, and Fisher (1978) gives a worst case lower bound on the optimality gap between the optimal solution and the greedy solution as given in Theorem 3. In fact, similar bound holds for any monotonic submodular functions with matroid constraints and cardinality is a special case of matroid constraint (Fisher, Nemhauser, and Wolsey 1978).

Algorithm 1: SAGA: Submodular Greedy Group Recommendation Algorithm

```
Input : set of items $\mathcal{X}$, set of observed items $\mathcal{I}$, group
        $\mathcal{G}$, affinity matrix $\mathbf{W}$, # of recommendations $\mathbf{k}$
1  $\mathcal{X} = \mathcal{X} \setminus \mathcal{I}, \mathcal{S} = \emptyset$ ;
2  for $i \in \mathcal{X}$ do
3      $\Delta(i) = Gscore(\mathcal{G}, \mathcal{I}, \{i\})$ ;  // compute and
       store the marginal gain for each
       item in a priority queue
4  repeat
5      $i^* = \text{argmax}_{i \in \mathcal{X}} \Delta(i)$ ;
6      $\delta = Gscore(\mathcal{G}, \mathcal{I}, \mathcal{S} \cup \{i^*\}) - Gscore(\mathcal{G}, \mathcal{I}, \mathcal{S})$ ;
7      $\Delta(i^*) = \delta$ ;
8      if $\delta < \max_{i \in \mathcal{X} \setminus \{i^*\}} \Delta(i)$ then
9          goto 5
10     $\mathcal{S} = \mathcal{S} \cup \{i^*\}$ ;
11     $\mathcal{X} = \mathcal{X} \setminus \{i^*\}$ ;
12 until $|\mathcal{S}| = \mathbf{k}$;
   Output: set of recommendations $\mathcal{S}$
```

Minoux (1978) proposed a faster "accelerated" version of the greedy algorithm using the fact that $g(\mathcal{S}_j \cup \{l\}) - g(\mathcal{S}_j) \leq g(\mathcal{S}_i \cup \{k\}) - g(\mathcal{S}_i) \implies g(\mathcal{S}_i \cup \{l\}) \leq g(\mathcal{S}_i \cup \{k\}), \forall j < i$. The algorithm carries out 'lazy' updates thus avoiding a complete scan to find the next item to recommend (Leskovec et al. 2007). By using priority queues, the retrieval at line 5 and updation at line 10 in Algorithm1 can be achieved in constant and logarithmic time respectively. Previous experimental results on large scale datasets showed that the accelerated greedy algorithm gives a substantial speedup boost (Leskovec et al. 2007). Our proposed accelerated greedy algorithm for group recommendation which we call **S**ubmodul**A**r **G**roup recommendation **A**lgorithm (**SAGA**) is given in Algorithm 1.

Theorem 3 (Nemhauser, Wolsey, and Fisher (1978))**.** *For a non-decreasing submodular function f, let $\mathcal{S}^*$ be the optimizer of* (2) *and $\hat{\mathcal{S}}$ be the set returned by the greedy algorithm iteratively adding to the current solution the item that provides the highest gain in the objective function, then* $f(\hat{\mathcal{S}}) \geq \left(1 - (1 - \frac{1}{\mathbf{k}})^{\mathbf{k}}\right) f(\mathcal{S}^*) \geq (1 - \frac{1}{e})\, f(\mathcal{S}^*)$

4 Experiments

In this section, we describe the experimental setup and report the results using user groups generated from the *1M* MovieLens[1] dataset. A major issue with GR research is the difficulty of evaluating the effectiveness of the recommendations (Baltrunas, Makcinskas, and Ricci 2010). Online evaluation is expensive and it can be performed only on a very limited set of users. Here we follow the offline experimental protocol proposed in Baltrunas, Makcinskas, and Ricci (2010). We compare our proposed algorithm against two state-of-the-art group recommendation algorithms (O'Connor et al. 2001; Amer-Yahia et al. 2009).

