DineTogether: A Social-Aware Group Sequential Recommender System

Zheyu Chen, Anqi Hu, Jie Xu, Chi Harold Liu

School of Software, Beijing Institute of Technology, Beijing, China
`Email:chiliu@bit.edu.cn`

Abstract. Group recommendations become important in many practical scenarios, e.g., couples would like to share movies together, families dine together in a restaurant, a group of friends plan to spend vacation in some points of interest. Although some previous research efforts have been done in this area, most of them only consider a small portion of contextual factors but ignore the time series features of the user historical behavior and next recommended location/event to do, that makes the system performance not as satisfactory as expected. Here, we propose a novel group sequential recommender system, called "DineTogether", for a group of people dinning together. It is able to capture comprehensive contextual, social factors when make recommendations. We design a computational model, called "Social-and-Time-Aware (STA)" model, and a novel algorithm, Generalized Random Walk with Restart (GRWR). Experiment results show that our approach outperforms the state-of-the-art group recommendation approaches.

Key words: Group recommender system, random walk with restart, social aware model

1 Introduction

The emergence of social applications makes an increasing number of online activities to be performed in groups, and thus we are witnessing a trend in developing techniques for making recommendations for socially acquainted individuals to help them decide on a suitable way to perform group activities. Existing research in this area have been designed to meet different application requirements, e.g., travel [1], healthcare [2, 3, 4], and crowd funding [5]. However, most of them lack a comprehensive consideration for the factors that impact the generation of group recommendation.

Different from personal recommendation, group recommendation can be more complicated due to the influence of many contextual factors, which are implicit (e.g., location), but still have impact on users' opinions and behaviors. Besides, other important features that affect the overall system performance include, and will be the focus of this paper, are social relationship and time series. The former can impact users' selection, since they are consciously or unconsciously influenced by opinions and preference of other members [6]. Furthermore, time

TRIDENTCOM 2017, September 28-29, Dalian, People's Republic of China
Copyright © 2018 EAI
DOI 10.4108/eai.8-1-2018.155564

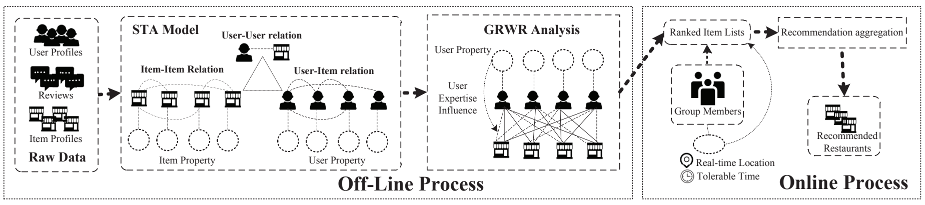

Fig. 1. Architecture of the proposed DineTogether system.

series is also quite importance, since an item's popularity changes when time progresses, as observed in [7] that most consumers are influenced by how widely a product is exposed in the market.

The most significant difference between group and individual recommendation is that the former has to deal with the disagreements among group members. According to [8], there are mainly two factors in forming user groups: group size and cohesiveness. Group size can be large, median, and small. Group cohesiveness reflects the similarity among each group member. The higher the cohesiveness a group has, the more similar the tastes of group members share. Existing research efforts along this direction can be divided into two types [9], (a) recommendation aggregation (RA), and (b) preference aggregation (PA). RA first uses a general recommendation algorithm for every individual to generate a set of individual recommendation lists, and then aggregates these lists into one single list for the whole group. Many rank aggregation methods can be used to handle the aggregation problem in RA, such as the Spearman Footrule Aggregation, the Kemeny Optimal Aggregation, Borda Count Aggregation, Average Aggregation, and Least Misery Strategy, etc [10]. On the other hand, PA first aggregates the users' individual preference model into a united model to create a virtual user that represents the whole group, and then preforms traditional recommendation on the virtual user. For example, The Pocket Restaurant Finder pools the preference of a group of people and presents a list of potential restaurants using a content-based algorithm [11]. Although different approaches have been proposed to aggregate the members' preferences, there is still no consensus about the optimal solution [12].

