This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2879658, IEEE Transactions on Knowledge and Data Engineering

Dynamic Connection-based Social Group Recommendation

Dong Qin, Xiangmin Zhou, *Member, IEEE,* Lei Chen, *Member, IEEE,* Guangyan Huang, and Yanchun Zhang

Abstract—Group recommendation has become highly demanded when users communicate in the forms of group activities in online sharing communities. These group activities include student group study, family TV program watching, friends travel decision, etc. Existing group recommendation techniques mainly focus on the small user groups. However, online sharing communities have enabled group activities among thousands of users. Accordingly, recommendation over large groups has become urgent. In this paper, we propose a new framework to accomplish this goal by exploring the group interests and the connections between group users. We first divide a big group into different interest subgroups, each of which contains users closely connected with each other and sharing the similar interests. Then, for each interest subgroup, our framework exploits the connections between group users to collect a comparably compact potential candidate set of media-user pairs, on which the collaborative filtering is performed to generate an interest subgroup-based recommendation list. After that, a novel aggregation function is proposed to integrate the recommended media lists of all interest subgroups as the final group recommendation results. Extensive experiments have been conducted on two real social media datasets to demonstrate the effectiveness and efficiency of our proposed approach.

Index Terms—Group Recommendation, Social Item, Social Connection, Collaborative Filtering

1 Introduction

Recommender systems have proven their values in many domains such as movies, music and tourism for delivering products to proper users. A report [3] from Unruly Media in 2012 demonstrates 65% of viewers who enjoyed online videos recommended. The Amazon recommendation secret [17] reported a 29% sales increase to $12.83 billion during its second fiscal quarter in 2012, up from $9.9 billion during the same time in the previous year because of the integration of recommendation into purchasing process. Recent recommender systems combine the contexts, such as time, location, and the user connections into systems for high quality recommendation to individuals [23], [30]. However, the suggested items are not intended for personal usage in many circumstances, but rather for group consumption [5], [24], [25]. For example, in the activities such as music play in gyms, teaching material selection, and friends movie watching, the decision of a group is important. All these demand the development of effective and efficient group recommender systems.

For effective and efficient recommendation to a group of users, several key issues need to be addressed. First, the preferences of a group should be represented informatively and precisely. The extracted preferences represent users' interests and determine which items have high probability accepted by group members. The second issue is how to integrate multiple middle recommendation lists to generate the final results for the group. The middle recommendation lists are generated with respect to various sources such as different group members and contextual information. We need a consensus function to combine them and satisfy as many group users as possible. The third issue is how to conduct recommendation efficiently. Efficient recommendation techniques are necessary to avoid unnecessary candidate access and reduce time costs, which is curial for online communities.

Techniques have been developed for effective recommendation to small groups. A typical method is to construct a pseudo profile by extracting each member's preference and then aggregating the preferences of all members [7], [45], [55]. Thus the recommendation to groups is converted into that to individuals, where classic individual user recommendation approaches can be applied. The drawback of these group profile-based approaches lies in the difficulty in finding a pseudo profile to represent the preferences of various users in a group. For one thing, different contexts could trigger different user preferences. For another, in group scenarios, users are easily affected by others when making decisions. Thus, representing the whole group as a single profile is poorly qualified, if not impracticable. Another type of methods generates a middle list for each individual in a group and then merges the middle results for all members into the final recommendation for the group [4], [19], [51]. The result merging is accomplished by a predefined group consensus function that delivers the final recommended items accepted by more members no matter how they love them [35], or generates the final results highly praised by some users only [54]. Recommender systems have been developed for small groups consisting of about ten users, such as Let's Browse [27], PolyLens [35], GroupRecoPF [12], MFCF [36] etc. However, none of these techniques is applicable to big groups. In online platforms, intensive interactions happen in groups involving thousands of users. In Flickr, "pet central" attracts 2,026 pet lovers, while "Sport Photography" connects 4,738 photographer lovers. In Twitter, #WorldHealthDay involves 4,867 users, while #themasters has 78.3k users to share ideas. When recommending social media such as images and hashtags to big groups directly, the existing systems incur high time cost because of the huge number of users and items

- *D. Qin and X. Zhou are with the School of Science, RMIT University, Melbourne, Australia.*
  E-mail: {dong.qin, xiangmin.zhou}@rmit.edu.au.
- *L. Chen is with the School of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong.*
  E-mail: leichen@cs.ust.hk
- *G. Huang is with the School of Information Technology, Deakin University, Burwood, Australia*
  E-mail: guangyan.huang@deakin.edu.au
- *Y. Zhang is with the School of Engineering and Science, Victoria University, Melbourne, Australia*
  E-mail: Yanchun.Zhang@vu.edu.au

to process. Meanwhile, the group profile-based recommendation describes a big group as a representation at a coarse level that is not discriminative, while individual profile-based methods have shown a strong bias against users having distinctive preferences, thus the quality of recommendation cannot be guaranteed.

Motivated by the limitation of existing recommender systems, we propose a novel approach that exploits the user connections and the user interest dynamics for big group recommendation in shared communities. First, we divide a big group into several subgroups based on the users' connections. Then, for each subgroup, we select the most similar users and items from the whole community as its recommendation candidate dataset. After that, the recommendation for each subgroup is generated by performing the *singular value decomposition* (SVD) over its candidate dataset. The final results are generated by integrating the recommendations for all subgroups using a dynamically adjusted combination function. We summarize the contributions in this paper as follows:

1) We propose a common-interest user subgroup extraction approach that mines various group preferences based on the user and item interactions in a group.
2) We propose a candidate user-item selection method that exploits the weak connections between users to obtain a subset of the whole dataset. The SVD is performed over the small subset, which improves the efficiency of SVD process while reducing the effects of noisy rating.
3) We propose a novel aggregation function that dynamically generates the final results based on all subgroup-based recommendations. We conduct extensive experiments on two real datasets to prove the superiority of our approach.

The remainder of the paper is organized as follows. Section 2 reviews the related research on group recommendation. Section 3 describes the framework of our group recommendation. We present our proposed algorithms, common-interest user subgroup extraction, candidate set selection and recommendation aggregation in section 4. The effectiveness and efficiency are evaluated in section 5. Finally, we conclude the whole work in section 6.

2 Related Work

We review the existing work on two topics closely related to our work, including collaborative filtering and group recommendation.

2.1 Collaborative filtering

Collaborative filtering (CF) is an effective way of conducting recommendation based on items' ratings information instead of contents. Typically, memory-based CF is performed by three steps. First, users' preferences are represented as their ratings on items (e.g., books, clothes or videos) in the corresponding domain. Then the system matches the target user's preference against other users' and finds the most "similar" ones. The similarity metrics commonly exploited for matching include Cosine similarity [28], [46], Pearson correlation [31], [42] and probability-based similarity [8], [21]. Finally it recommends the items highly rated by the similar users while not yet rated by the target user.

Model-based CF has proved its high quality of recommendation. This type of approach trains certain prediction models over the user-item-ratings matrix of a training dataset by exploiting the data mining and machine learning algorithms that find its patterns for recommendation. Commonly used model examples include Bayesian [32], latent semantic model [14] etc. SVD-based CF is the most successful one [22], which represents both users and items as a vector of f factors inferred from the ratings patterns. Thus, the user-item similarities are modelled as the inner products in that space. SVD has good scalability because it converts the relationship of users and items into f factors which reduces the dimensionality of the user-item-ratings matrix derived from the dataset [43], [44]. However, with the increase of data size, matrix computation becomes time costly and produces a big f, which leads to low efficiency. In [58], a recommendation algorithm called ApproSVD is proposed to approximate SVD. It samples some rows of a user-item matrix, rescales each row by an appropriate factor to form a relatively smaller matrix on which the dimensionality reduction is conducted. ApproSVD achieves higher accuracy and efficiency compared with the standard SVD. In [59], Zhou et al. proposed the incremental ApproSVD, which exploits the singular values and singular vectors of the original matrix to recompute the SVD of the updated matrix. The incremental ApproSVD and the original ApproSVD are combined to achieve the high scalability and effectiveness.

In SVD-based approaches, data sparsity is an inevitable problem. As the number of items is much bigger than that of users, the rating proportion to items is usually small in recommender system. Due to the rating sparsity, the SVD-based solutions usually produce inaccurate recommendation results. Methods have been proposed to address this issue. The first type of approaches added context information to overcome the problems caused by data sparsity. In [37], free-formatted text tags are injected into the traditional SVD-based approach to enhance the system performance. In [20], a model named *Multiverse Recommendation* was proposed, in which any type of context is considered as an additional dimension in the tensor representation of the data and the High Order SVD-decomposition is used for factor extraction. Another type of approach overcomes the problem of rating sparsity by predicting the missing ratings. In [13], users' trust is incorporated to their neighbours to predict the missing ratings. In [29], both the local context information and the global preference of users are used to overcome sparsity. In [11], the suitable imputation source is utilised to approximate all missing values and exploit the underlying data correlation structures, thus the high quality SVD-based recommendation is achieved. Item similarity [49] and demographic data [50] have also been used to predict the missing ratings. These approaches have addressed the data sparsity problem to some extent. However, the social information of users and items has not been fully exploited in recommendation. It is important to mine the relationship between users and the items that they might be interested in for recommendation. It has been proved that the gather-together of similar items and users achieves high quality SVD-based recommendation [6], [52]. Thus, in this work we will take different social connection information to find similar users and items for the user-item matrix construction, generating a compact matrix on which the SVD-based CF is conducted.