[1]http://grouplens.org/datasets/movielens/

# members	2	4	6	8
random	294	146	98	72
similar	190	40	18	10

Table 1: Number of Groups

Group Formation

The MovieLens dataset contains ratings on a scale of 1 to 5 of 6040 users for 3670 movies. In our experiments we use users only with a minimum of 100 ratings and ended up with a set of 2945 users and 3670 movies. Following the previous work by Amer-Yahia et al. and Baltrunas, Makcinskas, and Ricci, we create groups corresponding to two common real life scenarios *(i)* random groups and *(ii)* similar groups. Random groups are generated by randomly sampling a fixed number of users whereas similar groups are generated by randomly sampling from the user set provided that the individual user-user similarity is greater than a threshold value. We set 0.60 as the similarity threshold value. The detailed description of the group statistics is given in Table1.

Protocol

We carried out holdout validation by randomly selecting 30% of the item set and marking it as unrated wherever the rating values are observed. To reduce the variability in the result, the procedure is carried out five times and the reported results are the average values over the five selections. The affinity and similarity values are calculated based on the item and user features. The item and user features are extracted from the rating matrix using the non-negative matrix factorization based on weighted-regularized non-negative alternating least squares algorithm (Steck 2013). In our experiments, the dimension of the item/user feature space is set to 150. We used the rbf kernel as the item afinity function h, i.e. $\mathbf{W}_{ij} = \exp(-\gamma||x_i - x_j||^2)$ where x_i and x_j are the i^{th} and j^{th} item features. The γ value is chosen by running the algorithm for a set of values in the range $\{2^{-3}, \cdots, 2^3\}$ in multiples of two, and the reported results are for the best γ value for the respective algorithms. For the *item saturation function* f, we use natural logarithm $f(x) = \ln(1 + x)$, and for the *user saturation function* we experimented with two settings *(i)* identity function $g_u(x) = x$ and *(ii)* concave function $g_u(x) = \sqrt{x}$. The user affinity matrix $\mathbf{A}$ is defined as the cosine of the user feature vector i.e. $\mathbf{A}_{ij} = \cos(y_i, y_j)$ where y_i and y_j are the i^{th} and j^{th} user features.

Baselines

We used two baselines to compare against the performance of our proposed algorithm. First baseline is the popular aggregation strategy *Average Misery* (AM) and the second baseline is the scalarization based algorithm proposed by Amer-Yahia et al. (2009) called group recommendation with fully materialized disagreement list (FM). Both AM and FM require unobserved rating values to be generated beforehand, and we use the non-negative matrix factorization (used for the user/item feature extraction) to generate the unobserved rating values.

AM *Average Misery* is a popular *score aggregation* strategy where the first **k** items with the highest average score for the group is selected. Formally, *Average Misery* algorithm selects an item i^* such that

$$i^* \in \operatorname*{argmax}_{i \in \mathcal{Z}} rel(\mathcal{G}, i) \ . \tag{3}$$

where $rel(\mathcal{G}, i)$ is defined as the sum of the preference scores for the item by the users in the group $\mathcal{G}$, i.e. $rel(\mathcal{G}, i) = \sum_{u \in \mathcal{G}} r_u^i$.

FM In FM, an agreement term and a disagreement term are linearly combined using a trade-off parameter, and for each value of the trade-off parameter the discrete optimization problem is solved using thresholding (Amer-Yahia et al. 2009). Formally, the FM algorithm selects an item i^* such that

$$i^* \in \operatorname*{argmax}_{i \in \mathcal{Z}} \ \lambda \times rel(\mathcal{G}, i) + (1 - \lambda)(1 - dis(\mathcal{G}, i))$$

where $rel(\mathcal{G}, i)$ is as defined above, and $dis(\mathcal{G}, i)$ is the average pair-wise disagreement for the item i for the group $\mathcal{G}$. Formally,

$$dis(\mathcal{G}, i) = \frac{2}{|\mathcal{G}|(|\mathcal{G}| - 1)} \sum_{\substack{u,v \in \mathcal{G} \\ u \neq v}} |r_u^i - r_v^i|$$

Performance Measure

In our experiments, we measure the relevance and coverage of the recommended items to the group members. We use Discounted Cumulative Gain (DCG), a very popular performance measure widely used in information retrieval tasks to measure the utility of the recommended items to a user (Järvelin and Kekäläinen 2002), and Popularity Stratified Recall (PSR), which measures the coverage/popularity-bias of the recommended items (Steck 2011).