Towards, this end, in this paper, we design a recommender system named DineTogether that is able to track the social and temporal factors for group sequential recommendations. Specifically, we propose a novel computational model and a corresponding graph-based algorithm. The model, called "social-and-time aware (STA)" model, is an extension of traditional Markove Chain model that incorporates three types of relationships in one single framework, i.e., user-user (UU), user-item (UI), and item-item relations. We also propose an algorithm called "Generalized Random Walk with Restart" (GRWR), as an optimized version of traditional RWR algorithm, for analyzing STA model with explicit consideration of users' behaviors. Finally, based on real dataset of Yelp, experiment results shows that our proposal provides recommendations with higher accuracy.

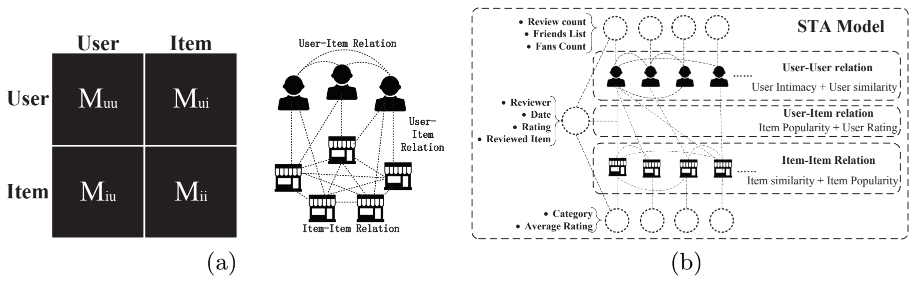

Fig. 2. (a) Proposed STA Model, and (b) detailed STA Model.

The rest of the paper is organized as follows. Section 2 describe the overall system architecture of DineTogether. Section 3 presents the proposed social-and-time aware model, and Section 4 describes the proposed optimized random walk with restart model GRWR for missing rate predictions. Section 5 describes the overall online processing workflow, and Section 6 shows the experimental results. Finally, Section 7 concludes the paper.

2 System Architecture

In this section, we briefly introduce the architecture of our group recommender system DineTogether. It aims to provide a suitable list of restaurants for groups who want to have meals together, and able to capture the key contextual factors including social influence, item popularity and locations. System architecture is shown in Fig. 1, of two key modules: online process model and offline process module.

For the latter, the input raw data are user and item profiles including reviews and implicit information of users' social relationships and item popularity. Then, it will be processed by STA Model (see Section 3) to extract implicit information, as three types of relations: UI, UU, and II. Then, we perform GRWR (see Section 4) by setting restart probability according to user expertise, so that the whole iterative process is influenced by users expertise index. After, we obtain the output as a set of ranked list of places for all group members. Online process model help users' real-time location and time tolerance can be accessed by the system. The original ranked lists generated by offline module will be filtered. Take location for example, we filter out those restaurants remotely located from group members or cost intolerable time to reach, as then final recommendation decision is made.

3 Proposed STA Model

Our proposed STA model is an extension of the traditional Markov Chain model, where the latter is usually used to predict the users' missing ratings on the user-item bipartite graph. In our work, we propose a computational model, to contain not only UI relation, but also UU and II relations, as shown in Fig. 2(a). These three relations simultaneously capture the social and temporal (i.e., item popularity) influences, as shown in Fig. 2(b) for details.

UI relation: it is an indispensable feature used in every recommender system.In our work, instead of using simple ratings directly, we combine with item popularity to determine the weight between user and item. Customers are easily influenced by other's behaviors, and this phenomenon will be more common when it comes to a group of people choosing items to share. Since they have different preferences and opinions, to avoid confliction, they usually tend to come up with a more “low-risk” and “fair” solution, to choose the most popular one. Thus, we consider that, users are more likely to move to the item with higher popularity, and the weight between user and item can be defined as:

$$p(U_i, I_j) = \alpha * f_{\text{rating}}(R_{ij}) + (1-\alpha) * f_{\text{review}}(I_j), \tag{1}$$

where $\alpha \in [0,1]$. $p(U_i, I_j)$ denotes the weight between U_i and I_j nodes in a graph. Mapping function $f_{\text{rating}}(R_{ij})$ represents the normalized rating from U_i to I_j, and $f_{\text{review}}(I_j)$ represents the normalized popularity for I_j, as:

$$f_{\text{rating}}(R_{ij}) = \frac{1}{2} + \frac{(r_{ij} - \text{AvgR}(U_i))}{\max(R)}, \quad f_{\text{review}}(I_j) = \frac{1}{2} + \frac{(v_j - \text{AvgV}(I_j))}{\max(V)}, \tag{2}$$

where r_{ij} is the score U_i rates I_j, and $\text{Avg}(U_i)$ denotes the average for all items, while $\max(R)$ is full score. v_j represents the I_j's total review count in most recent week. Likewise, $\text{AvgV}(I_j)$ is the average review count of all items.