2.2 Group recommendation

Most recommender systems make suggestions for individual users. However, in many circumstances, the selected items such as movies are not intended for personal usage but rather for consumption in groups. Recent work have claimed that the study of social relationship such as social trust and user similarity is a new way in studying group interest [41], [48], and reported

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2879658, IEEE Transactions on Knowledge and Data Engineering

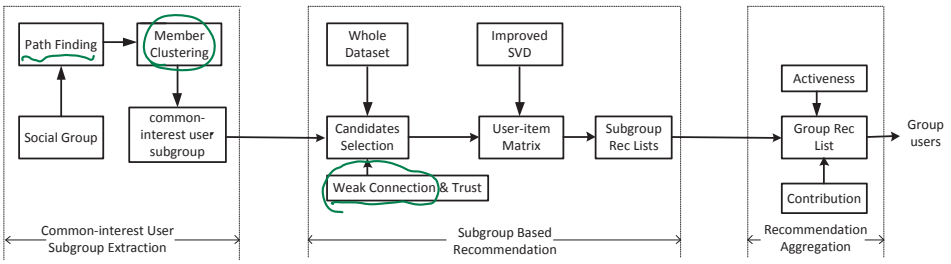


Fig. 1: Framework of the proposed group recommendation

significant recommendation performance improvement for group recommendation [18], [53]. In [10], Gartrell et al. combine both social and content information of group members to mine group interest. In [33], Ntoutsi et al. exploit hierarchical agglomerative clustering to divide all the users in the whole database into a number of clusters, each of which contains those having similar preferences. The collaborative filtering is conducted over the user clusters for efficient recommendation. In [34], Ntoutsi et al. first selects a subspace that contains the dominant attributes of items, and partitions all the items in the database over the subspace into a number of groups. All items in the same group belong to the same category with the similar attributes. Efficient recommendation is performed by the collaborative filtering over the item groups. In [38], a group recommendation model is implemented with various social factors and hierarchical relationships between users. In [39], the tie strength between users is utilised to develop a social group recommender system, which integrates the explanations about system recommendation and group social reality for a better understanding in the group recommendations.

Another main problem is to generate a recommendation that satisfies the biggest number of group members taking into account the individual preferences. Typical group recommender systems can be classified into two categories: (1) user profile aggregation-based; and (2) individual result aggregation-based. User profile aggregation-based methods focus on constructing an effective profile representation for several individuals in a group. In [7], tourism group profile is built based on the members' individual preferences and their social relationships. The probabilistic model COM [55] is proposed for the generative process of group activities. When making recommendations, COM estimates the preference of a group to an item by aggregating those of the group members with different weights, which considers members' topic-dependent influences and members' group behaviours. In [45], a group preference is treated as a combination of behavioural tendency such as content selection and the relationships among group users. Individual result aggregation-based approaches define a group satisfaction function for individual recommendation aggregation. Group satisfaction is normally measured by the least misery [35], most pleasure or average [54] in different scenarios. The average leads to recommendations biased toward users providing more ratings of contents or products, while the other methods support a small number of dissatisfied members only, ignoring the majority's preferences. To overcome their problems, some hybrid approaches were also proposed, which combined these three methods. In [19], a hybrid method based on the average and least misery strategies is proposed to aggregate the household member ratings. In [51], PageRank algorithm is utilised to analyse the social trust relationships between users as the factors that adjust the weight of different members during recommendation aggregation. Some methods model the resolution of disagreement among users for high performance. In [4], the authors first defined the group semantics to guide recommendation. The worthiness of an item for a group is estimated based on the balance between item's aggregation relevance to the group and the disagreement among group members. While recommending to big group is highly demanded on online platforms, all existing recommender systems focus on small groups. This paper focuses on the recommendation to big groups. The notations used in this paper are listed in Table 1.

TABLE 1: Notation

Notation	Meaning	Notation	Meaning
G	Social group	u	Social user
m	Social item	t	User-item interactions
$\mathcal{F}$	Influential factor	k	Subgroup number
ρ	Subgroup density	d	Group discussion
P	Path set	s	Path similarity
S	Subgroup	τ	User trust
$r(S)$	Interacted items of S	$u \sim S$	weak connection
C	Subgroup contribution	α	Interest activeness
g	Recommendation score		

3 Framework of our Solution

In this section we describe the framework of our proposed group recommendation system, followed by the problem formulation and the preliminary on heterogeneous information network.

3.1 Framework

In our dynamic connection-based group recommendation, the recommendation target is a group G. Given a group G, a set of social items I, and a group recommendation function g, our system delivers a list of media data with the top relevance scores to G. We address the problem of recommending social items to groups by taking account the interactions between users and items. The social group recommendation is formally defined as follows.

Definition 1. Given a social group G, a group relevance function g, social group recommendation automatically returns a list M which contains the top-K relevant items from the whole community, such that for any $m \in M$ and $m' \notin M$, $g(m, G) \geq g(m', G)$ holds.

In our approach, we conduct static recommendation to different subgroups, followed by dynamic aggregation of the subgroup recommendation results for the whole group. Our framework comprises three phases as shown in Figure 1.

The first phase is common-interest user subgroup extraction. Given a group users and their interacted social items, there exist rich interactions between users and items, such as commenting, uploading, favouring etc. We utilise these interactions to construct

a heterogeneous information network [47] and divide the whole group into the closely connected subgroups. In the second phase, the weak connections between each subgroup and users from the whole community are exploited to obtain the similar users and their interacted items as recommendation candidates. After that, singular value decomposition (SVD) is conducted on each candidate set to generate a subgroup-based recommendation list for each subgroup. In the third phase, all the generated lists are aggregated into the final one by a dynamic aggregation function considering the subgroup contribution and activeness.

Consider an example of a big group called *plant photo*. Users in this group are interested in various plant pictures, including *desert plant*, *water plant* and *arctic plant*. In our solution, the group users, group items, and the interactions between users and items are input into the subgroup extraction component of the system. By subgroup extraction, we can obtain three subgroups that contain *desert plant*, *water plant* and *arctic plant* respectively. Then, each subgroup and the whole database are passed to the subgroup-based recommendation component for the second stage processing. Suppose the current processed subgroup is *water plant*. All the photos in the database interacted with *water plant* subgroup are extracted as item candidates. The users interacted with the item candidates are selected as the user candidates from the whole community. Recommendations for *water plant* are generated based on the candidates of items and those of the users. The recommendation for each of other subgroups can be generated in the same way. All the recommendation lists for different subgroups are integrated as the final recommendation to the group *plant photo*.

3.2 Preliminary on heterogeneous information network

In this section, we describe the preliminary on the heterogeneous information network, which is used to find the similar users in social community in this work. Heterogeneous information network [47] is the logical network involving multiple typed objects and multiple typed links denoting different relations. Given a set of users and items, the heterogeneous information network can be constructed as a direct graph $G = (V, E)$, where V is the set of entity types and E is the set of relation types. In our work, there are two types of entities, user u and item m, and two types of relations, interaction t and being interacted t^{-1}. The relations include all the interactions between a user and an item such as commenting, viewing, sharing, favouring etc. We construct the heterogeneous information network by two steps. First, for each user $u \in V$, we check each item $m \in V$. If there exist interaction relationship, i.e. $u \xrightarrow{t} m$, we connect them and the direction is from u to v. Similarly, for each item $m \in V$, we check each user $u \in V$, if there exist being interaction relationship, i.e. $m \xrightarrow{t^{-1}} u$, we connect them and the direction is from m to u. In this way, a heterogeneous information network based on all users and items is constructed. Two users, u_a and u_b, can be connected by other users and items via a path $p : u_a \xrightarrow{t} m \xrightarrow{t^{-1}} u \xrightarrow{\cdots} u_b$. All the paths to all the user pairs in the set comprise a path set P over the information network.

4 Group Recommendation

In this section, we present the details of our proposed group recommendation framework. First, we extract several common-interest user subgroups by exploiting the relationships of members and items in a group. Then, for each subgroup, we search its similar users and their interacted items from the whole database as a candidate set on which the collaborative filtering is conducted to generate a recommendation list. Finally, a novel dynamic aggregation scheme integrates the recommendation lists for all subgroups to generate the final results for the group.

4.1 Common-interest user subgroup extraction

The traditional group recommendation methods operate on each user and then aggregate all the recommendation lists as the final results. When performing collaborative filtering individual-based recommendation, the system finds the similar users for each user and extracts the potential interest items from the user-item matrix constructed over the whole dataset. As we know, the volume of media data in online social networks is billions, the user-item matrix could be huge, thus the processing time on each user in a group is costly. When recommending to a big group with thousands of users, the efficiency issue becomes more serious. At the same time, a large number of users in a big group may have various interests, reflecting the group preferences from different aspects. Treating the whole group as a single group profile cannot capture the discriminative information on different user interests, while processing individual users separately ignores the interests of users who share less preference with other members, leading to inaccurate or biased recommendation results to the group users. Considering the high effectiveness and efficiency of our system, we conduct the group recommendation with respect to a small number of subgroups. The basic idea is to transform the operations for individuals into those for similar user sets, thus the number of operations over the whole dataset and the effect of interest bias can be greatly reduced.

In social networks, it is very common that big social groups do not have explicit subgroup tags. Take a group named outdoor activities in Flickr as an example. It could include several hidden subgroups such as hiking, fishing, diving etc. Intuitively, the closer the relationship between users is, the more common interests they share. Two users in the same subgroup usually connect more tightly than those from different subgroups. Thus, the user connections can be exploited to find different subgroups when no explicit subgroup tag provided. Existing subgroup extraction in shared communities only captures the direct user-item interactions [57]. However, the connection between two users is not always limited to the direct one in social community. For example, the connection of users A and B could be $A \rightarrow m_1 \rightarrow B$ and $A \rightarrow m_1 \rightarrow C \rightarrow m_2 \rightarrow B$, in which A, B, C are users and m_1, m_2 are items. It is necessary to exploit the connections to fully measure the similarity between social users. We propose a novel common-interest user subgroup extraction approach (CIUS), which exploits the social connection between users to divide a group into a number of user subgroups, each containing those with common interests. Our subgroup extraction includes two important parts: the user similarity measurement and the subgroup extraction.

User similarity We measure the similarity between users with the support of a heterogeneous information network over all users and their interacted items. Given a set of users and items, we construct a heterogeneous information network as shown in Section 3.2. There are two main categories to measure the similarity between users over a heterogeneous information network, the path-based similarity including *PathSim* [47], *path count* [40] and *random walk* [9], and the *tie strength* [26] based similarity.

