IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received January 26, 2018, accepted March 10, 2018, date of publication March 14, 2018, date of current version April 23, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2815740

Diversifying Group Recommendation

NGUYEN THANH TOAN[1], PHAN THANH CONG[1], NGUYEN THANH TAM[2], NGUYEN QUOC VIET HUNG[3], AND BELA STANTIC[3]
[1]Ho Chi Minh City University of Technology, Ho Chi Minh City 70000, Vietnam
[2]École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland
[3]Griffith University, Nathan, QLD 4111, Australia

Corresponding author: Nguyen Quoc Viet Hung (quocviethung.nguyen@griffith.edu.au)

ABSTRACT Recommender-systems have been a significant research direction in both literature and practice. The core of recommender systems are the recommendation mechanisms, which suggest to a user a selected set of items supposed to match user true intent, based on existing user preferences. In some scenarios, the items to be recommended are not intended for personal use but a group of users. Group recommendation is rather more since group members have wide-ranging levels of interests and often involve conflicts. However, group recommendation endures the over-specification problem, in which the presumingly relevant items do not necessarily match true user intent. In this paper, we address the problem of diversity in group recommendation by improving the chance of returning at least one piece of information that embraces group satisfaction. We proposed a bounded algorithm that finds a subset of items with maximal group utility and maximal variety of information. Experiments on real-world rating data sets show the efficiency and effectiveness of our approach.

INDEX TERMS Group recommendation, diversification.

I. INTRODUCTION

Recommender-systems has been an important research direction in both literature and practice, especially with the growth of traditional e-commerce applications (e.g. Netflix, Amazon) as well as new Web applications such as social networks (e.g. Facebook, Twitter) and mobile products (e.g. Instagram). The core of recommender systems is the recommendation mechanisms, which suggest to user a selected set of items supposed to match user true intent, often via the relevancy notion of user queries or preferences [1]. The suggestions relate to various decision-making processes, such as what products to buy, what user to follow, or what tweets to read.

Most of the techniques of recommendation systems are designed to individual users. However, in some scenarios the items to be recommended are not intended for personal usage but for a group of users. Such scenarios, for example, include planning a tour for colleagues, looking for a restaurant for close friends, or finding a movie for a family. Group recommendation is rather more complicated than individual recommendation since the preferences of group members are varying and often involve conflicts. Many challenges and associated approaches in individual recommendation are no longer applicable for group recommendation. A major issue in this research area relates to the difficulty of quantifying the objective function and evaluating the effectiveness of group recommendations.

While various group recommendation techniques have been proposed [2], their proposed objective functions are rather heuristics. As a result, there is no winner for all settings. Moreover, group recommendation endures the same over-specification problem as in individual recommendation, in which the presumingly relevant items do not necessarily match user true intent. While users enjoy receiving relevant items, they also tend to loose interest quickly if the recommended items are too similar to each other. Last, but not least, each group member has different taste and interest, receiving similar items might incur bias and conflicts between group members.

In this paper, we address the problem of diversity in group recommendation, which improves group satisfaction by increasing the variety of information shown to group members. The goal of recommendation diversification is to identify a list of items that are dissimilar with each other, but nonetheless relevant to the group's interests. Generating good recommendations is a non-trivial task. On one hand, group members expect to receive content items that are relevant to their interests. On the other hand, group members get bored quickly if all the recommended items are too similar to each other.

The problem of diversifying recommended items for a group of users is more challenging than for individuals. Dissimilar items may have different relevances to different group members and this disagreement among members must be resolved. Therefore, group recommendation diversification becomes a tri-criteria optimization problem, in which we have to (i) maximize the group satisfaction of items, (ii) minimize the disagreement between group members, and (iii) minimize the similarity between items.

TABLE 1. Ratings provided by group members.

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
u_1	10	10	3	3	1	-	7	7
u_2	-	1	9	9	-	1	7	7
u_3	1	-	6	6	10	10	7	7

rating scale: [1-10]
-: missing preference

Example 1: Consider a 3-member group $G = \{u_1, u_2, u_3\}$. Assume that there are 8 items in contention $\{i_1, i_2, ..., i_8\}$, and 2 items are required to be recommended for this group. The users provide preferences in the rating scale from 1 to 10 in Table 1. In terms of average rating, recommending $\{i_7, i_8\}$ to the group would be the best choice. However, i_7, i_8 is similar to each other (in terms of rating or content features). Since the true intention of each member is not known beforehand, it is better to include a novel item such as i_3 or i_4. Moreover, since some preferences could be missing, it would be unfair for items with incomplete preferences. Therefore, the returned set of items should maximize their utility to the group as well as the content dissimilarity between them.

Our contributions and the paper structure are summarised as follows.

- We formalize the problem of diversifying group recommendation and define its semantics as an objective function that aims at maximizing item satisfaction to group and minimizing the similarity between items in Section III.
- We incorporate aggregation functions to combine individual preferences into a single group utility in Section IV.
- We solve the diversified group recommendation in Section V. In that, we study the complexity and diminishing returns properties of the problem. Then we design an efficient algorithm based on the monotonicity and submodularity of the objective function. Finally, we prove the bounded optimality of our algorithm.
- We conduct a comprehensive experimental evaluation in Section VI.

Finally, we describe the related work in Section II and conclude the paper in Section VII.

II. RELATED WORK

A. INDIVIDUAL RECOMMENDATION

From the beginning, recommender systems are designed to make recommendations for individual users. Since recommendations are personalized, different users receive diverse suggestions. The common goal of individual recommendation is to predict user true preference for items he has not rated before, and return items with highest estimated preferences [1], [3]. Existing recommendation strategies are fallen into the most popular categories: content-based similarity analysis [4] and collaborative filtering [5], and some hybrid [6]. The item-based techniques leverages the notion of item similarity to recommend new similar items to old rated items; whereas, the collaborative filtering methods rely on other users who share similar interests. Some further techniques [7] incorporate addtional information such as social network profiles and interconnecting interests. Recommender systems have been applied in various real-world platforms, such as web sites recommendation [8], Amazon's product recommendation [9], Google's news personalization [10], and Netfix's movie recommendation [11]. For more surveys, see [12], [13]. While our work considers group recommendation as a general form, the proposed diversification algorithm can be applied to individual recommendation as well.

B. GROUP RECOMMENDATION

The problem of group recommendation has also been investigated intensively in the literature [14]–[17]. When it comes to a group of interests or a group of closed members, personalized recommendation becomes inapplicable since group members have wide ranging levels of interest and importance. Group recommendation is more challenging than individual recommendation, for survey, see [18]. Even if we know perfectly what is good for individual users, a more complicated question is how to combine individual user recommendation. Various group recommendation methods have been proposed for different data types (e.g. music, movies, TV program) and different groups (e.g. family, friends, social network). Existing methods have focused mostly on aggregating individual preferences to produce recommendations to a group [17]. One approach is to extend the individual recommendation to groups by aggregating group members into a single virtual user and making recommendations to that user, which provides a unified view of a group. For instance, Jameson *et al.* [17] summarizes different strategies for aggregating individual ratings, including average satisfaction, minimum misery, and maximum satisfaction. Another approach is making a seperate recommendation for each user first and then merging the individual lists into a single one for the group, which offers better flexibility for heuristics. For example, the PartyVote system [19] provides a democratic mechanism for selecting music at social events (each group member is guaranteed to have at least one of his preferred songs selected), but only works for group with large options. Vildjiounaite et al. present a TV program recommender system [20] for family based on view history, but not work foor static family group. Most popular group recommendation systems include PolyLens [21], MusicFX [22], and TV recommender [23]. In general, they incorporate group characteristics, social value functions, member rights, and system interfaces. While existing group recommendation rely on heuristics and there is no winner. Our approach increases the potential of capturing true preference intention of group members by providing theoretical and empirical study of diversification.