UU relation: it is defined as the social relationships between users, and has been used early in many recommender. Most research efforts consider it as the similarity among each user. For example, Memory-Based Collaborative Filtering utilizes the similarity of users to generate recommendations [13]. In this paper, we consider more than just user similarity but also user intimacy to determine the UU relation, as:

$$p(U_i, U_j) = \beta * \text{userSim}(U_i, U_j) + (1-\beta) * \text{intimacy}(U_i, U_j), \tag{3}$$

where $\beta \in [0,1]$. $p(U_i, U_j)$ denotes the transition probability between U_i and U_j. The $\text{Intimacy}(U_i, U_j)$, $\text{userSim}(U_i, U_j)$ represent the intimacy and preference similarity, respectively. Here, preference similarity can be calculated as below:

$$\text{userSim}(U_i, U_j) = \frac{\mathbf{R}_{U_i} + \mathbf{R}_{U_j}}{|\mathbf{R}|_{U_i} |\mathbf{R}|_{U_j}}, \tag{4}$$

where $\mathbf{R}_{U_i}$ denotes the U_i's rating list with each element as the score U_i gives to a certain item. If U_i has never rated certain item, the corresponding element is set 0. Furthermore, $\text{Intimacy}(U_i, U_j)$ is determined by the mutual friend relation:

$$\text{intimacy}(U_i, U_j) = \begin{cases} 0, & \text{mutualFriend}(U_i, U_j) = \text{true} \\ 1, & \text{mutualFriend}(U_i, U_j) = \text{false}. \end{cases} \tag{5}$$

Item-Item relation: many existing research simply sets the II relation to the similarity value, while the core principle of a MC model is that one user has certain possibility to transit from/to any neighboring node in a graph. That is, if a user arrives at item i, then the probability with which she travels to item j should be balanced by the similarity between I_i and I_j, and the I_j's current popularity. We define the II relation as:

$$p(I_i, I_j) = \gamma * \text{itemSim}(I_i, I_j) + (1-\gamma) * \text{ItemPop}(I_j),$$
$$\text{itemSim}(I_i, I_j) = \frac{\mathbf{C}_{I_i} * \mathbf{C}_{I_j}}{|\mathbf{C}|_{I_i} |\mathbf{C}|_{I_j}},$$

and,

$$\mathbf{C}_{I_j}[k] = \begin{cases} 0, & I_j \in \text{Category}[k] \\ 1, & I_j \notin \text{Category}[k] \end{cases} \tag{6}$$

where $\gamma \in [0,1]$. $p(I_i, I_j)$ is the transition probability from I_i to I_j. itemSim(I_i, I_j) is the similarity between I_i and I_j. ItemPop(I_j) is the popularity of I_j as discussed above, and $\mathbf{C}_{I_i}$ is the category list of I_i, an m-dimensional vector with each element corresponding to a type of items. If I_i belongs to k^{th} type, then $\mathbf{C}_{I_i}[k]$ is set 1, otherwise 0.

4 Generalized Random Walk with Restart

Random Walk with Restart model is able to provide a good relevant score between two nodes in a weighted graph, and widely adopted in all kinds of domains, including ecology, psychology, economics, as well as computer science. One of the best known algorithm, PageRank, is an classic instance of RWR. RWR can also be used for group recommendation, however it has to be run for every individual in a group, that takes long time to reach steady state. [14] provides a state-of-art solution for this issue, that only run RWR once no matter how large a group is and is proved to save $(N-1)T$ running time with the Average Aggregation strategy (where N denotes the number of group members, and T is the running time of each user). The optimized RWR algorithm in [14] sets all the related elements in $\mathbf{e}$ to be 1 and others 0. Instead of using $\mathbf{r_i}$ to represent one user state, it uses matrix R_i to represent it at one time, as:

$$R_{i+1} = c\tilde{M}R_i + (1-c)E, \quad i = 1, 2....k, \tag{7}$$

where $R_i = [\mathbf{r_i}^\mathbf{1}, \mathbf{r_i}^\mathbf{2}, ..., \mathbf{r_i}^\mathbf{m}]$, and $E = [\mathbf{e}^\mathbf{1}, \mathbf{e}^\mathbf{2}, ..., \mathbf{e}^\mathbf{m}]$. If $I - c\tilde{M}$ is nonsingular, i.e., $R_n = (1-c)(I - c\tilde{M})^{-1}\mathbf{E}$, by optimizing the traditional RWR, [14] successfully adopted it for group recommendation. However, their solution cannot meet our requirement to integrate social and temporal factors into the computing process. In this paper, we propose a new algorithm based on [14], that captures the user expertise influence for group recommendation, called "Generalized Random Walk with Restart" (GRWR).

Consider m denotes the number of users, k is the iteration step, I is a m-by-m unit matrix, then GWRW is presented as:

$$\hat{R_{i+1}} = \tilde{M}\hat{R}_i B + E(I-B), i = 1, 2....k. \tag{8}$$

Compared to (7), (8) replaces the transition probability c with a n-by-n diagonal matrix B, and $(1-c)$ with $(I-B)$. B is shown as:

$$B = \{b_{ij}\}, \quad \forall i, j \in [1, m], \quad b_{ij} = \begin{cases} y, & y \in (0,1), i = j \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

To obtain the final steady matrix R, we solve the following linear equations to have $R \in \mathbb{R}^{n\times m}$, and $R = MRB + E(I-B)$, where $M \in \mathbb{R}^{n\times n}$, $E \in \mathbb{R}^{n\times m}$, $B = \mathrm{diag}(\beta_1, \beta_2, ..., \beta_m)$ and $I = I_m$ is a m-by-m identity matrix. All matrices except R are known as a priori. Next, look at each column of R individually and let $R = [r_1, r_2, ..., r_m]$, $r_i \in \mathbb{R}^{n\times 1}, \forall i \in [1, m]$. Note that right-multiplying R by a diagonal matrix B is equivalent to scale the i-th column by a factor of b_i. Thus we have:

$$RB = [\beta_1 r_1, \beta_2 r_2, ..., \beta_m r_m]. \tag{10}$$

Then, let matrix $C = E(1-B) = [c_1, c_2, ..., c_m] \in \mathbb{R}^{n\times m}$. The linear equation can be reformulated as:

$$[r_1, r_2, ..., r_m] = M[\beta_1 r_1, \beta_2 r_2, ..., \beta_m r_m] + [c_1, c_2, ..., c_m], \tag{11}$$

where $r_i = \beta_i M r_i + c_i, \forall i \in [1, m]$. Then, each r_i can be obtained by solving the above linear equation, as: $r_i = (I - \beta_i M)^{-1} c_i$.

If $(I - \beta_i M)$ is non-singular, from (8), we can also infer that when $b_{11} = b_{22} = ... = b_{mm} = c$, then:

$$\begin{aligned} \hat{R_{i+1}} &= \tilde{M}\hat{R}_i B + E(I-B) \\ &= \tilde{M}\left[b_{11}\mathbf{r^1}\ b_{22}\mathbf{r^2}\ ...\ b_{mm}\mathbf{r^m}\right] + \left[(1-b_{11})\mathbf{e^1}\ (1-b_{22})\mathbf{e^2}\ ...\ (1-c)b_{mm}\mathbf{e^m}\right] \\ &= \tilde{M}\left[c\mathbf{r^1}\ c\mathbf{r^2}\ ...\ c\mathbf{r^m}\right] + \left[(1-c)\mathbf{e^1}\ (1-c)\mathbf{e^2}\ ...\ (1-c)\mathbf{e^m}\right] \\ &= c\tilde{M}\left[\mathbf{r^1}\ \mathbf{r^2}\ ...\ \mathbf{r^m}\right] + (1-c)\left[\mathbf{e^1}\ \mathbf{e^2}\ ...\ \mathbf{e^m}\right] \\ &= c\tilde{M}R_i + (1-c)E = R_{i+1}. \end{aligned} \tag{12}$$

We can see that (7) and (8) are identical, with the only difference is that in (7), every user has the same transition probability.