To select a good function for our user similarity, we conduct a set of preliminary experiments on a data set D_p consisting of 50000 images and 2000 users to test the accuracy of different functions following [16]. D_p is a random subset of *Flickr* in Section 5. We invited 3 undergraduate students to give similarity between users as ground truth. The path-based similarity can be calculated directly from the built heterogeneous information network. The *tie strength*-based similarity is calculated based on the relationships between users [26]. It first extracts different relationships among users, each of which is assigned four variables: time amount T, intimacy I, intensity E and reciprocity R. Then, the *tie strength* between two users is calculated by the linear combination of the variables over all their relationships. Based on this method, we extract from our dataset D_p five relationships: following, collecting, favouring, sharing, and co-acting. Suppose we have two users u_i and u_j. The following relationship exists between u_i and u_j if u_i follows u_j. Likewise, they have collecting relationship if u_i collects items of u_j, favouring relationship if u_i favours items of u_j, and sharing relationship if u_i shares items of u_j, co-acting relationship if u_i and u_j act on the same items. For a particular relationship of two users, its time amount T is the length of the time slot from the relationship happening until now. The value of T is normalized to a range $[0, 1]$. Its intimacy value is set to 1, which means this relationship exists between these two users. We assign different relationships different intensity values. Here, we assign the intensity of following 5, that of collecting 4, that of favouring 3, that of sharing 2, and that of co-acting 1. A bigger intensity value indicates a more strong relationship. The reciprocity R is the number of items co-acted by these two users. The final *tie strength* between these two users is calculated by:

$$Y = \beta_0 + \sum_{j=1}^{5} \beta_1^j T_j + \sum_{j=1}^{5} \beta_2^j I_i + \sum_{j=1}^{5} \beta_3^j E_j + \sum_{j=1}^{5} \beta_4^j R_j \qquad (1)$$

where Y is the *tie strength* value, β_i $(i = 0, ..., 4)$ are coefficients associated to the different variables. These coefficients are estimated from our ground truth by linear regression [26].

To evaluate the effectiveness of different measurement, we define the similarity variance $\hat{e}$, which is the difference between the similarity value obtained by a measurement and that from the ground truth.

$$\hat{e} = \frac{1}{n} \sum_{u_i, u_j} |sim(u_i, u_j) - gt(u_i, u_j)| \qquad (2)$$

where $sim()$ is the user similarity obtained by a measurement, $gt()$ that from the ground truth, and n the number of total user pairs. Table 2 reports the comparison results on the similarity variances generated by different user similarity measurements over our dataset.

TABLE 2: Comparison of user similarity measure

Measurement	*PathSim*	*path count*	*random walk*	*tie strength*
$\hat{e}$	0.218	0.539	0.431	0.232

Clearly, compared with other measures, *PathSim* is more accurate. This is because *PathSim* fully captures the semantics of entities, while the other two path-based methods tend to be biased to entities with large links. The *Tie strength* only considers the direct relationship between users, which ignores the effect of indirect social user relationships, leading to a bigger similarity variance. Thus, we exploit *PathSim* as user similarity measurement to find the similar user subgroups in the heterogeneous information network. The *PathSim* works as follows:

$$s(u_i, u_j) = \frac{2 \times |\{p_{u_i \rightsquigarrow u_j} : p_{u_i \rightsquigarrow u_j \in P}\}|}{|\{p_{u_i \rightsquigarrow u_i} : p_{u_i \rightsquigarrow u_i \in P}\}| + |\{p_{u_j \rightsquigarrow u_j} : p_{u_j \rightsquigarrow u_j \in P}|\}} \qquad (3)$$

where $u_i \rightsquigarrow u_j$ is a path instance between u_i and u_j, $u_i \rightsquigarrow u_i$ is a path instance between u_i and u_i, and $u_j \rightsquigarrow u_j$ is a path instance between u_j and u_j. In Eq. 3, the numerator is the number of path instances between two entities in path set P. The denominator is the number of path instances through a specific entity. The more paths between u_i and u_j, and the less u_i or u_j interacts with other items, the more similar u_i and u_j is. We illustrate it in Figure 2(a).

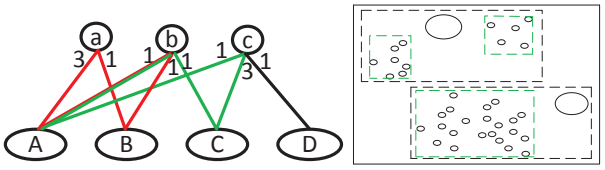


Fig. 2: (a) User similarity (b) Subgroup extraction

Given four users A, B, C, D and three items a, b, c, we only consider users are connected by one item. The connections by more users and items can be operated similarly. The red lines are all the path instances between A and B, and the green lines are all those between A and C. The numbers indicate the interactions between users and items. There are more paths between A and B than those between A and C. All B's interactions with items contained in the path between A and B, while one path of C is not in the set of paths between A and C. Applying Eq. 3, $sim(A, B) = \frac{2\times(3+1)}{(9+1+1)+(1+1)} = 0.62$, $sim(A, C) = \frac{2\times(1+3)}{(9+1+1)+(1+9)} = 0.38$, which is consistent with our analyses.

***k* subgroups extraction** Using *PathSim*, we further conduct the common-interest user subgroup extraction. The simple way of achieving this is to directly cluster all users into k sets in which users are close-connected. However, in a social group, the similarity of users is not the only decisive factor when extracting the subgroups. The users who have large influence attract more social interactions on a specific interest point. The influential users have two obvious characteristics: large followers and intensively interacted posts by the others. These distinct characteristics make them very far from less influential users by *PathSim* similarity. Thus traditional clustering treats them as noise. However, these users always work as the keys to connect the others when forming a common-interest user subgroup. For example, in Figure 2(b), the bigger circles represent the higher influential users. The closer distance between users indicates the higher similarity between them. The black dashed boxes mean the real subgroups. If we only consider the similarity, the green dashed boxes are the final subgroups. Obviously, these influential users are regarded as noise when we apply k-means because they are not close to any normal users. To find the influential users, we propose *influential factor* to measure the influence of users in a given group.

Definition 2. Given a user u in a group G, his influential factor $\mathcal{F}(u)$ is defined as the sum of his follower number and the average interaction frequency of its posts in a given group. The value of $\mathcal{F}(u)$ is normalised into the range of [0,1]. The top k influential users are the k-users having the biggest influential factors in a group.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2879658, IEEE Transactions on Knowledge and Data Engineering

Recall that similarity-based subgroup extraction can miss the influential users in the whole group. On the other hand, if we select the most k influential users as the centroids for clustering, we may generate poor quality clusters, because these influential users may be close to each other but put into different clusters as seeds. To overcome these problems, we propose to select the k clustering centroids by considering both the influential factors and the distance between users. Basically, we find k users from the social group G who are not only influential but also far from each other. The selected users should satisfy the following objective function 4, which can be solved by genetic algorithm (GA) [15].

$$max \sum_{u_i,u_j \in G} (1 - s(u_i,u_j) + \frac{\mathcal{F}(u_i) + \mathcal{F}(u_j)}{2}) \quad (4)$$

Once we obtain k centroids, we can utilise the revised k-means to cluster the remaining users based on these selected centroids to produce k subgroups. Note that there is no need to update the centroid point as traditional k-means does because less influential users have no dominance in subgroups generation.

Algorithm 1 shows the algorithm for extracting subgroups. Given a social group, our algorithm performs mainly as follows: (1) for each user u in G, calculate its influential factor $\mathcal{F}(u)$ (lines 3-6); select k users by GA as centroids (line 7); (2) put the k extracted users in k subgroups respectively as centroid users (lines 8-11); (3) construct the heterogeneous information network (line 12); (4) calculate the *PathSim* value of each user with all the centroid users, and add each use to its closest subgroup S_i based on its *PathSim* with the centroid users (lines 13-18). As different

Algorithm 1 Subgroup Extraction

```
1: Input: G - Social group
2: Output: C - k subgroups
3: /* influential factor calculation*/
4: for each users u in G do
5:     calculate influential factor F(u)
6: end for
7: select k users U_k from G via GA
8: /*k subgroup extraction*/
9: for for each user u_i in U_k do
10:    S_i ← u_i /* S_i ∈ C */
11: end for
12: heterogeneous information network construction
13: for for each user u_i in U_k do
14:    for each users u_j in G − U_k do
15:        u ← u_j such that PathSim(u_i, u_j)_min
16:        S_i = S_i ∪ u
17:    end for
18: end for
19: C = {S_1, ..., S_k}
20: return C
```

groups may have different sizes, it is unfair to set a fixed subgroup number k for groups. We define *subgroup density* ρ which is decided by the number of interest subgroups and the group size. Let n be the number of users in a group, k that of its subgroups. The *subgroup density* of this group is computed by: $\rho = k/n$. Given a fixed *subgroup density* for all groups, the number of subgroups for any group can be adaptively decided to achieve the high recommendation quality. Thus, we will evaluate the effect of ρ, instead of k, on the system effectiveness in Section 5.3.1.

4.2 Subgroup-based recommendation

Among the collaborative filtering recommendation approaches, Netflix Prize competition has demonstrated that the SVD-based techniques are superior to the classic nearest-neighbor-based ones. It represents both users and items as a vector of f factors inferred from the ratings patterns. The user-item similarities are modelled as the inner products of two vectors. However, the data sparsity in rating matrix severely degrades the SVD-based recommendation performance. If we exploit all the users and items in the whole sharing community, the rating matrix constructed would be extremely large and sparse due to the large number of users and small number of ratings. Thus, it is not suitable for utilizing SVD-based collaborative filtering directly. Fortunately, a *dense* sub-matrix of the original rating matrix captures the important latent relationship between users and items [43], [44]. Intuitively, similar users share the common interests and tend to view some common items. When too many unrelated users and items are involved in matrix construction, a large number of noises will be introduced to the latent factors during the SVD operation over this matrix. Removing the unrelated users and items before matrix construction will lead to a dense and small rating sub-matrix of the original matrix. Given a subgroup, we select the related users and items based on the similarity between the interaction of each user in database and that of this subgroup. Given a social user u in database and a subgroup S, the interaction of u with respect to S is represented as a set of social items $I_{uS} = \{m_1, ..., m_n\}$, where m_i is an item interacted with both u and any users in S. The interaction of subgroup S is represented as a set of two dimensional vectors $I_S = \{< m_1, nu_1 >, ..., < m_p, nu_p >\}$, where m_i is an item interacted with any user in S and nu_i the frequency of an item m_i being interacted. Note that p is bigger than n since I_S includes all m_i in I_{uS}. The similarity between I_{uS} and I_S can be calculated by:

$$Sim_I(I_{uS}, I_S) = n \cdot \sum_{i=1}^{n} nu_i \quad (5)$$

Eq. 5 considers two factors, the size of database user interaction set and the item interaction frequency nu_i of the subgroup. If the user interacted with too many items that did not appear in I_S, his interest only overlaps little with that of S. Thus, the similarity of I_{uS} and I_S is low. Higher nu_i indicates more frequent interactions and similar interests with subgroup S. The number of items in dataset is extremely large. Representing all of them and calculating the similarity between the interactions of each user and that of each subgroup incurs high time cost. To accelerate this process, we take advantage of weak connections between users to roughly collect the potential users before the Sim_I-based further data selection. The weak connection between users is defined as follows.