Discounted Cumulative Gain (DCG): DCG is used in the context of ranking. It measures the quality of a ranked list by the sum of the graded relevance discounted by the rank of the item. In our experiments, we used the below formulation:

$$\sum_{i \in \mathcal{S}} \frac{2^{r_i} - 1}{log(i + 1)} \ ,$$

where r_i is the *graded* relevance score of the i^{th} item. In our settings, the i^{th} item is the i^{th} item entering $\mathcal{S}$ for the greedy algorithm. We compute the average values of the DCG, over all the users in any group, and the reported results are average over the total groups. Higher values for DCG is desired, as it indicates more relevant recommendations appears at the top of the recommendation list.

Popularity Stratified Recall (PSR): A good recommendation strategy should cover items which are relevant to the users but rare. PSR is proposed to measure the ability of a recommender system to recommend items from the tail of the item-popularity distribution. The submodular formulation of our objective increases the chance of recommending unpopular relevant items. Formally, PSR is defined as:

$$\frac{1}{|\mathcal{G}|} \frac{\sum_{u \in \mathcal{G}} \sum_{i \in \mathcal{S}_u^+} \left(\frac{1}{N_i^+}\right)^\beta}{\sum_{u \in \mathcal{G}} \sum_{i \in \mathcal{T}_u} \left(\frac{1}{N_i^+}\right)^\beta} \ ,$$

where $\mathcal{S}_u^+$ is the set of recommended items that are known to be relevant for user u (among the **k** recommended items), $\mathcal{T}_u$ is the set of items in the test set that are known to be truly relevant for user u, N_i^+ is the number of relevant ratings for item i in the test set and $\beta \in (0, 1)$ is a hyperparameter which adjusts for the popularity bias. We used $\beta = 0.5$ in our experiments. Higher values for PSR indicates more relevant unpopular recommendations appears at the top of the recommendation list.

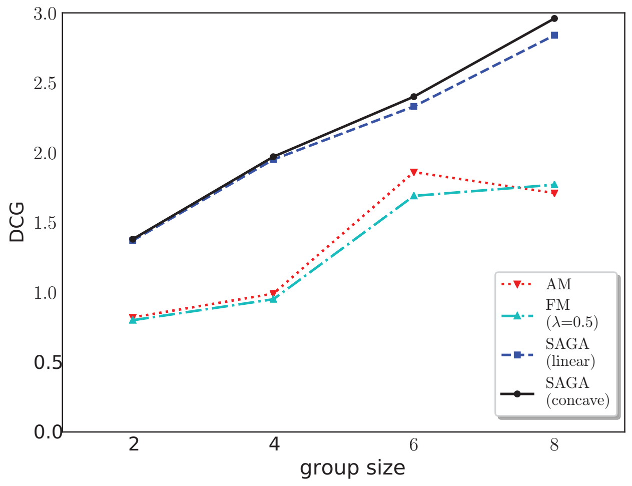

Figure 2: DCG as function of group size (random)

Results & Discussion

Random Groups The results for the random groups are given in Figures 2 and 3. Figure 2 depicts the performance of different algorithms as a function of group size ($|\mathcal{G}|$), for a fixed number of recommendations (**k** = 5) for random groups. The performance of AM and FM algorithms are very similar with AM performing marginally better than FM for groups of smaller sizes. In effect, the leverage of employing disagreements between group members for recommendation is insignificant in case of random groups. Similarly, the performance of the linear and concave versions of the proposed SAGA algorithm are very similar. Howbeit, SAGA ,in general, performs significantly better than both AM and FM.