C. RESULT DIVERSIFICATION

Most information systems focus on increasing the utility of retrieved items and neglect diversity. However, there haas been a push towards diversifying the search results in the last decades [24]. Diversification implies a trade-off between selecting data of relevance to user intent and filtering data having similar characteristics. As such, diversification is often characterized as a bi-criteria optimization problem, in which the twin objectives of being relevant and being dissimilar compete with each other [25]. Most representative diversification techniques include threshold-based approach: Swap [26] and Motley [27]; function-based approach: MMR [28] and MSD [24]; and graph-based approach: Affinity Graph [29] and GrassHopper [30]. For survey and benchmark, see [31]. While our work is orthorgonal to diversification works by developing specific mechanisms for diversified group recommendation, further improvements can be incorporated from these works.

III. MODEL AND PROBLEM STATEMENT

A. SETTING

A group G consists of a set of members $\{u_1, u_2, \ldots, u_n\}$. Denote $D = \{d_1, d_2, \ldots, d_m\}$ is the universal set of items available in the system, from which a recommender has to suggest a set of at most k items to the group G.

User preference $pref(u)$ is modelled in the form of a vector of length m where the value at position j, denoted as $pref(u, d_j)$, provides his preference for the corresponding item d_j. We consider a universal preference model, which is either numeric or ordinal preferences. Under numeric preference model, the user expresses his preference for an item as a real number between 0 and 1, where 1 represents the highest preference (e.g. normalizing movie ratings in 10 or 100 score band to [0, 1]). Under an ordinal preference model, user relies on a discrete set of values (e.g. 'not liked', 'neutral', 'liked', 'very liked'). It is noteworthy that ordinal values are more generic than categorical values, since there exist an ordering between preferences.

Let L be the rating domain. Then, user preferences are modeled as an $n \times m$ *rating matrix*:

$$M = \begin{pmatrix} m_{11} & \ldots & m_{1k} \\ \ldots & \ldots & \ldots \\ m_{n1} & \ldots & m_{nk} \end{pmatrix}$$

where $m_{ij} \in (L \cup \{\ominus\})$ for $1 \leq i \leq n$, $1 \leq j \leq m$. Here, the special label $\ominus$ denotes that a user did not assign a preference to an item. We write $M(u, d)$ to denote the preference of user u for item d.

A recommender will rank the items in a decreasing order of *group utility*, which reflects the degree to which the item is preferred by the members. For any utility function $r : D \rightarrow R_{\geq 0}$, which returns the non-negative relevance score for each item in D with respect to a group of users, our goal is to find a subset I of k items, which are most preferred to the group and diversified among themselves. Here the positive integer k is of particular practical relevance for recommendation systems [1]. An appropriate value for k depends on the user and the application context.

Denote $S(d, d')$ be a measure of similarity between two items d_i and d_j. Our model accepts any non-negative, symmetric similarity function (i.e. $S(d, d') = S(d, d')$ and $S(d, d') \geq 0$). When we describe the objective function as well as the proposed optimization algorithm, it is convenient to introduce the weight factor $q(d) = \sum_{d' \in D} S(d, d')r(d')$, which measures the importance of item i. To be specific, if d_i is similar to many items that are preferred by the group, it is more important than the items whose neighbors are not preferred. For example, if d is close to the center of a big cluster preferred by the group, the value of $q(d)$ is large.

B. RECOMMENDATION PROCESS

We follows a two-step approach, which is illustrated in Fig. 1. The input is a rating matrix provided by a group for some items. The first step is responsible for *Aggregating Individual Preferences*, which combines different user preferences into a single group utility that ranks the items. Section IV will discuss in details the design principles and concrete group utility functions. However, the group utility often ranks similar items consecutively (otherwise, they are less likely to be similar). While group members embrace the overall satisfaction, they also tend to loose interest quickly if the recommended items are too similar to each other. Since individual as well as group true intention is not known beforehand, we increase the chance of recommendation matching true preferences by *Diversifying Group Aggregation*, which optimizes both utility and diversity as a bi-objective problem. Section V realizes this step by proposing an efficient solution to the diversified group recommendation problem formulated below.

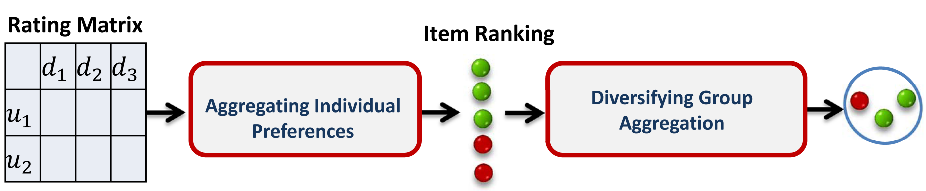

FIGURE 1. Diversified group recommendation.

C. DIVERSIFIED GROUP RECOMMENDATION

Our goal is to find a subset $I_G \subseteq D$ of k items which are both useful to the group G and diversified among themselves. To this end, we propose the following optimization problem.

Problem 1 (Diversified Group Recommendation): Given a group of users G, a universal set of items D, a group aggregation function $r : D \to R_{\geq 0}$, returns a list of item I_G of k items that maximize the objective function:

$$I_G = \arg\max_{I \subseteq D} g(I) \tag{1}$$

where

$$g(I) = w \sum_{d \in I} q(d) r(d) - \sum_{d,d' \in I} r(d) S(d, d') r(d') \tag{2}$$

where w is a positive regularization parameter that defines the trade-off between the two terms, and I consists of the item that will be returned in the group recommendation.

Intuitively, in the objective function $g(I)$, the first term measures the weighted overall utility of I with respect to the group, and $q(d_i)$ is the weight for $r(d_i)$. It favors relevant examples from big clusters. In other words, if two items are equally preferred by the group, one from a big cluster and the other isolated, by using the weighted relevance, we prefer the former. The second term measures the similarity among the items within I. That is, it penalizes the selection of multiple preferred items that are very similar to each other. By including this term in the objective function, we seek a set of items which are preferred to the group, but also dissimilar to each other.

IV. AGGREGATE INDIVIDUAL PREFERENCES

Now we design the group utility function $r(.)$ that maximizes average satisfaction and ensures some degree of fairness. The output of the utility function is a vector of length m.

A. DESIGN PRINCIPLES

Our function takes into account the following requirements:

(R1) User satisfaction: reflects the degree to which the item is preferred by the members. The more group members prefer an item, the higher its score should be for the group.