In GRWR, we assume that a user's restart probability differs from each other because of the influence of her own situation. Thus, let a_{ii} represent only $user_i$'s transition probability and $1 - a_{ii}$ denote its restart probability. Whether a user will return to the initial state or not depends on her choice of selecting items. This feature can be measured by a user's number of followers and the review counts. We have: $a_{ii} = \sigma(f_{\exp}(U_i))$, where

$$f_{\exp}(U_i) = \frac{n * rv(U_i)\mathrm{fans}(U_i) - \sum_{n}^{k=1} rv(U_k)\mathrm{fans}(U_k)}{\sum_{n}^{k=1} rv(U_k)\mathrm{fans}(U_k)} \tag{13}$$

Here $rv(U_i)$ represents the number of U_i's total reviews. fans(U_i) is the number of U_i's total followers. Function $\sigma(x)$ is a sigmoid function to normalize the user experience index, as: $\sigma(x) = \frac{1}{1+e^{-x}}$.

5 Online Processing

The input of the online processing module is a set of ranked lists for each user. Since the dataset we used in this paper is from Yelp, it is impossible and unpractical to recommend some point-of-interests that too far away to users. Based on a user's real-time location and time tolerance, we filter the input ranked lists to make sure all the items in the set of ranked lists has suitable location for all group members. After, we apply the Average Aggregation method on the new lists to generate the final recommendation for a group, to satisfy the requirement of considering all the key factors that will influence its quality, including time, location and social relationship.

6 Performance Evaluation

In this section, we evaluate the performance of proposed DineTogether system by using the real world dataset from Yelp, and report the experimental results and analysis.

6.1 Setting

In the original dataset from Yelp, there is no profile of groups. Therefore, we need to generate groups. It is known that there are mainly two factors in forming user groups: group size and cohesiveness. The former represents the number of members in a group, and we divided group size into ascending levels, i.e., from 2 people per group to 60 people per group. The latter represents if members share the same preferences for different restaurants, which can also be defined as the similarity between group members. In the experiment, member similarity is calculated by (4). The analysis of our dataset shows that the users' bi-similarity falls within the range of [0.092, 0.668], with average 0.305 and median 0.356. Thus, we next evaluate the effectiveness of the proposed recommender system in different group cohesiveness varying from 0.1 to 0.7.

6.2 Evaluation Metrics and Comparable Approaches

To better reflect the real-life scenarios, we chronologically assigned the 60% of the dataset as the training data and the most recent 40% as the test data. To evaluate the effectiveness of our proposed group recommender system, we adopted the standard measurement, recall, precision, and normalized discounted cumulative gain (nDCG) as the evaluation metrics. Precision measures the fraction of relevant instances among the retrieved instances. In this experiment, it is defined as the ratio of the recommended restaurant, that are visited by users in the test set, to all the restaurants in the recommendation list, as:

$$\text{Precision} = \frac{\sum_{u \in \mathcal{U}} N_{\text{inList}}(u) \bigcap N_{\text{Relevant}}(u)}{\sum_{u \in \mathcal{U}} N_{\text{inList}}(u)}, \tag{14}$$

where $\mathcal{U}$ is the collection of all users. N_{inList} denotes those restaurants in the recommendation list, N_{Relevant} denotes the visited restaurants in test data. Similarly, recall can be calculated as:

$$\text{Recall} = \frac{\sum_{u \in \mathcal{U}} N_{\text{inList}}(u) \bigcap N_{\text{Relevant}}(u)}{\sum_{u \in \mathcal{U}} N_{\text{Relevant}}(u)}. \tag{15}$$

Different from single item recommendation that do not distinguish the ranking of recommended items, results of our recommender system is a ranked list of items. Thus, standard measurements precision and recall cannot fully satisfy the requirement of evaluating ranked recommended list. We adopted nDCG, as an evaluation metric for our recommendation result.