Definition 3. Given a common-interest user subgroup S, its interacted item set $r(S)$ and a user $u \notin S$, if u interact with any item in $r(S)$, u weak connects to S, denoted as $u \sim S$.

With the support of Sim_I, we propose a novel data selection approach called CBCS (connection-based candidates selection) that adaptively selects the user and item candidates used for rating matrix construction for a given user subgroup. Given a subgroup, CBCS performs in three steps. First, we exploit the weak connection between each user in database and this subgroup to filter out the users far from the subgroup at low time cost. Then, we identify the similar user candidates of each member in the subgroup from the remaining database users by calculating the Sim_I-based similarity between the interaction of each database

user and that of the subgroup. Finally, we identify the interacted items of the selected user candidates. As such, a compact set of user and item candidates is generated for the next step item recommendation for this subgroup.

Given a subgroup, we construct a rating matrix over its selected candidate set, on which the SVD-based CF is performed to generate a list of top relevant items. Specifically, each user or item is represented as a f-dimensional vector of the latent factors that are the projections of this user or item on the singular values generated from SVD. A naive similarity matching between an item and a user is to compute the inner product of their representations. However, this naive method suffers from low recommendation quality, because users would like to accept the recommendations from their trusted ones in practice. Likewise, if a subgroup trusts a user, the recommended items from this user should given a higher weight. We take advantage of the *following* relationship to capture the trust between a user and a subgroup as defined below.

Definition 4. In a *following* relationship described as a directed graph $\mathcal{G} < U, E >$, where U is a set of users and E is a set of relationships among users. Given $u_i \in U$ and $u_j \in U$, the directed edge $e_i \in E$ from u_i to u_j indicates that u_i follows u_j. If there exists a path from u_i to u_j, we say u_i trusts u_j. Let the path length from u_i to u_j be $|l(u_i, u_j)|$. The trust value τ is defined as $|l(u_i, u_j)|^{-1}$. Given a user u_i and a subgroup S, the trust between them is defined as the average trust value between u_i and each u_k in S.

The recommendation score between an item m and a subgroup $S = \{u_1, ..., u_n\}$ of n users is defined as the trust-based average inner product as follows.

$$SimT(m, S) = \sum_i^n \frac{\tau_i}{n} \cdot \sum_i u_i \cdot m \tag{6}$$

With the support of $SimT$ measure, we can easily identify the top-k relevant items of a given group.

4.3 Dynamic recommendation aggregation

After obtaining the recommendation lists for all common-interest user subgroups, we aggregate them to generate the final results for the whole group. The traditional aggregation approaches, including average satisfaction, minimum misery, and maximum satisfaction, assign the same weight to all individuals in the group, which does not reflect the real contributions of different group users in practice. In addition, these existing approaches treat a group as a static user set, i.e., the interests of the group are not changeable. Obviously, this assumption is not reasonable. The interests of a group change over time frequently. For example, fashion group is interested in some clothes in summer, while dislike them in winter. To overcome the limitations of existing techniques, we propose a dynamic group aggregation scheme (DGAS) that considers two factors of *subgroup contribution* and *interest activeness*. DGAS gives weights to the subgroups based on their *contributions* to the whole group. Then it detects the *interest activeness* of each subgroup by group discussions.

Given a common interest user subgroup S, each user $u \in S$ has different interactions with items, such as uploading, favouring, commenting, sharing, collecting, commenting, and rating etc. As a user may join more than one group, its interactions may involve the items that are not contained in the current group G. The interaction frequencies of a user with respect to different groups reveal this user's contributions to them. Likewise, the contribution of subgroup S to G is measured by the interaction frequency between each user in S and items in G. Let the interaction between a subgroup S and G to be $I_{SG} = \{< u_1, m_1, \omega_1 >, ..., < u_m, m_n, \omega_{nm} >\}$, where u_i is a user in S, m_i an item in G, ω_i the interaction frequency between u_i and m_i. Then the contribution of interest subgroup S to G is calculated as follows.

$$C(S, G) = \sum_i^{nm} \omega_i \tag{7}$$

Besides contribution, we measure the interest activeness of a subgroup to reflect the influence of its users. Intuitively, in social communities such as Flickr, users not only interact with social items but also initiate discussions on specific topics. An initiated discussion in a group during a time period attracts a specific set of users from more than one interest subgroup to exchange their ideas. It is natural that the current discussed topic is closely related to the current active group interests. For example, a group called *global politics* triggers different discussions in different time periods such as Park Geun-hye on Feb. 2017, and US Election on Nov. 2016. The users who discussed under the formal discussion are more likely interested in Korean politics, while the latter discussion also attracts the users being interested in American politics. Thus, we exploit the information on the group discussion topics to identify the activeness of this group. Let the latest group discussion be a vector set $d =< \{t_1, u_1\}, ..., \{t_n, u_n\} >$, where t_i is the time length from now to topic discussion time of u_i. Given an interest subgroup S, we take the number of active users and the discussed time of a discussion d to measure the interest activeness of S in G, which is computed as follows.

$$\alpha(S, G) = \sum_i \lambda^{t_i} \cdot \begin{cases} 1 & u_i \in S \\ 0 & u_i \notin S \end{cases} \tag{8}$$

where $\lambda < 1$ is to adjust time effect. The more users in a subgroup are involved in the discussion of the current topic, the more active this subgroup is. The interests of previously discussed would fade with time. Considering the contribution of subgroup and its interest activeness, the recommendation score g between an item m and a group $G = \{S_1, ..., S_n\}$ of n interest subgroups is defined as follows.

$$g(m, G) = \sum_i^n \frac{C_i(S_i, G) + \alpha_i(S_i, G)}{2} \cdot SimT(m, S_i) \tag{9}$$

where the interest subgroup contribution C_i and active parameter α_i are values among [0,1] after normalization.

4.4 Cost analysis of DCSGR

In this section, we analyse the time cost of our proposed group recommendation approach DCSGR from its three components respectively. Recall that a social group G of n users can be divided into k subgroups. Suppose we have a subgroup S of n_S users, and the whole dataset consisting of N users and M items. Let the time cost of computing the similarity between two users be t_1, and the time cost of extracting an influential user from the group be t_2. As shown in Algorithm 1, there is no need to update the centroid point as traditional k-means does since we treat the extracted influential users as the centroid point. Thus, the time cost of extracting k common-interest user subgroups is:

$$T_E = k * t_2 + (n_S - k) * k * t_1 \tag{10}$$

where k is a constant and n_S the user number of a subgroup. The time complexity of subgroup extraction is $O(n_S)$, which is linear to the number of users in a subgraph.

Next, we analyse the time cost of subgroup-based recommendation. For each extracted subgroup, our approach performs SVD-based collaborative filtering over it. Suppose a $r * c$ matrix is formed with respect to a subgraph, including its extracted candidate users, items from the whole dataset and their interactions. The time complexity of generating k subgroup recommendation lists is $O(k(r^2c + c^3))$ [1], which is exponential to the number of users and that of items.

Finally, DCSGR aggregates all the subgroup-based recommendation lists as the final one with the consideration of subgroup contribution and activeness. Suppose the time cost of comparing an element in a vector is t_3, the time cost of an interest subgroup contribution is $t_3 * n_S$ and activeness is $t_3 * n$. The time complexity of aggregation is $O(n)$, which is linear to the number of users in the whole group.

From the analyses above, the most costly part is the subgroup-based recommendation. The traditional approaches treat each single group member as recommending target and conduct SVD collaborative filtering over the whole dataset. The time complexity is as follows: $O(n(N^2M + N^3))$. Since DCSGR transforms the operations for individuals into similar user sets and conducts SVD on a subset of whole dataset, i.e., $k < n, r < N, c < M$, the time cost of DCSGR is much lower than that of the traditional single user based group recommendation.

4.5 Extending DCSGR for hybrid recommendation

Recall that our DCSGR can perform collaborative filtering-based recommendation in an effective and efficient way. In this section, we further discuss how to extend DCSGR approach for hybrid-based recommendation. We design an alternative approach based on DCSGR, called DCSGR-h, which embeds the content and demographics together with the collaborative filtering.

Borrowing the idea of content representation in [56], we exploit the TF/IDF model over the titles and descriptions of items, which describes the content of each item as a vector $v = \{t_1, t_2, ..., t_n\}$, where n is the number of terms in the whole corpus, and t_i a $tf - idf$ value of a term. Then the similarity between two items over content attribute is calculated by the cosine similarity of their content vectors. Given a target subgroup S and an item m for recommendation, let $r(S)$ be the items interacted by users in S, we define the cosine similarity-based relevance over content between S and m as follows:

$$SimC(m, S) = ave_{m' \in r(S)}\ cosine(m, m') \tag{11}$$

We exploit user multiple types of demographics in our DCSGR-h, including age, gender, occupation and location. Like in [50], their demographic similarity between two users u_1 and u_2 is calculated by $p(u_1, u_2) = \frac{1}{4}\sum_{i=1}^{4} d_i(u_1, u_2)$, where $d_i \in [0, 1]$ is the similarity over the ith attribute. The location similarity of two users is decided by their $L_2 - norm$ over normalized space. For other attributes including age, gender, occupation, if the values of two users over an attribute are equal, the similarity over it will be equal to 1, and to 0 otherwise. Given an item m, its demographic information is obtained from the set of users who viewed it, denoted as U_m. Given a subgroup S, we define the demographics similarity between m and S based on the similarity between U_m and S, which is as below:

$$SimD(m, S) = \max_{i=0}^{|U_m|} \max_{j=0}^{|S|} p(u_m^i, u^j) \tag{12}$$

where $u_m^i \in U_m$ and $u^j \in S$. The similarities between an item m and a subgroup S over collaborative filtering score, content and demographics are integrated to form the final relevance of them, which is as follows:

$$s(m, S) = \lambda_1 SimT(m, S) + \lambda_2 SimC(m, S) + \lambda_3 SimD(m, S) \tag{13}$$

where $\lambda_1 + \lambda_2 + \lambda_3 = 1$. All subgroup recommendation results are aggregated for the final group recommendation by DGAS scheme.