(R2) Fairness: reflects the level at which members disagree with each other.

B. GROUP UTILITY FUNCTION

1) NUMERIC PREFERENCE MODEL

User satisfaction (R1) is modelled as follows. The satisfaction of an item d to a group G, denoted as $pref(G, d)$, is an aggregation over $pref(u, d)$ where $u \in G$. We consider two main aggregation strategies:

- Average: $pref(G, d) = \frac{1}{|G|} \sum_{u \in G} M(u, d)$. This approach considers each item independently and each user equally. A disadvantage of this approach is the sensitivity to outliers, e.g., some users could provide very high rating scores or very low rating scores. An improved version to avoid the outlier sensitivity is Beta model [32], in which 5% upper scores and 5% lower scores will be excluded from the aggregation.
- Least-Misery: $pref(G, d) = \min_{u \in G} M(u, d)$. This is another most prevalent mechanisms being employed recently [17]. This mechanism captures cases where some user has a strong preference (e.g., a vegetarian cannot go to a steakhouse) and that user's preference acts as a decider.

Fairness (R2) is modelled as follows. The disagreement of a group G over an item d, denoted as $dis(G, d)$, reflects the degree of consensus in the preference scores for d among group members. Intuitively, the closer the preference scores for d between users u and v, the lower their disagreement for d. We consider the following two main disagreement computation methods:

- Average pair-wise disagreements:

$$dis(G, d) = \frac{2}{|G|(|G| - 1)} \sum_{u,v \in G} |M(u, d) - M(v, d)|$$

  where $u \neq v$ and $u, v \in G$. This function computes the average of pair-wise preference differences for the item among group members.
- Disagreement variance:

$$dis(G, d) = \frac{1}{|G|} \sum_{u \in G} (M(u, d) - mean)^2$$

  where $mean = \frac{1}{|G|} \sum_{u \in G} M(u, d)$ is the mean of all the individual preferences for the item. This function computes the mathematical variance of the preferences for the item among group members.

Finally, we combine the user satisfaction and the fairness for an item in the utility function. Formally, the utility function, denoted as $r(G, d)$ (or $r(d)$ for short), combines the group satisfaction and the group disagreement of d for G into a single group recommendation score using the following formula:

$$r(G, d) = w_1 \times pref(G, d) + w_2 \times (1 - dis(G, d)) \tag{3}$$

where $w_1 + w_2 = 0$ and each specifies the relative importance of satisfaction and fairness in the overall utility score.

2) ORDINAL PREFERENCE MODEL

The preference of a user u is transformed to an ordering τ of a subset $I \subseteq D$; i.e., $\tau = [i_1 \geq i_2 \geq \ldots]$, with each $i_j \in I$ and $\geq$ is some ordering relation on I. τ is also called a rank list on D; i.e. $\tau(d) = j$ is the rank of d w.r.t. τ. Let $|\tau|$ denote the number of elements in τ. τ might not be a full list; i.e. $|\tau| < |D|$. Denote $\Theta = \{\tau_1, \ldots, \tau_n\}$ is the set of all user preferences. The problem output is to determine an aggregated ranking $r(.)$ such that $r(.)$ is a full list over the union of elements of $\tau_1, \ldots, \tau_n$; i.e. $r : \cup_{\tau \in R} \cup_{i \in \tau} i \to [1, |D|]$.

The research efforts on solving the ranking aggregation problem can be categorized into the following methods.

- *Score based:* This approach aggregates the final ranking by computing the ranking scores for each item (higher the score, better the rank). The process consists of following steps. In the first step, for each item $d \in D$, we will compute the normalized ranking scores $w_{\tau_1}(o), \ldots, w_{\tau_n}(d)$ over all user preferences $\tau \in \Theta$. Several normalization computations [33] include score normalization, Z-score normalization, rank normalization, and Borda rank normalization. In the second step, we will compute the aggregated ranking score $r(d)$ of the item d by combining its normalized ranking scores; i.e. $r(d) = f(w_{\tau_1}(d), \ldots, w_{\tau_n}(d))$. One simple way to implement the aggregation function $f(.)$ is using the sum, min, or max [34]; e.g. $r(d) = \sum_{\tau \in \Theta} w_\tau(d)$. A complex implementation is using a weighted version of the sum [34]: $r(d) = h_\Theta(d) \sum_{\tau \in \Theta} w_\tau(d)$, where $h_\Theta(d)$ is the number of occurrences of d over all user preferences in Θ with the idea is that the items appear in more user preferences are likely to be more important. In the final step, the aggregated ordering can be simply obtained following the decreasing order of the aggregated ranking scores.

- *Distance based:* This approach computes the ranking aggregation by solving an optimization formulation, in which the objective function is the distance between user orderings. Formally, $\Delta(.)$ is the distance measurement between two or many orderings. The ranking aggregation problem then becomes finding an aggregated ranking $r(.)$ such that the distance value $\Delta(r, \tau_1, \ldots, \tau_n)$ is minimal. A wide range of distance measures has been proposed [35], such as Spearman footrule distance (which uses the absolute difference between the ranks of an item according to the given rankings τ_i and τ_j) and Kendall distance (which uses the number of pairwise adjacent transpositions needed to transform from ranking τ_i to another ranking τ_j).
In general, the optimization formulation of ranking aggregation is intractable (e.g., using the Kendall distance with $k = 4$ is NP-Hard [36]). Therefore, a wide range of important properties that an aggregation solution needs to satisfy have been studied in the literature, such as Condorcet property [37].
- *Probability based:* The methods in this category capture the item ranking via probabilistic interpretation. Technically, for two given items d_i and d_j, we will compute the probability of an item d_i having a greater order of an item d_j; i.e. $Pr(d_i > d_j)$. Several probabilistic models to compute $Pr(d_i > d_j)$ have been proposed, such as Bradley-Terry model [38] and Thurstone model [39]. The Bradley-Terry model formulates a logistic function over the true ranking scores of d_i and d_j (i.e., $Pr(d_i > d_j) = \frac{e^{\tau(d_i)}}{e^{\tau(d_j)} + e^{\tau(d_j)}}$) and performs a log-likelihood maximization to compute all the pairwise probability values and the true ranking scores simultaneously. The Thurstone model follows a similar process, in which the ranking score for each item has a Gaussian distribution.
With the similar idea of computing pair-wise ranking probabilities, one can use Markov chains [36] in which the states correspond to the items to be ranked and the transition probability from state i to state j is the probability of the item d_i has a higher order of the item d_j w.r.t. some user rankings $\tau \in \Theta$. As such, computing the aggregated ordering is equivalent to determining the stationary probability distribution of the Markov chains, which can be done in polynomial time [36]. Probabilistic models in general and Markov chains in particular not only offer a parameterizable approach but also open ways to integrate different heuristics into the probability formulation (e.g., one could use other distributions rather than Gaussian distribution). As such, the ranking aggregation can be iteratively refined by these heuristics, producing a fine-grained aggregated ranking.

In sum, while the *score-based* focuses on computing a unified utility score for the ranking, the *distance-based* method aims to minimize the differences between the final utility and individual ones. Taking advantages of the two, the *probability-based* allows more fine-grained combination of ordinal preferences.