As reported in [10, 15, 16], different RA methods show similar performance on efficiency and accuracy. Thus, we simply adopted the most commonly used RA strategy, Average Aggregation method, to aggregate the recommendation list generated by different recommendation algorithms. In this experiment, we compared our approach with the following methods:

– User-Based Collaborative Filtering - Average Aggregation (UBCF-AA)
– Item-Based Collaborative Filtering - Average Aggregation (IBCF-AA)
– SVD Method - Average Aggregation (SVD-AA) [17]
– Traditional Random Walk with Restart - AA (TRWR-AA)

6.3 Results and Analysis

Fig. 3(a) shows how the group size influences the group recommendation quality. In this experiment, we set the the group cohesiveness to be 0.31 that is around the average group cohesiveness value, and the size of ranked recommended list to be 10. We can see that, with the increase of group size, precision seems to increase first and then fall down to a certain point, as well as the nDCG value (i.e., around the point of group size 10), while the recall value generally has a tend to increase. This can be explained as after group size is bigger than 12, more members, more dissimilarity and conflicts among them. For recall, we assume that, in bigger groups, the system has greater chance to find the items that are liked by group members. Therefore, its value will increase when group size becomes larger. The most suitable group size in which the DineTogether system has the best performance is around 8-12 people per group.

Next, we investigate the impact of the size of recommended items. The results are showed in Fig. 3(b). We can see that with the increase of the size of recommended list, nDCG value does not show a clear trend of increasing or decreasing, while both precision and recall tend to keep growing. It is is more obvious when the size is smaller than 15. After, precision and recall maintain a vary slow rate of growth. Therefore, we summarize that it is more reasonable and efficient to recommend around 10 to 15 items for group users.

Fig. 3(c) shows the relationship between group cohesiveness and recommendation performance. In this experiment, we set the group size to be 10 and the size of ranked recommended list to be 10. We observe that no matter it is precision, recall or nDCG, the higher the similarity among group members, the better the performance of our system gains, as consistent to what we expect.

Finally, Fig. 3(d) shows the performance of different group recommender approaches. Here we set parameters identical for all of them, as: group cohesiveness to be 0.654, group size to be 10, and size of recommended list to be 10. From the figure, we observe that our approach generally has better performance than others, since the obtained precision, recall and nDCG values are generally higher. This is because, unlike other approaches, our system takes social influences and time into consideration when building the STA Model and performing the GRWR.

7 Conclusion

In this paper, we build a social-and-time-aware group recommender system, called DineTogether, for people who plan to have meals together. We proposed an STA model and GRWR algorithm for capturing the social and temporal impact. Experiment results on real dataset Yelp show that our approach is more effective and accurate if compared with the state-of-the-art group recommendation approaches.

Acknowledgement

This work is financially supported by National Natural Science Foundation of China (No. 61772072), and in part by Young Elite Scientists Sponsorship Program by CAST, China.

References

1. H. Li, Y. Ge, and H. Zhu, “Point-of-interest recommendations: Learning potential check-ins from friends,” in *KDD*, 2016.
2. M. Chen, P. Zhou, and G. Fortino, “Emotion communication system,” *IEEE Access*, vol. 5, pp. 326–337, 2017.
3. M. Chen, Y. Ma, Y. Li, D. Wu, Y. Zhang, and C. Youn, “Wearable 2.0: Enable human-cloud integration in next generation healthcare system,” *IEEE Communications Magazine*, vol. 55, no. 1, pp. 54–61, Jan. 2017.