5 Experimental Evaluation

We report our experimental results to evaluate the effectiveness and efficiency of our proposed dynamic connection-based social group recommendation (DCSGR).

5.1 Experimental setup

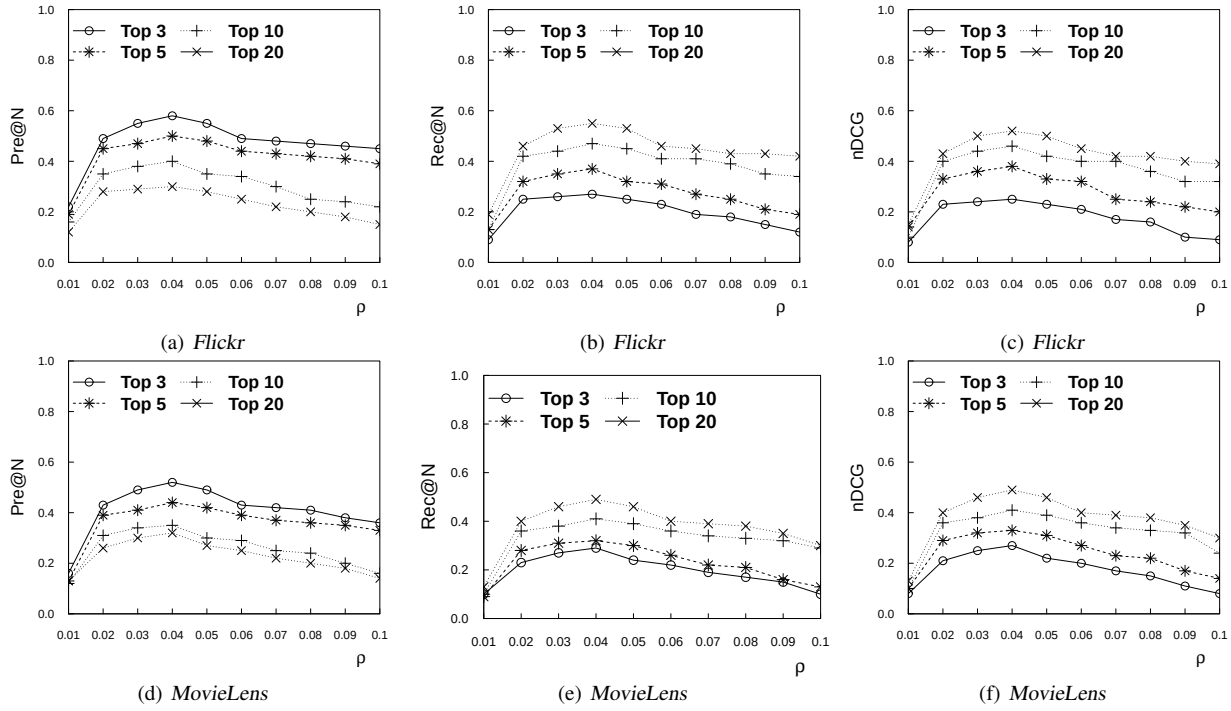
We conduct experiments on three real datasets, *Flickr* dataset, *MovieLens* and *MovieLens(1M)* datasets. The *Flickr* dataset was crawled on Flickr from Jan 2015 to Dec 2016, which includes 2M images and 500 popular groups. These groups are created by the group owner explicitly, and their information including group members and items can be collected by using the Flickr API. For each image, we collected all its interacted users. For each group, we collected all its members, images and discussions. The size of collected groups are ranged from tens to thousands. Since there is no explicit user ratings on Flickr, we treat the number of interactions of a user with an image as its rating. The *MovieLens* and *MovieLens(1M)*dataset was extracted from 20M MovieLens database [2]. Users rated movies they have watched from 1 to 5. *MovieLens* includes 20 million ratings of 27,278 movies by 138,493 users. All selected users had rated at least 20 movies. *MovieLens(1M)* contains 1M ratings of 3,900 movies by 6,040 users. Among our three datasets, only *MovieLens(1M)* provides users' demographic information.

Since *MovieLens* and *MovieLens(1M)* have no explicit groups, we make up groups following [55]. Specifically, each user is represented as a vector of item ratings. The user-to-user similarity is computed using Pearson correlation coefficient. Groups are defined as a set of users in which the average user-to-user similarity is higher than a predefined threshold. Here, the threshold is defined as the average user-to-user Pearson correlation coefficient of all user pairs in *MovieLens*. We first randomly select a set of users and keep them as a group if it satisfies our group definition. We finally obtained 500 groups from *MovieLens*. These 500 groups are divided them into two parts. The first part includes all 100 small groups, each of which contains no more than 100 users. The second part includes all 400 big groups, each of which contains more than 100 users. Similarly, 100 groups are obtained from *MovieLens(1M)* with 20 small groups and 80 big groups. We test the performance over both small and big group sets.

5.2 Evaluation methodology

We evaluate our proposed DCSGR in terms of effectiveness and efficiency. First, we evaluate the effect of the *subgroup density* ρ to obtain its optimal value. Then we evaluate the effects of CBCS

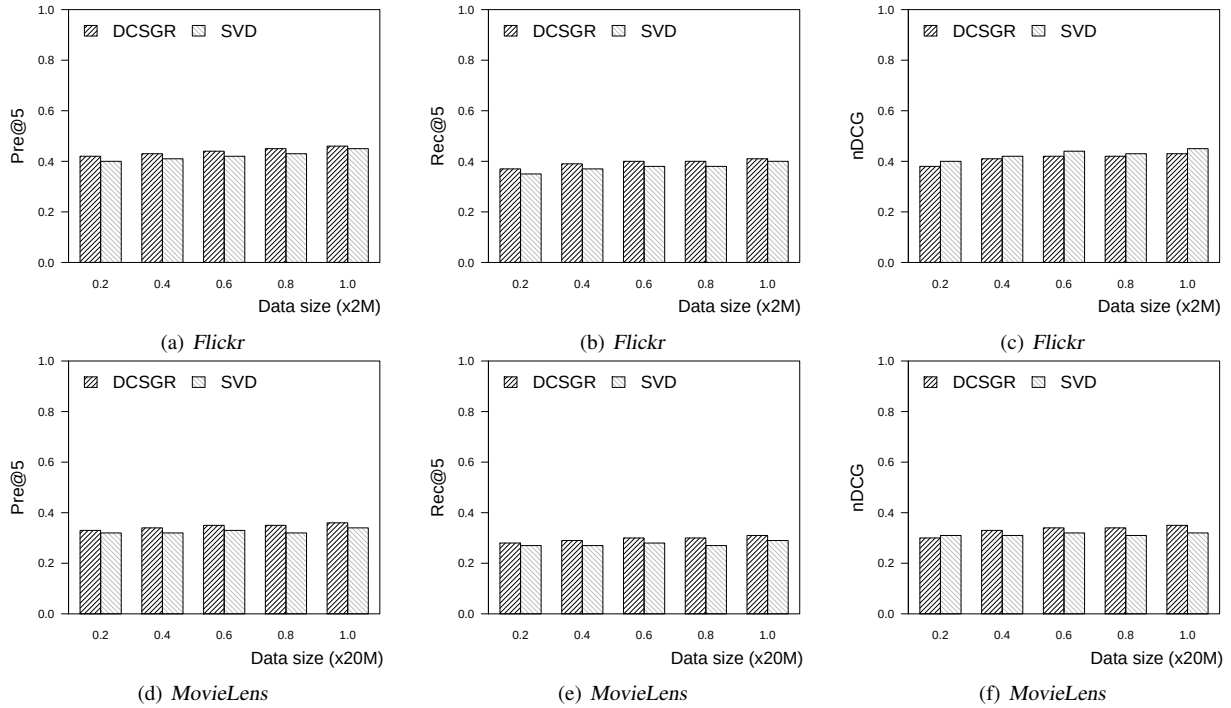
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2879658, IEEE Transactions on Knowledge and Data Engineering



(a) *Flickr* (b) *Flickr* (c) *Flickr*

(d) *MovieLens* (e) *MovieLens* (f) *MovieLens*

Fig. 3: Effect of subgroup density ρ



(a) *Flickr* (b) *Flickr* (c) *Flickr*

(d) *MovieLens* (e) *MovieLens* (f) *MovieLens*

Fig. 4: Effect of CBCS

strategy and group size on the recommendation quality. Finally, the effectiveness and efficiency of our DCSGR are evaluated.

For each group, we compare our proposed DCSGR and its three alternatives DCSGR-AVL, DCSGR-LM and DCSGR-h with four existing approaches, COM [55], CF-RD [4], MFCF [36] and TRIP [7]. COM is the state-of-art content-based recommendation approach and estimates the preference of a group to an item by aggregating the preferences of the group users with different weights. CF-RD is a typical collaborative filtering method and calculates the group recommendation based on the relevance and disagreement of the group. MFCF is a state-of-art collaborative filtering-based competitor, which builds a virtual user to present the group and generates recommendation by Matrix Factorization technique. TRIP is a state-of-art hybrid approach which combines demographic and rating information to generate recommendation. DCSGR is our proposed dynamic connection-based social group recommendation. DCSGR-AVG is our proposed alternative with averaging aggregation strategy, and DCSGR-LM is another al-

ternative with least-misery aggregation. DCSGR-h is the hybrid version of DCSGR.