V. DIVERSIFY GROUP AGGREGATION

In this section, we present the optimization solution for diversified group recommendation problem. We start by analyzing the problem complexity, and then study the properties of the objective function, followed by a greedy algorithm.

A. PROBLEM COMPLEXITY

Recall that in the diversified group recommendation problem, we want to find a subset of k items that collectively maximize the objective function. Unfortunately, by the following theorem, it is NP-hard to find the optimal solution.

Theorem 1: Problem 1 is NP-hard.

Proof: We prove Theorem 1 by reduction to the Densest k-Subgraph problem, which is known to be NP-Complete [40]. Let $G = (V, E)$ be an undirected graph with vertices V and edges E. Let W be the $|V| \times |V|$ binary connectivity matrix (symmetric), i.e., $W_{i,j} = 1$ if $\{i, j\} \in E$, and $W_{i,j} = 0$ otherwise. Then, the Densest k-Subgraph problem requires identifying a subgraph of k vertices with a maximal number of edges:

$$\arg\max_{\hat{V} \subseteq V,\ |\hat{V}|=k} \sum_{i,j \in \hat{V}} W_{i,j}$$

which is equivalent to

$$\arg\max_{I=(V \setminus \hat{V}), |\hat{V}|=k} 2 \sum_{i \in \hat{V}, j \in I} W'_{i,j} + \sum_{i,j \in I} W'_{i,j} \tag{4}$$

where $W'_{i,j} = 1 - W_{i,j}$. Now we will show that Eq. 4 can be viewed as an instance of the optimization problem in Eq. 1. To this end, let all utility scores be one ($r(d) = 1$ for all $d \in D$) and choose $w = 2$. Then, our objective function $g(I)$ becomes:

$$\begin{aligned} g(I) &= 2\sum_{d\in I} q(d) - \sum_{d_1,d_2\in I} S(d_1,d_2) \\ &= 2\sum_{d_1\in I}\sum_{d_2\in D} S(d_1,d_2) - 2\sum_{d_1,d_2\in I} S(d_1,d_2) \\ &\quad + \sum_{d_1,d_2\in I} S(d_1,d_2) \\ &= 2\sum_{d_1\in(D\setminus I)}\sum_{d_2\in I} S(d_1,d_2) + \sum_{d_1,d_2\in I} S(d_1,d_2) \end{aligned} \tag{5}$$

The latter is equivalent to the objective function in Eq. 4, so that selection of k items corresponds to the finding the densest subgraph of $(|V| - k)$ nodes. □

B. DIMINISHING RETURNS PROPERTIES

Given that Equation 1 is NP-hard in general, we seek for a provably near-optimal solution. It turns out that it is possible to find such a solution based on the diminishing returns properties of the objective/goodness function $g(.)$.

The first property considers the influence of the utility scores. We observe that the higher utility a item is to the group, the higher are the chances of it to be part of the recommendation.

Proposition 1 (Strength of Utility): Let D be a corpus of items, r an utility ranking, $I \subset U$ a recommendation, and $d \in D \setminus I$ a non-recommended item. Let r' be an utility score defined such that $r'(d) > r(d)$ and $r'(x) = r(x)$ for $x \in D \setminus \{d\}$. For $w \geq 2$ it holds that:

$$g_{r'}(I \cup \{d\}) \geq g_r(I \cup \{d\})$$

Proof: The strength of utility follows by this transformation ($w \geq 2$):

$$\begin{aligned} & g_{r'}(I \cup \{d\}) - g_r(I \cup \{d\}) \\ &= g_{r'}(I) + wq(d)r'(d) - 2\, r'(d)\sum_{x\in I} S(x,d)r'(x) \\ &\quad - g_r(I) - wq(d)r(d) + 2\, r(d)\sum_{x\in I} S(x,d)r(x) \\ &= wq(d)[r'(d) - r(d)] - 2\sum_{x\in I} S(x,d)r(x)[r'(d) - r(d)] \\ &= w\sum_{x\in D} M(x,d)r(x)[r'(d) - r(d)] \\ &\quad - 2\sum_{x\in I} M(x,d)r(x)[r'(d) - r(d)] \\ &= (w-2)\sum_{x\in I} M(x,d)r(x)(r'(d) - r(d)) \\ &\quad + w\sum_{x\in D\setminus I} M(x,d)r(x)[r'(d) - r(d)] \geq 0 \end{aligned}$$

□

Our notion of goodness further shows monotonicity. That is, when adding more items to an existing recommendation, the goodness of the overall recommendation will increase.

Proposition 2 (Monotonicity): Let D be a corpus of items, r a utility ranking, $I \subset D$ a recommendation, and $I' \subseteq (D\setminus I)$ a set of non-selected items. For $w \geq 2$ it holds that:

$$g(I \cup I') \geq g(I)$$

Proof: Monotonicity follows by the following transformation ($w \geq 2$):

$$\begin{aligned} & g(I \cup I') - g(I) \\ &= w\sum_{x\in I'} q(x)r(x) - (\sum_{x\in I', x'\in I} r(x)S(x,x')r(x') \\ &\quad + \sum_{x\in I, x'\in I'} r(x)S(x,x')r(x') \\ &\quad + \sum_{x,x'\in I'} r(x)S(x,x')r(x')) = w\sum_{x\in I'} r(x)\sum_{x'\in D} S(x,x')r(x') \\ &\quad - (2\sum_{x\in I, x'\in I'} r(x)S(x,x')r(x') + \sum_{x,x'\in I'} r(x)S(x,x')r(x')) \\ &\geq 2\sum_{x\in I'} r(x)\sum_{x'\in D} S(x,x')r(x') - (2\sum_{x\in I, x'\in I'} r(x)S(x,x')r(x') \\ &\quad + \sum_{x,x'\in I'} r(x)S(x,x')r(x')) = 2\sum_{x\in I'}(\sum_{x'\in D} S(x,x')r(x') \\ &\quad - \sum_{x'\in I\cup I'} S(x,x')r(x')) = 2\sum_{x\in I'}\sum_{x'\notin I\cup I'} S(x,x')r(x') \geq 0 \end{aligned}$$

□

Finally, our goodness function shows submodularity, which refers to the property that marginal gains in goodness start to diminish due to saturation of the objective. That is, the marginal benefit of adding items to the recommendation decreases w.r.t. the size of the recommendation.

Proposition 3 (Submodularity): Let D be a corpus of items, r a utility ranking, $I \subset D$ a recommendation, and $d, d' \in D \setminus I$ non-selected items. Then, it holds that:

$$g(I \cup \{d\}) + g(I \cup \{d'\}) \geq g(I \cup \{d,d'\}) + g(I)$$

Proof: Submodularity follows by the following transformation:

$$\begin{aligned} & g(I \cup \{d\}) + g(I \cup \{d'\}) \\ &\quad \geq g(I \cup \{d,d'\}) + g(I) \Leftrightarrow g(I \cup \{d'\}) - g(I) \\ &\quad \geq g(I \cup \{d,d'\}) - g(I \cup \{d\}) \Leftrightarrow wq(d')r(d') \\ &\qquad - 2\, r(d')\sum_{x\in I} r(x)S(x,d') \\ &\quad \geq wq(d')r(d') - 2\, r(d')\sum_{x\in I\cup\{d\}} r(x)S(x,d') \\ &\quad \Leftrightarrow 2\, r(d)r(d')S(d,d') \geq 0 \end{aligned}$$

□

C. GREEDY ALGORITHM

Now we attempt to develop an algorithm for solving the diversified group recommendation problem. First of all, we propose a greedy algorithm that well approximates the optimization objective in general. Then, we give a complexity analysis and finally we provide an illustrative example of the algorithm.