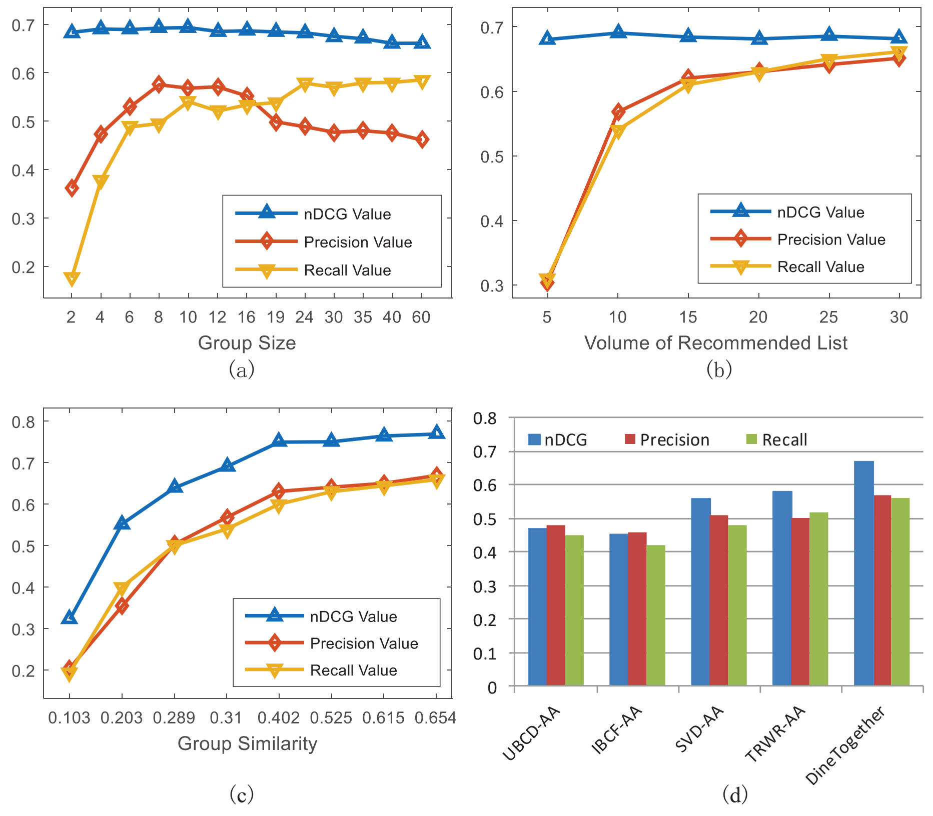

Fig. 3. Simulation results of (a) the impact of group size; (b) Top-N group recommendation. (c) the impact of group cohesiveness, and (d) comparison of different group recommendation approaches.

4. M. Chen, Y. Hao, K. Hwang, L. Wang, and L. Wang, “Disease prediction by machine learning over big healthcare data,” *IEEE Access*, vol. 5, no. 1, pp. 8869–8879, 2017.
5. V. Rakesh, W.-C. Lee, and C. K. Reddy, “Probabilistic group recommendation model for crowdfunding domains,” in *ACM WSDM*, 2016, pp. 257–266.
6. L. Quijano-Snchez, B. Daz-Agudo, and J. A. Recio-Garca, “Development of a group recommender application in a social network,” *Knowledge-Based Systems*, vol. 71, pp. 72–85, 2014.
7. P. Kotler, G. Armstrong, J. Saunders, and V. Wong, “Principles of marketing, 2nd edition,” *Corporate Communications*, no. 3, 2001.
8. S. Amer-Yahia, S. B. Roy, A. Chawlat, G. Das, and C. Yu, “Group recommendation: Semantics and efficiency,” *VLDB*, vol. 2, no. 1, pp. 754–765, 2009.
9. T. Pessemier, S. Dooms, and L. Martens, *Comparison of group recommendation algorithms*. Kluwer Academic Publishers, 2014.
10. L. Baltrunas, T. Makcinskas, and F. Ricci, “Group recommendations with rank aggregation and collaborative filtering,” in *ACM RecSys*, 2010, pp. 119–126.

11. J. F. Mccarthy, “Pocket restaurantfinder: A situated recommender system for groups,” in *Workshop on Mobile Ad-Hoc Communication at the 2002 ACM Conference on Human Factors in Computer Systems*, 2002.
12. J. Masthoff, “Group modeling: Selecting a sequence of television items to suit a group of viewers,” *User Modeling and User-Adapted Interaction*, vol. 14, no. 1, pp. 37–85, 2004.
13. G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin, *Incorporating contextual information in recommender systems using a multidimensional approach*, 2005, vol. 23, no. 1.
14. L. Guo, J. Shao, K. L. Tan, and Y. Yang, “Wheretogo: Personalized travel recommendation for individuals and groups,” in *IEEE MDM*, vol. 1, 2014, pp. 49–58.
15. A. Jameson and B. Smyth, “Recommendation to groups,” in *The Adaptive Web*, 2007, pp. 596–627.
16. C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, “Rank aggregation methods for the web,” in *ACM WWW*, 2001, pp. 613–622.
17. M. G. Vozalis and K. G. Margaritis, “Applying svd on item-based filtering,” in *IEEE ISDA'05*, 2005, pp. 464–469.