To evaluate the effectiveness of our recommendation approach, we use three popular metrics in [55], average precision@N (*Pre@N*), average recall@N (*Rec@N*) and normalized discounted cumulative gain (*nDCG*), where N is the number of recommendations. *Pre@N* is the fraction of the top N recommendations that are adopted by a group, while *Rec@N* is the fraction of items accepted by a group that are contained in the top N recommendations. A recommendation is accepted by a group if this recommendation is interacted by any member of the group, such as viewed, liked or commented. Formally, given a group, the *Pre@N* and *Rec@N* are calculated as:

$$Pre@N = \frac{|top\ N\ recommendations| \cap |accepted\ items|}{|top\ N\ recommendations|} \quad (14)$$

$$Rec@N = \frac{|top\ N\ recommendations| \cap |accepted\ items|}{|accepted\ items|} \quad (15)$$

nDCG measures how well a method can rank the true item higher in the recommendation list. It is calculated as follows:

$$DCG = rel_1 + \sum_{i=2}^{N} \frac{rel_i}{log_2(i)} \qquad (a)\ nDCG = \frac{DCG}{IDCG} \qquad (b) \quad (16)$$

where $rel_i = 1$ if the ith item in the recommendation list is accepted by the group, and $rel_i = 0$ otherwise. *IDCG* is the maximum possible discounted cumulative gain (*DCG*) with the top N relevant items. We average the *nDCG* values of all groups as the final result.

On each dataset, we divide the whole data into two parts by time. We use the first 3/4 data for recommendation, and the rest for result verification. We conduct all sets of tests over *Flickr* and *MovieLens* datasets. In addition, to compare our approaches with demographic information-based approaches, we report the test results over *MovieLens(1M)* that contains demographic information. All experiments are conducted on a server using an Intel Xeon E5 CPU with 256 GB RAM running RHEL v6.3 Linux.

5.3 Effectiveness

We first evaluate the effects of subgroup density ρ, CBCS, and group size n. Then, we compare our proposed DCSGR with existing competitors to prove its high effectiveness performance.

5.3.1 Effect of subgroup density ρ

We evaluate the effect of subgroup density on the effectiveness of our DCSGR framework in terms of three metrics: *Pre@N*, *Rec@N* and *nDCG*, by varying the value of ρ from 0 to 1. Figures 3 (a)-(f) show the effectiveness changes of our system with respect to different ρ values using the *Flickr* and *MovieLens* datasets. As we can see, with the increasing of ρ, the results on three metrics show the same trend over two datasets. From 0.01 to 0.04, the effectiveness increases gradually, and reaches to its peak values. With the further increasing of ρ, the effectiveness of our DCSGR drops. This is caused by two reasons. On the one hand, when ρ is set to a value between 0.01 to 0.04, a bigger ρ ensures more distinct group interests can be captured, which enhances the effectiveness of our system accordingly. On the other hand, after the peak point 0.04, with the increase of ρ, there are too many interest subgroups. Thus similar users are categorized into different subgroups, failing to well extract different group interests. Thus we set ρ to 0.04 as its optimal value to obtain an optimal effectiveness of the system.

5.3.2 Effect of CBCS

We conduct experiments over two datasets to test the effect of our CBCS strategy on the recommendation quality of the system. For each interest subgroup, we randomly add 20%, 40% until 100% data from the whole dataset for the recommendation. On each step, we recommend items by DCSGR and the traditional SVD approach over the whole dataset separately. Figures 4 (a)-(f) show the results. Here, we only show the system effectiveness in terms of *Pre@5* , *Rec@5* and *nDCG* for space limitation following the setting in [55]. Form the Figures, we can see that the effectiveness of our DCSGR system with CBCS embedded is comparable with the traditional SVD-based system. This is because our CBCS filters out the unrelated users and items, thus the latent factors obtained by SVD over the selected candidate set precisely represent the item and user attributes. The unrelated items and users result in more redundant factors. Each redundant factor represents an attribute that does not belong to our subgroup users. Thus, CBCS strategy keeps the high effectiveness of the traditional SVD-based recommender systems.

5.3.3 Effect of group size n

Experiments are conducted over *Flickr*, *MovieLens* and *MovieLens(1M)* by varying the group size from 100 to 1000. As our *Flickr* and *MovieLens* datasets do not provide demographic information, we only report the results of DCSGR-h and TRIP over *MovieLens(1M)* dataset [2]. For DCSGR-h approach, we test the effectiveness of using different values of parameter combinations with a weight change step 0.05. The optimal values of three weight parameters in Eq. 13 for DCSGR-h are selected when the best performance is achieved. We finally report its effectiveness under the setting of optimal weights $\lambda_1 = 0.85$, $\lambda_2 = 0.10$, $\lambda_3 = 0.05$.

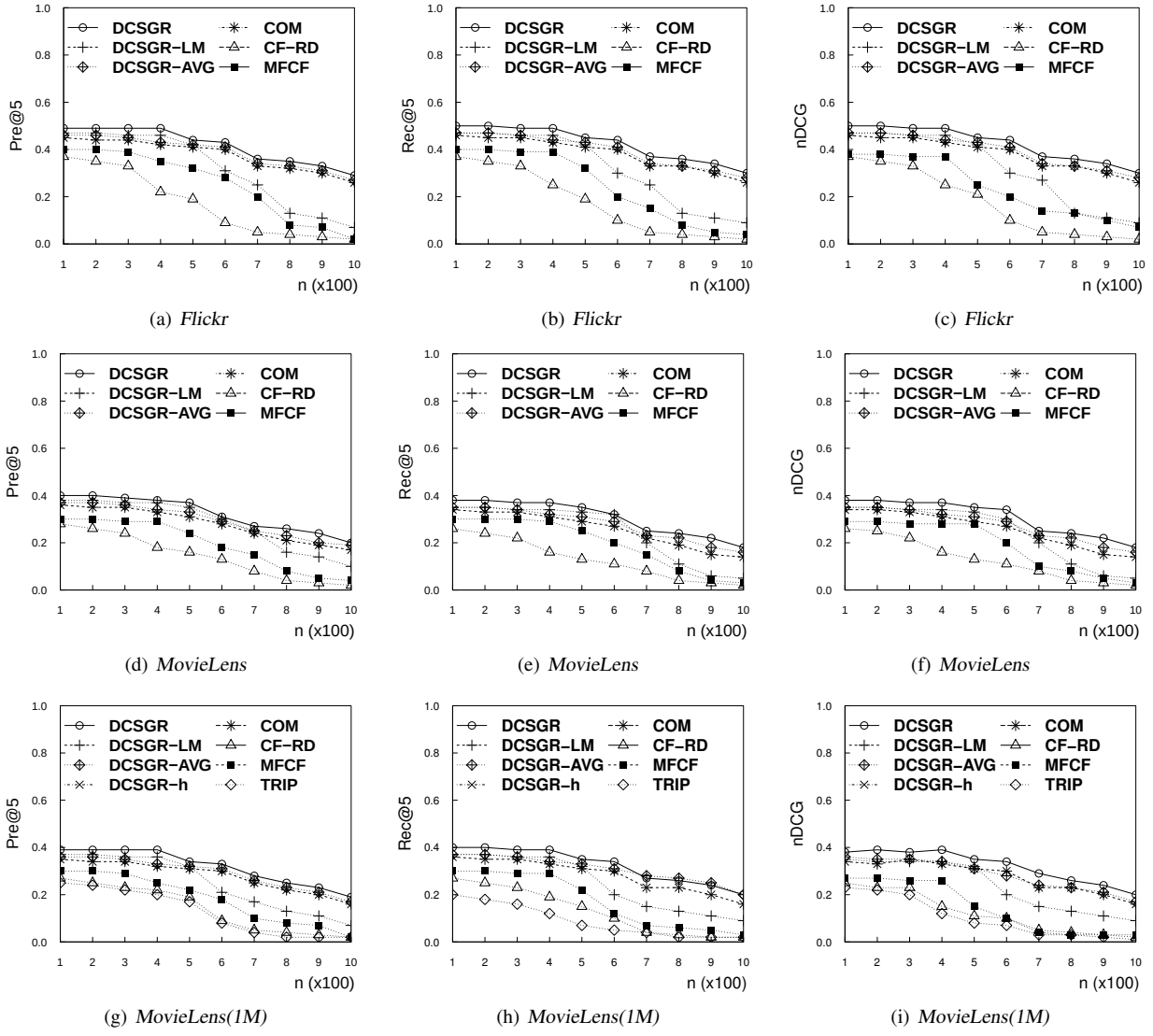
Figures 5 (a)-(i) show the average *Pre@N*, *Rec@N* and *nDCG* of our system at each n value. Here we still report the results of *Pre@5* , *Rec@5* and *nDCG* only following the setting in [55]. Clearly, when the group size is smaller than 600, all approaches keep high performance. With the further increase of group size after 600, our approaches perform better than COM, followed by MFCF, CF-RD and TRIP. This is caused by two reasons. For one thing, our approaches fully exploit the interests of a group. Thus the difference between users is taken into consideration when making recommendation. For another, COM represents the relatively steady group interests by some extracted changeable and vauge topics, which are not suitable for group interest representation. The effectiveness of CF-RD, MFCF and TRIP drop quickly with the increase of group size, because they treat the group as a whole in recommendation, which is not discriminative for big groups. Thus, we can draw the conclusion that the existing approaches work well for small group but cannot survive when the group size increases greatly. Our proposed DCSGR and its two alternatives DCSGR-LM, DCSGR-AVG are suitable for big group recommendation as they perform more stably with the increase of group size.

5.3.4 Recommendation comparison

In this section, we compare the effectiveness of our proposed methods DCSGR, its three variants DCSGR-h, DCSGR-LM, DCSGR-AVG and four existing approaches, CF-RD, COM, MFCF and TRIP. Among them, DCSGR-h and TRIP are hybrid approaches which need demographic information to generate recommendation.