Figure 3 illustrates the performance of different algorithms as a function of recommendation size (**k**), for a fixed group size ($|\mathcal{G}| = 4$). The plot follows the same pattern as in the case of varying group sizes for random groups. The performance of AM and FM are very close to each other whereas the performance of linear and concave version of the SAGA algorithm are close to each other. Yet, the two

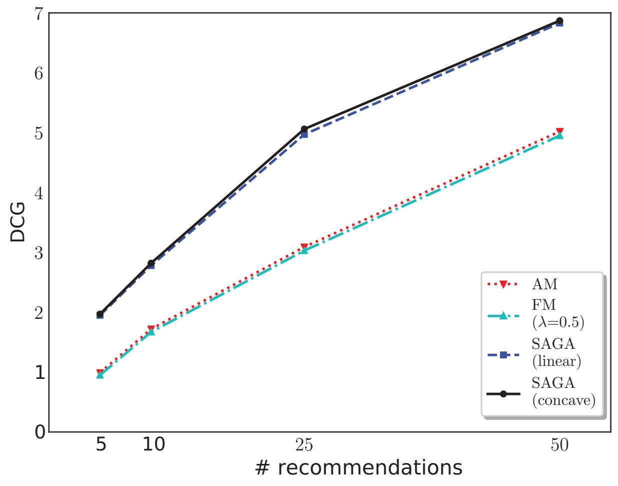

Figure 3: DCG as function of # recommendation (random)

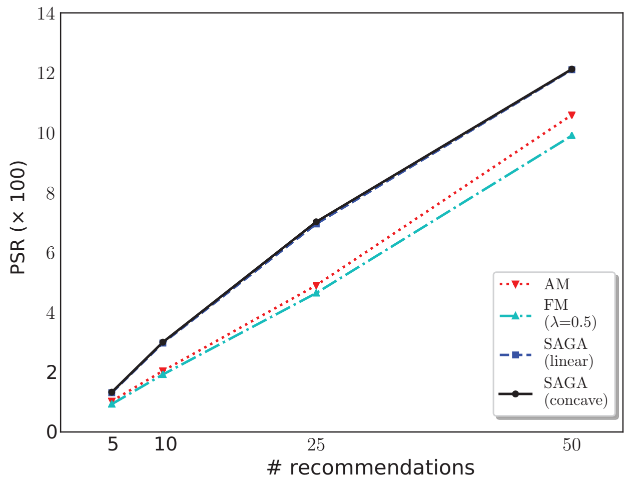

Figure 5: PSR as function of # recommendation (random)

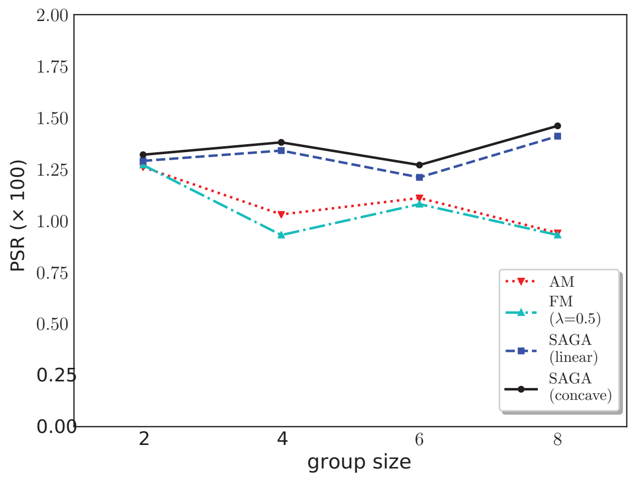

Figure 4: PSR as function of group size (random)

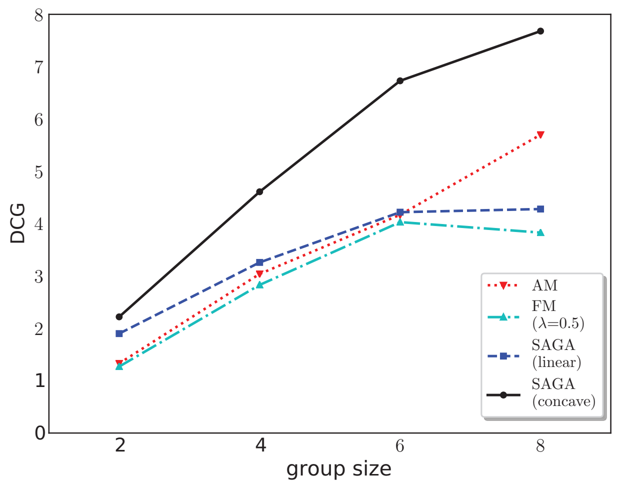

Figure 6: DCG as function of group size (similar)

formulations of SAGA give superior DCG values compared to both AM and FM. For different values of the group size ($\mathcal{G} \in \{10, 25, 50\}$), we observed the same pattern. Due to space limitation, we omitted the corresponding plots.