1) ALGORITHM DESCRIPTION

In light of the complexity result in Theorem 1, we look for heuristics that can approximate the optimization objective. The challenge is that using simple greedy algorithms based on thresholding [26], [27] has no guarantee on the group utility of the output. To overcome this challenge, we propose the following greedy algorithm, whose near-optimizality can be bounded, based on the diminishing returns properties in Section V-B. The idea is that we try to expand the list of recommended items one-by-one to maximizing the objective function for k iterations. At each iteration, we need to identify the item d to maximize $g(I \cup \{d\})$. As a result, a naive way to find d is to traverse all of the remaining items $D \setminus I$ for each iteration.

Algorithm 1 A Greedy Algorithm for Diversified Group Recommendation

input : A set of items D, a group of users G
A non-negative group utility function r
An item similarity matrix S
A weight $w \geq 2$, and a budget k

output: A subset I of k items to be recommended for the group

`// Step 1: Initilization`
1 Compute the utility score $r(d)$ for each item $d \in D$ in context of G ;
2 Compute the weight factor $q(d) = \sum_{d' \in D} S(d, d') \cdot r_{d'}$ for each item $d \in D$;
3 Initialize I as an empty list ;
4 Initialize the ranking score $s(d) = wq(d)r(d)$ for each item d ;
`// Step 2: Greedy Selection`
5 **for** *k iterations* **do**
6 | Pick $x = \arg\max_{d \in D, d \notin I} s(d)$;
7 | Append x to I ;
8 | Update the ranking score $s(d) = s(d) - 2\ r(x)S(d, x)r(d)$ for remaining items $d \in D \setminus I$;
9 **return** I

The details of our greedy algorithm are given in Algorithm 1. It takes a set of items D, a group of users G, a group utility function $r(.)$, a item similarity matrix S, a regularization parameter w, and a budget k as input and returns a ranking list I of k items to be recommended for a group of users (the first item has the highest rank). In the initialization step, we begin by computing the utility score of each item. As mentioned above, our approach does not have any restriction on the utility model. Besides, we also compute the weight factor $q(.)$ and the ranking score $s(.)$ for each item. In the greedy selection step, we perform k iterations to select k items into the ranking list I. At each iteration, we add one more item with the highest ranking score into the current list (line 6). Our algorithm is guaranteed to converge since it iterates k times and the measures (utility, similarity, etc.) can be normalized to [0, 1] before-hand.

2) ALGORITHM ANALYSIS

Now we analyze our proposed algorithm through the following guarantees. First, we observe that the approximation error of the proposed algorithm is bounded.

Guarantee 1 (Near-Optimality): Alg. 1 is a (1- 1/e)-approximation for diversified group recommendation.

Proof: Following the analysis of diminishing returns properties in [41], we have the fact that given a subset I of k items constructed greedily by selecting a item x one at a time with largest marginal increase $g(I \cup \{x\}) - g(I)$, we have $g(I) \geq (1 - 1/e)g(I^*) \approx 0.63\ g(I^*)$, where I^* is the optimal solution. Our greedy heuristic in Alg. 1 resembles this fact in the sense that the ranking score of each item d is initialized as $wq(d)r(d)$ (line 4) and subtracted a quantity of $2\ r(x)S(d, x)h(d)$ (line 8) each iteration, which ends up equal to $g(I \cup \{x\}) - g(I)$. As a result, selecting the item with highest ranking score is equivalent to maximizing the marginal increase of g at each iteration. □

Guarantee 2 (Complexity): The time complexity and the space complexity of Alg. 1 is $\mathcal{O}(|D|^2 + k|D|)$ and $\mathcal{O}(|D|^2)$, respectively.

Proof: For time complexity, we have a quadratic term and a linear term. The quadratic term $|D|^2$ comes from the computation of weight factor in the initialization step. The linear term $k|D|$ comes from the fact that in each of k iterations, we compute the ranking score for all remaining tags and iterate them for choosing the one with highest score.

For space complexity, it can be easily seen that the only expensive cost is to store the similarity between all pairs of items, whose exact size is $\frac{|D||D-1|}{2}$. The similarity between an item and itself is unnecessary to be stored; i.e. $S(d, d) = 0$. □

Further, our algorithm shows stability in the recommendation, which is important to support incremental recommender systems. If a group is first presented with the top-5 items, but then extends the result to the top-10, the expectation is clearly that the top-10 remain unchanged.

Guarantee 3 (Stability): For I as returned by Alg. 1, let $I_{k_1} = \{d_1, \ldots, d_{k_1}\}, I_{k_2} = \{d'_1, \ldots, d'_{k_2}\}$ be selections with $d_i \in I$ for $1 \leq i \leq k_1$ and $d'_j \in I$ for $1 \leq j \leq k_2$, and $0 < k_1 \leq k_2$. Then, it holds that $d_i = d'_i$ for $1 \leq i \leq k_1$.

Proof: In Alg. 1, the construction of I is performed stepwise and elements are never removed from I. The selection also is deterministic: we always add the item with highest ranking score (line 6). Thus, a larger selection sequence comprises a smaller selection sequence as a prefix. □

Finally, we also highlight that the selection heuristic is fair in the sense that it is genuinely driven by the utility function.

Guarantee 4 (Fairness): Let D be a corpus of items. For any set of items $I \subset D$, there exists an utility function r, s.t. Alg. 1 returns $I^* = I$.

Proof: Given I, we define r as $r(d) = 1$ if $d \in I$ and $r(d) = 0$ otherwise. Then, the ranking score $s(d)$ is positive if $d \in I$, whereas $s(d) = 0$ if $d \notin I$. Hence, the algorithm selects only elements from I. □

Example 2: Continuing the motivating scenario in Example 1, our algorithm runs as follows. Using average aggregation with $w_1 = 1$ and $w_2 = 0$, we have $r(d_1) = r(d_2) = r(d_5) = r(d_6) = \frac{10+1}{3} = 3.67$, $r(d_3) = r(d_4) = \frac{3+9+6}{3} = 6$, $r(d_7) = r(d_8) = 7$. Let us define $S(d_i, d_j) = 1/|r(d_i) - r(d_j)|$, we have $q(d_1) = q(d_2) = q(d_5) = q(d_6) = 6/(6-3.67) + 7/(7-3.67) = 4.68$, $q(d_3) = q(d_4) = 3.67/(6-3.67) + 7/(7-6) = 8.58$, $q(d_7) = q(d_8) = 3.67/(7-3.67) + 6/(7-6) = 7.10$. Now running Alg. 1 with $w = 2$, at the beginning, we have $s(d_1) = s(d_2) = s(d_5) = s(d_6) = 2*4.68*3.67 = 34.35$, $s(d_3) = s(d_4) = 2*8.58*6 = 102.96$, and $s(d_7) = s(d_8) = 2*7.10*7 = 99.4$. Therefore, at the first iteration, we can select d_3 (or d_4). Then, the ranking score is updated, $s(d_1) = s(d_2) = s(d_5) = s(d_6) = 34.35 - 2*6/(6-3.67)*3.67 = 15.45$, $s(d_4) = 102.96 - 2*6/(6-6)*6 \ll 0$, $s(d_7) = s(d_8) = 99.4 - 2*6/(7-6)*7 = 15.4$. Therefore, at the second iteration, we can select d_1 (or d_2, d_5, d_6). The procedure continues until we reach the pre-defined budget k of recommendation.