First, we evaluate the effectiveness of DCSGR, DCSGR-LM, DCSGR-AVG, CF-RD, COM and MFCF on *Flickr* and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2879658, IEEE Transactions on Knowledge and Data Engineering



(a) *Flickr* (b) *Flickr* (c) *Flickr*

(d) *MovieLens* (e) *MovieLens* (f) *MovieLens*

(g) *MovieLens(1M)* (h) *MovieLens(1M)* (i) *MovieLens(1M)*

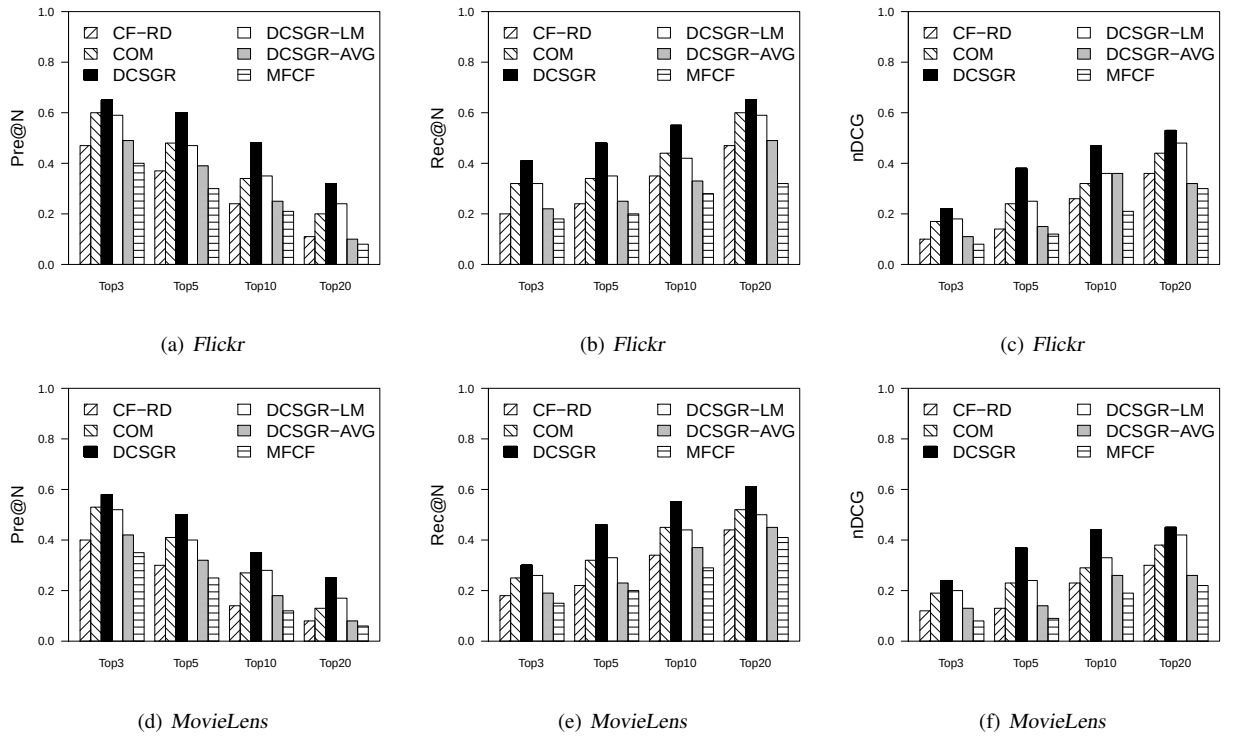
Fig. 5: Effect of group size

MovieLens. The subgroup density ρ is set to the optimal value for each approach. Figures 6 (a)-(f) show the comparison of different approaches in terms of the average *Pre@N*, *Rec@N* and *nDCG* of each group. As we can see, DCSGR performs best, followed by COM. The reason lies in the advantages of our group interest extraction scheme that well processes big group recommendation as analysed in Section 5.3.3. Moreover, DCSGR performs better than DCSGR-AVG and DCSGR-LM, because DCSGR considers the group interest dynamics. DCSGR-LM satisfies the least satisfied users, while DCSGR-AVG satisfies more users no matter how much they like when integrating the recommendation lists with respect to different interest subgroups. Accordingly, the group interests can not be well captured by them. CF-RD does not perform well since it treats the interests of a group as static and fails to exploit the dynamics characteristics of different group interests. MFCF performs the worst because it builds a virtual profile for the whole group, which is hard to represent the diverse interests of big groups. Thus, our proposed DCSGR is superior to the other approaches in terms of recommendation quality.

Then, we compare all the approaches over *MovieLens(1M)* dataset. From Figure 6 (g)-(h), we can see that DCSGR performs better than CF-RD, COM, MFCF, DCSGR-LM and DCSGR-AVG over *MovieLens(1M)* as for *Flickr* and *MovieLens* datasets. Meanwhile, DCSGR-h performs best, while TRIP and DCSGR achieve comparable performance in terms of effectiveness. This is caused by two reasons. For one thing, DCSGR-h and TRIP exploit more information in recommendation including rating, content and demographics, while DCSGR only uses rating information. For another, DCSGR-h and DCSGR construct group user profile over subgroups, which captures more discriminative information on user preference comparing with the whole group based user profile in TRIP. Thus, DCSGR takes advantages of finer information representation, which compensates the lacking of features in user profile construction, leading to similar effectiveness with TRIP. DCSGR-h benefits from enough features and finer representations of group profiles, leading to the best effectiveness.

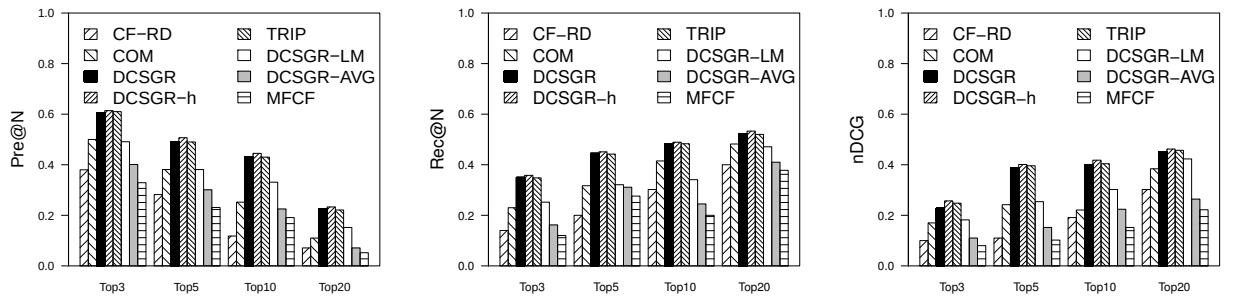
5.4 Efficiency

We evaluate the efficiency of our approaches over our three datasets. Like effectiveness evaluation, we first compare six approaches, including DCSGR, DCSGR-LM, DCSGR-AVG, CF-RD, COM and MFCF, over *Flickr* and *MovieLens*. Then we



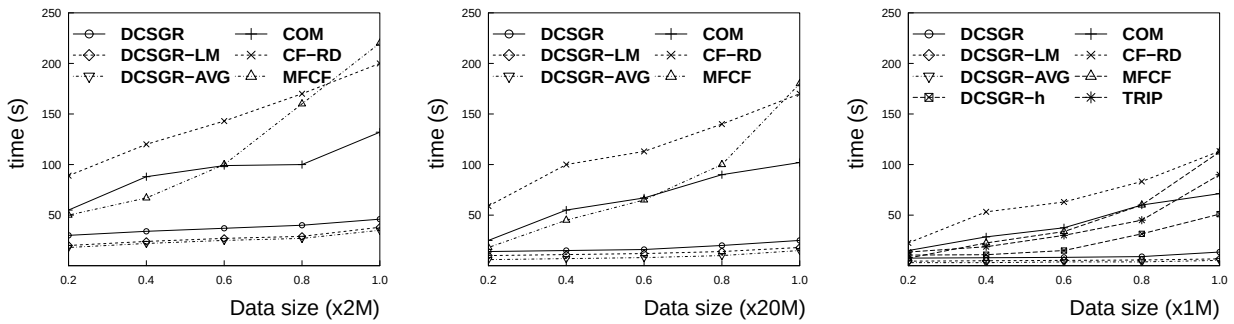
(a) *Flickr* (b) *Flickr* (c) *Flickr*

(d) *MovieLens* (e) *MovieLens* (f) *MovieLens*



(g) *MovieLens(1M)* (h) *MovieLens(1M)* (i) *MovieLens(1M)*

Fig. 6: Effectiveness comparison



(a) *Flickr* (b) *MovieLens* (c) *MovieLens(1M)*

Fig. 7: Efficiency comparison

test all the approaches over *MovieLens(1M)*. For each interest subgroup, we randomly add 20%, 40% until 100% data from the whole dataset for the recommendation. Note that we have incorporated the candidate selection time into our DCSGR system when we record the time cost of our approach.

From Figure 7 (a)-(b), we can see that our DCSGR performs much better than other competitors. Clearly, the cost of our DCSGR increases slowly. For one thing, our approach selects the similar users and items as the candidate set, which makes the rating matrix highly compact, and the number of latent factors is much less than that generated by the original SVD. For another, the number of similar users in a subgroup and that of items in the

whole dataset increase slowly. Comparing with COM, our DCSGR improves the efficiency 2-4 times. The higher time cost of COM is mainly caused by its individual-based group user processing scheme, which generates a recommendation list for each single user in a group and integrates all the lists for different users. Thus, more operations are required for a specified group recommendation. Comparing with CF-RD, our DCSGR improves the recommendation efficiency 3-7 times. As CF-RD adopts individual user-based group recommendation, identify the relationships of users in a group and those between items in browsing history, even more operations are required in recommendation using this approach. Comparing with MFCF, our DCSGR improve the efficiency 2-5 times. This is because MFCF conducts collaborative filtering over the whole dataset and exploits more complex features in group recommendation. Thus our DCSGR has great superiority over existing systems in terms of efficiency.

We evaluate the efficiency of our approach by including the comparison with demographic information-based approach, TRIP and DCSGR-h, over *MovieLens(1M)* dataset. The experimental results are reported in figure 7 (c). Clearly, the efficiency of DCSGR is better than CF-RD, COM, MFCF, DCSGR-LM and DCSGR-AVG as performed over *Flickr* and *MovieLens*. The time costs of DCSGR-h and TRIP is much higher than that of DCSGR, because the hybrid approaches exploit more information when generating recommendation. Considering the effectiveness and efficiency, DCSGR achieves comparable effectiveness with TRIPS and DCSGR-h, while achieves high efficiency improvement. Thus, our proposed DCSGR-based collaborative filtering is more suitable to big group recommendation.