Figures 4 and 5 shows the coverage performances of different algorithms as functions of group size ($|\mathcal{G}|$) and number of recommendations (**k**). The AM and FM algorithms perform equally but sub-optimal. The SAGA linear and concave versions performs better than AM and FM algorithms with the SAGA concave version performs only marginally better than the linear version. Thus SAGA algorithms recommend relevant but novel items.

Similar Groups The results for the similar groups are given in Figures 6 and 7. Figure 6 illustrates the performance of different GR algorithms as a function of group size, for a fixed number of recommendations (**k** = 5) for similar groups. It is very clear from the plot that AM performs better than FM. Also, as the group size is increased the performance improvement becomes significant. The linear SAGA algorithm performs marginally better than AM for smaller number of groups, but as more number of users are added to the group, the performance deteriorates. Unlike in the case of random groups, in case of similar groups the concave formulation of the *user saturation function* gives significant performance boost over AM, FM and SAGA with linear *user saturation function*. We observed the same pattern by varying the size of the recommendation set.

In Figure 7 the DCG values of different algorithms are plotted as a function of the size of the recommendation set for a fixed group size ($|\mathcal{G}|$) $= 4$. The performance of AM and FM algorithms follows an identitical pattern. For smaller number of recommendations, AM performs better than FM and as the size of the recommendation set increases, FM outperforms AM. But in both cases, the significance is negligible. The linear SAGA algorithm performance degrades below FM and AM as the recommendation size set

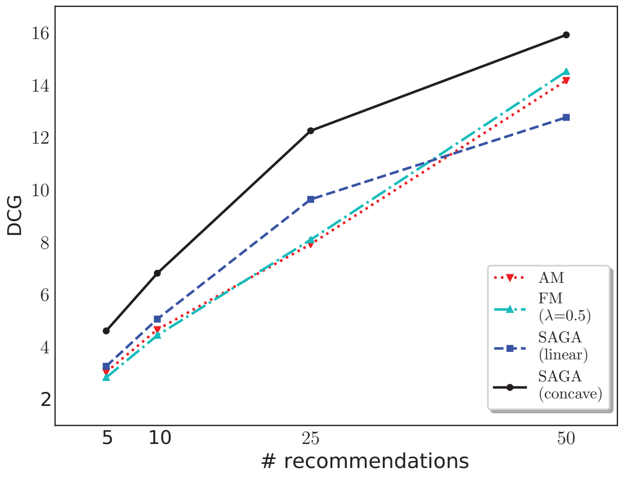

Figure 7: DCG as function of # recommendation (similar)

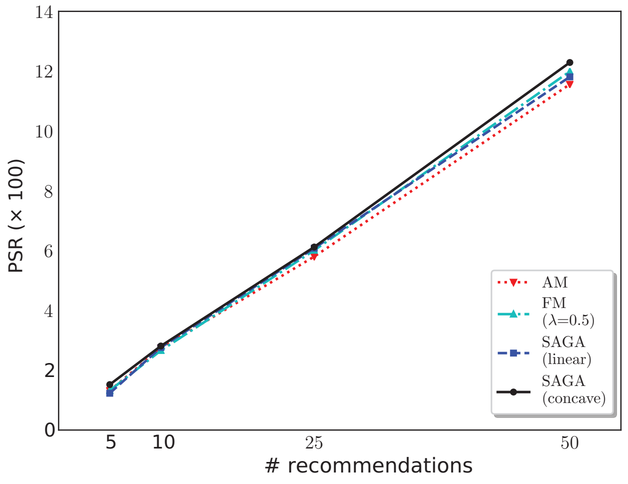

Figure 9: PSR as function of # recommendation (similar)