VI. EVALUATION

A. EXPERIMENTAL SETUP

1) DATASETS

We utilize the following datasets (see also Table 2):

- *MovieLens:* The dataset contains 1682 movies rated by 943 users. 100,000 ratings ranging from 1 to 5 were given by these users. Each user rated at least 20 movies [1].
- *TripAdvsor:* The dataset contains 37K ratings about 2K hotels provided by 34K users [42].
- *Amazon:* The dataset contains 16K MP3 player reviews provided by 15K users for 686 items [43].

TABLE 2. Datasets.

Dataset	#Items	#Users	#Ratings
MovieLens	1,682	943	100K
TripAdvisor	2,232	34,187	37,181
Amazon	686	15,004	16,680

These datasets can be used for both numeric preference model and ordinal preference model, as the ratings are limited to a small ordered set of integers.

2) EVALUATION METRICS

We use the following measures:

- *Subtopic recall (S-Recall):* A popular metric to evaluate the diversity of recommendation is subtopic recall. For example, an item on Amazon has description and belong to different categories/tags, which may cover many subtopics, so that a set of items is diverse if it contains many subtopics. For a group G, the metric measures the proportion of unique subtopics retrieved in the recommendation result I_G:

$$S\text{-}Recall(G, I_G) = \frac{|\bigcup_{d \in I_G} subtopics(d)|}{subtopics(\{d \in D | \exists u \in G, M(u,d) \neq \ominus\})} \quad (6)$$

  where the term in the divisor represents the set of all items receiving at least one rating.
- *Normalized utility:* This metric measures the utility of the diversified recommendation w.r.t. the top-k item set returned, i.e., it indicates how well a recommendation preserves utility when diversifying the result. Formally, normalized utility ($\in [0, 1]$) of a top$k$ item recommendation I from corpus D is defined as the sum of selected utility scores over the sum of the k highest utility scores:

$$nR(D, r, I) = \frac{\sum_{d \in I} r(d)}{\max_{I' \subseteq D, |I'|=|I|} \sum_{d \in I'} r(d)} \quad (7)$$

  Here, a higher score indicates higher utility and $nR(D, r, I) = 1$ means that I is exactly the recommendation of items with the highest utility scores.

There are other metrics for recommendation systems [18]. However, it would be inapplicable to use them in our setting since they do not consider the diversity aspect. Moreover, novel recommendation systems rely on user study [2], which we also perform in Section VI-D.

3) GROUP FORMATION

We report efficiency and effectiveness results for the recommendation under different group utility functions. Performance is evaluated by mainly varying three parameters: k – the number of items in the generated recommendation, n – group size, and m – total number of items. A group is simulated by random sampling from the user pool following the long-tail distribution of user-rating (user with more ratings is more likely to be chosen). The total running time of the system is the aggregated running time of group utility function and the recommendation.

4) BASELINES

For a competitive evaluation, a *brute-force* algorithm is implemented by enumerating and evaluating all possible combinations for selecting the best recommendation according to the good function $g(.)$. Traditional group recommendation techniques are compared via a baseline called *utility-only*, which returns the top-k items with highest utility values. Another baseine is k-*medoids* clustering [44], which generates k clusters of item sets according to group utility and picks a representative from each cluster. We study the two competitive group utility functions: *average-based* (average satisfaction and average disagreement) [32] and *score-based* [34].

[1] http://www.grouplens.org/node/73

5) EXPERIMENTAL ENVIRONMENT

Experimental results have been obtained on an Intel Core i7 system (3.4GHz, 12GB RAM). The results are averaged over 100 runs.

B. RUNTIME PERFORMANCE

1) EFFECTS OF DATA DOMAINS

We evaluate the running time of recommendation with the item size $n = 100$, recommendation size$k = 5$, and group size $m = 10$. The users and items are chosen randomly from each dataset. The three recommendation algorithms are compared: *brute-force*, *k-medoids*, and *greedy*. The average-based group utility function is used with $w_1 = w_2 = 0.5$. The trade-off parameter between utility and diversity is $w = 2$.

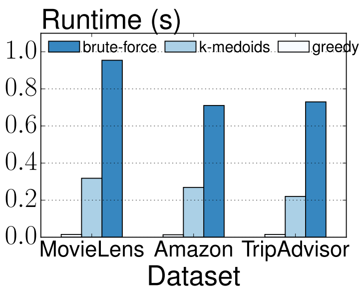

FIGURE 2. **Effects of data size on running time.**

Fig. 2 illustrates the efficiency result for three aforementioned datasets. It can be clearly seen that our proposed *greedy* algorithm is the fastest one since it runs in linear time if the pre-processing time of item similarity computation is not taken into account. Another interesting finding is that the running time on MovieLens dataset is quite slow since its rating matrix is more dense than the other datasets.

2) EFFECTS OF RECOMMENDATION SIZE

To study the effects of the top-k value – recommendation size – on the computation time required by our greedy algorithm for diversified group recommendation. We vary k from 10 to 50, which is a suitable range for user cognitive load. According to the previous experiment, we choose MovieLens for representative evaluation with 1000 users and 1000 items are chosen randomly. We also study two group utility functions: *average-based* and *score-based*.

Fig. 3 shows the computation time (in ms) with respect to the recommendation size. We observe that a solution is obtained quickly, in less than 170ms for $k = 50$, which can be seen as the maximum number of items a group can discuss with each other. In fact, we observe a linear trend of computation time despite the super quadratic time complexity of our algorithm. This highlights that our approach is efficient for real datasets.

C. EFFECTIVENESS OF GROUP RECOMMENDATION

Next, we study the effects of varying the top-k value on the diversity and utility of the result. We use S-Recall

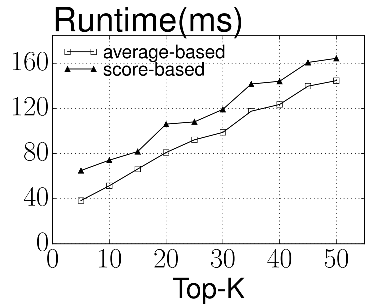

FIGURE 3. **Effects of recommendation size on running time.**

and Normalized Utility to measure diversity and utility of the group recommendation returned by our approach, respectively. We randomly set the tuning parameter w_1, w_2 (trade-off between satisfaction and fairness) and w (trade-off between utility and diversity) according to uniform distributions $\mathscr{U}(0, 1)$ and $\mathscr{U}(2, 10)$, respectively. For each dataset, 1000 users and 1000 items are chosen randomly. The results are averaged across the used datasets. Three group recommendation methods are compared: *greedy*, *k-medoids*, and *utility-only*. The *brute-force* method is intractable, thus is not evaluated.

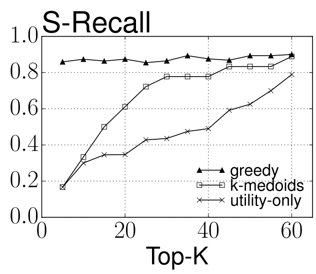

FIGURE 4. **Top-*k* vs. diversity.**

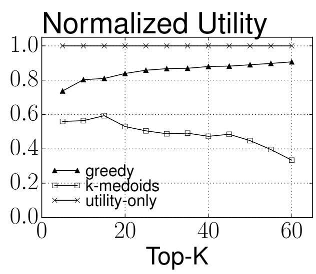

FIGURE 5. **Top-*k* vs. utility.**

The results are depicted in Fig. 4 and Fig. 5. Both the *k-medoids* and *utility-only* baselines are limited by the trade-off between utility and diversity. They have either small S-Recall with large Normalized Utility or large S-Recall with small Normalized Utility. Our proposed *greedy* algorithm outperforms *k-medoids* in terms of normalized utility and even better than two of them in terms of S-Recall. When increasing the top-k value, the *greedy* method increases S-Recall and Normalized Utility increase as well. This is expected because of two reasons. First, when the size of the recommendation set increases, more dissimilar item are included as a result of the output objective of Alg. 1, leading to higher S-Recall. Second, if we consider larger results, more useful items are chosen by our algorithm since the chance that they are dissimilar is higher, leading to higher Normalized Utility. We conclude that our algorithm is stable and (except for a very few outliers) non-decreasing with the number of representative items presented to user.

D. USER STUDY

To evaluate our techniques also from a user perspective, we conducted a user study using the CrowdFlower system. We designed two surveys in which a user is assigned to a certain evaluation task, called HIT. In each HIT, a number of questions on the result quality had to be answered. We allowed a maximum number of 10 users for each HIT and finally count all user responses to determine a trend in the result perception.

For this experiment, we designed HITS that ask users to compare two recommendations of items for 100 random groups. A first list (*utility-only*) is built by selecting the items according to their utility scores. A second list (diversified group recommendation, DGR) contains the items selected by our technique. Then, we built a HIT for each group (so there are 100 HITS in total) that comprises two questions. First, we asked users to rate the diversity of the DGR list against the baseline by five choices: from (1) *highly less diverse* to (5) *highly more diverse*. In the second question, we asked users which of the lists they prefer. we further considered only cases in which the number of identical items in the two lists is less than 70% to prevent users from being confused with close to identical list.

For the first question on the diversity of the lists of recommended items, the percentages of user answers are shown in Fig. 6. We observe that 55.73% of the users answered that the recommendation derived with our technique is *highly* (15.72%) or *slightly* (40.01%) more diverse; whereas only few users considered it to be *slightly* (14.02%) or *highly* (3.08%) less diverse than the baseline. This confirms that our technique is sound and indeed increases diversity of group recommendation. As illustrated in Fig. 7, 79.09% of the users prefer the DGR list over the baseline, which highlights the importance of diversification for satisfying the true preference intent of group members and suggests that our recommendation technique helps to achieve it.

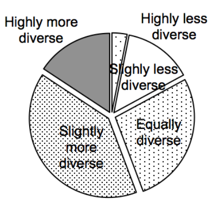

FIGURE 6. Diversity of recommendation.

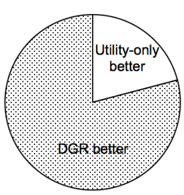

FIGURE 7. Quality of recommendation.

VII. CONCLUSIONS

In this paper, we developed a framework for diversifying group recommendation. We followed a two-step approach: firstly computing a group utility function, then optimizing the diversification problem as bi-objective criteria. While the former preserves history information about group member preferences, the latter increases the chance of returning at least one recommendation that matches the true intention of group. Given that diversification problem turns out to be NP-hard, we proposed a greedy diversification mechanism that achieves bounded optimality and a scalable complexity running time. Experiments on real datasets show the efficiency and effectiveness of our approach. The recommendation runs fast and linearly, achieving less than one second interaction practicality. Compared to other baselines, our approach overcomes the trade-off between diversity and utility by being 1.4 and 1.6 times better respectively.

While our work is orthogonal to a broad range of literature in group recommendation and result diversification, further improvements can be incorporated such as parallelization, optimization of similarity computation, and other formulations of group utility functions and diversifications tailored to specific data domains.

REFERENCES

[1] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in *Recommender Systems Handbook*. New York, NY, USA: Springer, 2011, pp. 1–35.
[2] S. B. Roy, S. Thirumuruganathan, S. Amer-Yahia, G. Das, and C. Yu, "Exploiting group recommendation functions for flexible preferences," in *Proc. IEEE 30th Int. Conf. Data Eng. (ICDE)*, Mar./Apr. 2014, pp. 412–423.
[3] G. Adomavicius and 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.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.
[4] M. Deshpande and G. Karypis, "Item-based top-n recommendation algorithms," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 143–177, 2004.
[5] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2011, pp. 448–456.
[6] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proc. 10th Int. Conf. World Wide Web*, 2001, pp. 285–295.
[7] H. Liu, P. Maes, and G. Davenport, "Unraveling the taste fabric of social networks," *Int. J. Semantic Web Inf. Syst.*, vol. 2, no. 1, pp. 42–71, 2008.
[8] B. Mobasher, J. Srivastava, and R. Cooley, "Automatic personalization based on Web usage mining," *Commun. ACM*, vol. 43, no. 8, pp. 142–151, Aug. 2000.
[9] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Comput.*, vol. 7, no. 1, pp. 76–80, Jan./Feb. 2003.
[10] A. S. Das, M. Datar, A. Garg, and S. Rajaram, "Google news personalization: Scalable online collaborative filtering," in *Proc. 16th Int. Conf. World Wide Web*, 2007, pp. 271–280.