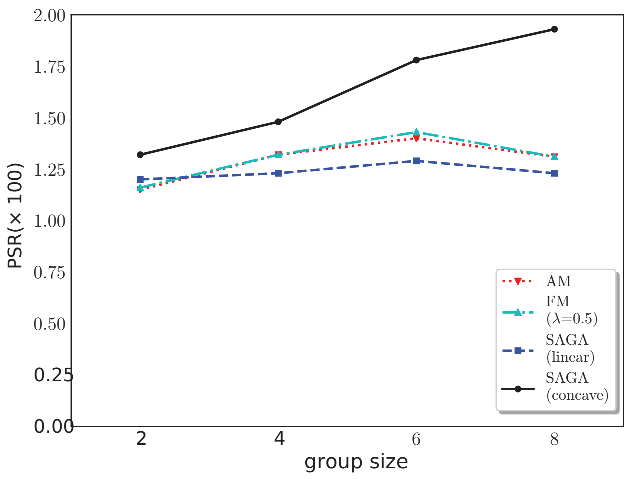

Figure 8: PSR as function of group size (similar)

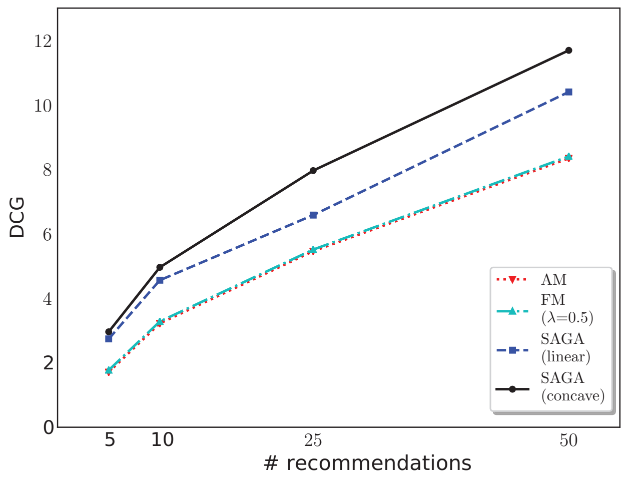

Figure 10: DCG as function of # recommendation (random)

increases. The concave SAGA algorithm gives the best DCG values for all recommendation set size and the performance imporovement is significant. We achieved very similar results by varying the group size.

The performances of coverage metrics for different algorithms are given in Figures 8 and 9. Unlike in the case of random groups, all the algorithms perform equally well, and there is no added benefit of using SAGA algorithm if ones motive is only in serendipitous recommendations.

We will now analyze the performance degradation of the concave SAGA algorithm compared to the linear SAGA algorithm for random groups. In case of random groups, for each group the average user-user similarity value is low (average value is 0.14). Hence for smaller sized groups the performance gain obtained by employing submodular structure over the *user saturation function* is less than employing the linear structure. In other words, for smaller groups since the user interests are very different and the relevant movies are often conflicting, adding an item by the diminishing return property does not result in overall performance gain. We argue that for medium and larger groups, employing a submodular structure does result in performance improvement. Figure 10 contains the DCG values for the random group of size 8. It can be verified that submodularity over *user saturation function* yields better performance.

5 Conclusion

The core idea presented in the paper is that the group recommendation problem can be modeled as a submodular optimization problem. In fact, several important modeling options in group recommendation and social choice theory turn out to be specific instances of the group consensus score function we proposed in this work. The advantage of our approach is that the bundle of items recommended to a group not only depends on the aggregate preference of users (in the group) towards the items but also the affinity (or dissimilarity) between the items. Experiments on real data sets attest to the efficacy of our approach.

6 Acknowledgment

S.P.P thanks Nicolas Usunier and Yves Grandvalet for many fruitful discussions which led us to the core idea presented in this paper. We thank our anonymous reviewers for their valuable comments and suggestions.