[11] J. Bennett and S. Lanning, "The netflix prize," in *Proc. KDD Cup Workshop*, New York, NY, USA, 2007, p. 35.
[12] M.-H. Hsu, "A personalized English learning recommender system for ESL students," *Expert Syst. Appl.*, vol. 34, no. 1, pp. 683–688, 2008.
[13] J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang, "Recommender system application developments: A survey," *Decision Support Syst.*, vol. 74, pp. 12–32, Jun. 2015.
[14] S. Amer-Yahia, S. B. Roy, A. Chawlat, G. Das, and C. Yu, "Group recommendation: Semantics and efficiency," *VLDB Endowment*, vol. 2, no. 1, pp. 754–765, 2009.
[15] L. Boratto, S. Carta, A. Chessa, M. Agelli, and M. L. Clemente, "Group recommendation with automatic identification of users communities," in *Proc. IEEE/WIC/ACM Int. Joint Conf. Web Intell. Agent Technol. (WI-IAT)*, Sep. 2009, pp. 547–550.
[16] L. M. de Campos, J. M. Fernández-Luna, J. F. Huete, and M. A. Rueda-Morales, "Managing uncertainty in group recommending processes," *User Model. User-Adapted Interact.*, vol. 19, no. 3, pp. 207–242, 2009.
[17] A. Jameson and B. Smyth, "Recommendation to groups," in *The Adaptive Web*. Berlin, Germany: Springer, 2007, pp. 596–627.
[18] J. Masthoff, "Group recommender systems: Combining individual models," in *Recommender Systems Handbook*. Boston, MA, USA: Springer, 2011, pp. 677–702.
[19] D. Sprague, F. Wu, and M. Tory, "Music selection using the partyvote democratic jukebox," in *Proc. Work. Conf. Adv. Vis. Interfaces*, 2008, pp. 433–436.
[20] E. Vildjiounaite, V. Kyllönen, T. Hannula, and P. Alahuhta, "Unobtrusive dynamic modelling of tv programme preferences in a finnish household," *Multimedia Syst.*, vol. 15, no. 3, pp. 143–157, 2009.
[21] M. O'Connor, D. Cosley, J. A. Konstan, and J. Riedl, "PolyLens: A recommender system for groups of users," in *Proc. ECSCW*, 2001, pp. 199–218.
[22] J. F. McCarthy and T. D. Anagnost, "MusicFX: An arbiter of group preferences for computer supported collaborative workouts," in *Proc. ACM Conf. Comput. Supported Cooperat. Work*, 1998, pp. 363–372.
[23] Z. Yu, X. Zhou, Y. Hao, and J. Gu, "Tv program recommendation for multiple viewers based on user profile merging," *User Model. User-Adapted Interact.*, vol. 16, no. 1, pp. 63–82, 2006.
[24] S. Gollapudi and A. Sharma, "An axiomatic approach for result diversification," in *Proc. 18th Int. Conf. World Wide Web*, 2009, pp. 381–390.
[25] T. Deng and W. Fan, "On the complexity of query result diversification," *VLDB Endowment*, vol. 6, no. 8, pp. 577–588, 2013.
[26] C. Yu, L. Lakshmanan, and S. Amer-Yahia, "It takes variety to make a world: Diversification in recommender systems," in *Proc. EDBT*, 2009, pp. 368–378.
[27] A. Jain, P. Sarda, and J. R. Haritsa, "Providing diversity in k-nearest neighbor query results," in *Proc. PAKDD*, 2004, pp. 404–413.
[28] J. Carbonell and J. Goldstein, "The use of MMR, diversity-based reranking for reordering documents and producing summaries," in *Proc. SIGIR*, 1998, pp. 335–336.
[29] B. Zhang *et al.*, "Improving Web search results using affinity graph," in *Proc. SIGIR*, 2005, pp. 504–511.
[30] X. Zhu, A. B. Goldberg, J. Van Gael, and D. Andrzejewski, "Improving diversity in ranking using absorbing random walks," in *Proc. NAACL*, 2007, pp. 97–104.
[31] D. C. Thang, N. T. Tam, N. Q. V. Hung, and K. Aberer, "An evaluation of diversification techniques," in *Proc. Int. Conf. Database Expert Syst. Appl.*, 2015, pp. 215–231.
[32] A. Jsang and R. Ismail, "The beta reputation system," in *Proc. BECC*, 2002, pp. 41–55.
[33] M. E. Renda and U. Straccia, "Web metasearch: Rank vs. score based rank aggregation methods," in *Proc. SAC*, 2003, pp. 841–846.
[34] J. H. Lee, "Analyses of multiple evidence combination," in *Proc. SIGIR*, 1997, pp. 267–276.
[35] P. Diaconis, *Group Representations in Probability and Statistics* (Lecture Notes–Monograph Series). USA: JSTOR, 1988, pp. 1–192.
[36] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, "Rank aggregation methods for the Web," in *Proc. WWW*, 2001, pp. 613–622
[37] H. P. Young and A. Levenglick, "A consistent extension of condorcet's election principle," *SIAM J. Appl. Math.*, vol. 35, no. 2, pp. 285–300, 1978.
[38] R. A. Bradley and M. E. Terry, "Rank analysis of incomplete block designs: I. The method of paired comparisons," *Biometrika*, vol. 39, nos. 3–4, pp. 324–345, 1952.
[39] L. L. Thurstone, "The method of paired comparisons for social values," *J. Abnormal Social Psychol.*, vol. 1, no. 4, pp. 384–400, 1927.
[40] U. Feige, D. Peleg, and G. Kortsarz, "The dense k-subgraph problem," *Algorithmica*, vol. 29, no. 3, pp. 410–421, 2001.
[41] G. Nemhauser, L. Wolsey, and M. Fisher, "An analysis of approximations for maximizing submodular set functions–I," *Math. Programm.*, vol. 14, no. 1, pp. 265–294, 1978.
[42] H. Wang, Y. Lu, and C. Zhai, "Latent aspect rating analysis without aspect keyword supervision," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2011, pp. 618–626.
[43] H. Wang, Y. Lu, and C. Zhai, "Latent aspect rating analysis on review text data: A rating regression approach," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2010, pp. 783–792.
[44] M. Drosou and E. Pitoura, "Disc diversity: Result diversification based on dissimilarity and coverage," in *Proc. PVLDB*, 2012, pp. 13–24.

NGUYEN THANH TOAN is currently pursuing the bachelor's degree with Ho Chi Minh City University of Technology. His main research interests include the areas of database systems, machine learning, decision support systems, and software design.

PHAN THANH CONG received the bachelor's degree in computer science from Ho Chi Minh City University of Technology. His research interests include artificial intelligence, data mining, recommender systems, and big data analytics.

NGUYEN THANH TAM received the B.Sc. degree in computer science from Ho Chi Minh City University of Technology and the master's degree from École Polytechnique Fédérale de Lausanne, Switzerland, where he is currently pursuing the Ph.D. degree. His research interests include database technology, information theory, and machine learning. He has authored several papers in international journals and international referred conferences, such as ICDE, IJCAI, TKDE, JVLDB, and SIGIR.

NGUYEN QUOC VIET HUNG received the master's and Ph.D. degrees from École Polytechnique Fédérale de Lausanne, Switzerland. His research focuses on data integration, data quality, information retrieval, trust management, recommender systems, machine learning, and big data visualization, with special emphasis on web data, social data, and sensor data. He is currently a Lecturer with Griffith University. He has authored or co-authored several papers in top-tier venues, such as SIGMOD, SIGIR, ICDE, IJCAI, JVLDB, and TKDE.

BELA STANTIC is currently a Professor with the School of Information and Communication Technology, Griffith University. His research interests include the efficient management of complex data structures, including big data, spatio-temporal, and high dimensional data. He successfully applied his interdisciplinary and published over 90 peer-reviewed conference and journal papers. He presented many invited and keynotes talks and served on program committees of over 100 conferences and was/is doing the editorial duties for many journals.