References

Amer-Yahia, S.; Roy, S. B.; Chawlat, A.; Das, G.; and Yu, C. 2009. Group recommendation: Semantics and efficiency. *Proceedings of the VLDB Endowment* 2(1):754–765.

Bach, F. 2013. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends® in Machine Learning* 6(2-3):145–373.

Baltrunas, L.; Makcinskas, T.; and Ricci, F. 2010. Group recommendations with rank aggregation and collaborative filtering. In *RecSys*, 119–126. ACM.

Boratto, L., and Carta, S. 2010. State-of-the-art in group recommendation and new approaches for automatic identification of groups. In *Information retrieval and mining in distributed environments*. Springer. 1–20.

Boratto, L.; Carta, S.; and Fenu, G. 2016. Discovery and representation of the preferences of automatically detected groups: Exploiting the link between group modeling and clustering. *Future Generation Computer Systems* 64:165–174.

Ehrgott, M., and Gandibleux, X. 2002. *Multiple Criteria Optimization. State of the art annotated bibliographic surveys*. Kluwer Academic, Dordrecht.

Ekstrand, M. D.; Riedl, J. T.; and Konstan, J. A. 2011. Collaborative filtering recommender systems. *Foundations and Trends in Human-Computer Interaction* 4(2):81–173.

Fisher, M. L.; Nemhauser, G. L.; and Wolsey, L. A. 1978. An analysis of approximations for maximizing submodular set functions-ii. *Polyhedral combinatorics* 73–87.

Jameson, A., and Smyth, B. 2007. Recommendation to groups. In *The adaptive web*. Springer. 596–627.

Järvelin, K., and Kekäläinen, J. 2002. Cumulated gain-based evaluation of ir techniques. *TOIS* 20(4):422–446.

Khoshkangini, R.; Pini, M. S.; and Rossi, F. 2016. A self-adaptive context-aware group recommender system. In *Advances in Artificial Intelligence*. Springer. 250–265.

Kim, J. K.; Kim, H. K.; Oh, H. Y.; and Ryu, Y. U. 2010. A group recommendation system for online communities. *International Journal of Information Management* 30(3):212–219.

Leskovec, J.; Krause, A.; Guestrin, C.; Faloutsos, C.; VanBriesen, J.; and Glance, N. 2007. Cost-effective outbreak detection in networks. In *KDD*, 420–429. ACM.

Liu, Y.; Wang, B.; Wu, B.; Zeng, X.; Shi, J.; and Zhang, Y. 2016. Cogrec: A community-oriented group recommendation framework. In *Social Computing: ICYCSEE*, 258–271. Springer Singapore.

Lu, T., and Boutilier, C. 2011. Budgeted social choice: From consensus to personalized decision making. In *IJCAI*, volume 11, 280–286.

Masthoff, J. 2011. Group recommender systems: Combining individual models. *Recommender systems handbook* 677–702.

Minoux, M. 1978. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*. Springer. 234–243.

Nemhauser, G. L.; Wolsey, L. A.; and Fisher, M. L. 1978. An analysis of approximations for maximizing submodular set functions-i. *Mathematical Programming* 14(1).

O'Connor, M.; Cosley, D.; Konstan, J. A.; and Riedl, J. 2001. Polylens: a recommender system for groups of users. In *ECSCW*, 199–218. Springer.

Puthiya Parambath, S. A.; Usunier, N.; and Grandvalet, Y. 2016. A coverage-based approach to recommendation diversity on similarity graph. In *RecSys*, 15–22. ACM.

Quijano-Sánchez, L.; Díaz-Agudo, B.; and Recio-García, J. A. 2014. Development of a group recommender application in a social network. *Know.-Based Syst.* 71(1):72–85.

Skowron, P.; Faliszewski, P.; and Jerome Lang, J. 2015. Finding a collective set of items: From proportional multirepresentation to group recommendation. In *AAAI*, 2131–2137. AAAI Press.

Steck, H. 2011. Item popularity and recommendation accuracy. In *RecSys*, 125–132. ACM.

Steck, H. 2013. Evaluation of recommendations: rating-prediction and ranking. In *RecSys*, 213–220. ACM.