6 Conclusion

In this paper, we study the problem of social group recommendation in online services. We first propose a novel interest subgroup extraction approach to represent the multi-interests of a group. Then we adaptively select the similar users and items based on each subgroup to generate a compact and small rating matrix for efficient collaborative filtering. Finally, we propose a dynamic aggregation function to combine all recommendation lists as the final results delivered to the group. We have conducted extensive experiments to evaluate our proposed recommendation approach. The experimental results have proved the high effectiveness and efficiency of our system compared with existing competitors.

References

[1] http://rakaposhi.eas.asu.edu/s01-cse494-mailarchive/msg00028.html.

[2] https://grouplens.org/datasets/movielens/.

[3] http://www.marketingcharts.com/industries/media-and-entertainment-21011.

[4] S. Amer-Yahia, S. B. Roy, A. Chawlat, G. Das, and C. Yu. Group recommendation: Semantics and efficiency. *The VLDB Journal*, 2(1):754–765, 2009.

[5] A. Anagnostopoulos, L. Becchetti, C. Castillo, A. Gionis, and S. Leonardi. Online team formation in social networks. In *WWW*, pages 839–848. ACM, 2012.

[6] Q. Ba, X. Li, and Z. Bai. Clustering collaborative filtering recommendation system based on svd algorithm. In *ICSESS*, pages 963–967, 2013.

[7] I. Christensen, S. Schiaffino, and M. Armentano. Social group recommendation in the tourism domain. *Journal of Intelligent Information Systems*, 47(2):209–231, 2016.

[8] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Trans. Information Systems*, 22(1):143–177, 2004.

[9] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE TKDE*, 19(3), 2007.

[10] M. Gartrell, X. Xing, Q. Lv, A. Beach, R. Han, S. Mishra, and K. Seada. Enhancing group recommendation by incorporating social relationship interactions. In *Proceedings of the 16th ACM international conference on Supporting group work*, pages 97–106. ACM, 2010.

[11] M. A. Ghazanfar and A. Prügel-Bennett. The advantage of careful imputation sources in sparse data-environment of recommender systems: Generating improved svd-based recommendations. *Informatica*, 37(1):61–92, 2013.

[12] T. Gross, C. Beckmann, and M. Schirmer. Grouprecopf: Innovative group recommendations in a distributed platform. In *PDP*, pages 293–300. IEEE, 2011.

[13] G. Guo, J. Zhang, and D. Thalmann. Merging trust in collaborative filtering to alleviate data sparsity and cold start. *Knowledge-Based Systems*, 57:57–68, 2014.

[14] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Information Systems*, 22(1):89–115, 2004.

[15] C. R. Houck, J. Joines, and M. G. Kay. A genetic algorithm for function optimization: a matlab implementation. *Ncsu-ie tr*, 95(09):1–10, 1995.

[16] H.-C. Hsiao, Y.-H. Lin, A. Studer, C. Studer, K.-H. Wang, H. Kikuchi, A. Perrig, H.-M. Sun, and B.-Y. Yang. A study of user-friendly hash comparison schemes. In *Computer Security Applications Conference*, pages 105–114. IEEE, 2009.

[17] http://rejoiner.com/resources/amazon-recommendations-secret-selling online/.

[18] X. Hu, C. Chen, X. Chen, and Z.-K. Zhang. Social recommender systems based on coupling network structure analysis. *CoRR:1204.1949*, 2012.

[19] X. Hu, X. Meng, and L. Wang. Svd-based group recommendation approaches: an experimental study of moviepilot. In *Proceedings of challenge on context-aware movie recommendation*, pages 23–28, 2011.

[20] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *RecSys*, pages 79–86. ACM, 2010.

[21] G. Karypis. Evaluation of item-based top-n recommendation algorithms. In *CIKM*, pages 247–254. ACM, 2001.

[22] Y. Koren. The bellkor solution to the netflix grand prize. *Netflix prize documentation*, 81:1–10, 2009.

[23] T. Kurashima, T. Iwata, G. Irie, and K. Fujimura. Travel route recommendation using geotags in photo sharing sites. In *CIKM*, pages 579–588. ACM, 2010.

[24] T. Lappas, K. Liu, and E. Terzi. Finding a team of experts in social networks. In *SIGKDD*, pages 467–476. ACM, 2009.

[25] L. Li, H. Tong, N. Cao, K. Ehrlich, Y.-R. Lin, and N. Buchler. Replacing the irreplaceable: Fast algorithms for team member recommendation. In *WWW*, pages 636–646, 2015.

[26] F. Liberatore and L. Quijano-Sanchez. What do we really need to compute the tie strength? an empirical study applied to social networks. *Computer Communications*, 110:59–74, 2017.

[27] H. Lieberman, N. Van Dyke, and A. Vivacqua. Let's browse: a collaborative browsing agent. *Knowledge-Based Systems*, 12(8):427–431, 1999.

[28] G. Linden, B. Smith, and J. York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet computing*, 7(1):76–80, 2003.

[29] H. Liu, Z. Hu, A. Mian, H. Tian, and X. Zhu. A new user similarity model to improve the accuracy of collaborative filtering. *Knowledge-Based Systems*, 56:156–166, 2014.

[30] A. Majid, L. Chen, G. Chen, H. T. Mirza, I. Hussain, and J. Woodward. A context-aware personalized travel recommendation system based on geotagged social media data mining. *Geographical Information Science*, 27(4):662–684, 2013.

[31] M. R. McLaughlin and J. L. Herlocker. A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In *SIGIR*, pages 329–336. ACM, 2004.

[32] K. Miyahara and M. Pazzani. Collaborative filtering with the simple bayesian classifier. *PRICAI 2000 Topics in Artificial Intelligence*, pages 679–689, 2000.

[33] E. Ntoutsi, K. Stefanidis, K. Nørvåg, and H.-P. Kriegel. Fast group recommendations by applying user clustering. In *International Conference on Conceptual Modeling*, pages 126–140. Springer, 2012.

[34] E. Ntoutsi, K. Stefanidis, K. Rausch, and H.-P. Kriegel. Strength lies in differences: Diversifying friends for recommendations through subspace clustering. In *CIKM*, pages 729–738. ACM, 2014.

[35] M. Oconnor, D. Cosley, J. A. Konstan, and J. Riedl. Polylens: a recommender system for groups of users. In *ECSCW*, pages 199–218. Springer, 2001.

[36] F. Ortega, A. Hernando, J. Bobadilla, and J. H. Kang. Recommending items to group of users using matrix factorization based collaborative filtering. *Information Sciences*, 345:313–324, 2016.

[37] O. N. Osmanli and İ. H. Toroslu. Using tag similarity in svd-based recommendation systems. In *AICT*, pages 1–4. IEEE, 2011.

[38] L. Quijano-Sánchez, J. A. Recio-García, and B. Díaz-Agudo. Modelling hierarchical relationships in group recommender systems. In *International Conference on Case-Based Reasoning*, pages 320–335, 2015.

[39] L. Quijano-Sanchez, C. Sauer, J. A. Recio-Garcia, and B. Diaz-Agudo. Make it personal: a social explanation system applied to group recommendations. *Expert Systems with Applications*, 76:36–48, 2017.

[40] P. RESNIK. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI*, pages 448–453, 1995.

[41] L. Q. Sánchez, B. D. Agudo, and J. A. R. García. *Impact of social factors and organizations in group recommendation processes*. PhD thesis, Universidad Complutense de Madrid, 2015.

[42] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *Conference on Electronic Commerce*, pages 158–167. ACM, 2000.

[43] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Application of dimensionality reduction in recommender system-a case study. Technical report, DTIC Document, 2000.

[44] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Incremental singular value decomposition algorithms for highly scalable recommender systems. In *CCIS*, pages 27–28. Citeseer, 2002.

[45] S. Seko, T. Yagi, M. Motegi, and S. Muto. Group recommendation using feature space representing behavioral tendency and power balance among members. In *RecSys*, pages 101–108. ACM, 2011.

[46] H. Shan and A. Banerjee. Generalized probabilistic matrix factorizations for collaborative filtering. In *ICDE*, pages 1025–1030. IEEE, 2010.

[47] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *The VLDB Journal*, 4(11):992–1003, 2011.

[48] J. Tang, X. Hu, and H. Liu. Social recommendation: a review. *Social Network Analysis and Mining*, 3(4):1113–1133, 2013.

[49] M. G. Vozalis and K. G. Margaritis. Applying svd on generalized item-based filtering. *IJCSA*, 3(3):27–51, 2006.

[50] M. G. Vozalis and K. G. Margaritis. Using svd and demographic data for the enhancement of generalized collaborative filtering. *Information Sciences*, 177(15):3017–3037, 2007.

[51] J. Wang, Z. Liu, and H. Zhao. Group recommendation based on the pagerank. *Journal of Networks*, 7(12):2019–2024, 2012.

[52] X. Xie and B. Wang. Web page recommendation via twofold clustering: considering user behavior and topic relation. *Neural Computing and Applications*, pages 1–9, 2016.

[53] X. Yang, H. Steck, Y. Guo, and Y. Liu. On top-k recommendation using social networks. In *Proceedings of the sixth ACM conference on Recommender systems*, pages 67–74. ACM, 2012.

[54] Z. Yu, X. Zhou, Y. Hao, and J. Gu. Tv program recommendation for multiple viewers based on user profile merging. *User modeling and user-adapted interaction*, 16(1):63–82, 2006.

[55] Q. Yuan, G. Cong, and C.-Y. Lin. Com: a generative model for group recommendation. In *SIGKDD*, pages 163–172. ACM, 2014.

[56] E. Zhong, N. Liu, Y. Shi, and S. Rajan. Building discriminative user profiles for large-scale content recommendation. In *SIGKDD*, pages 2277–2286. ACM, 2015.

[57] X. Zhou, L. Chen, Y. Zhang, L. Cao, G. Huang, and C. Wang. Online video recommendation in sharing community. In *SIGMOD*, pages 1645–1656, 2015.

[58] X. Zhou, J. He, G. Huang, and Y. Zhang. A personalized recommendation algorithm based on approximating the singular value decomposition (approsvd). In *Joint Conferences on Web Intelligence and Intelligent Agent Technology*, pages 458–464. IEEE Computer Society, 2012.

[59] X. Zhou, J. He, G. Huang, and Y. Zhang. Svd-based incremental approaches for recommender systems. *Computer and System Sciences*, 81(4):717–733, 2015.

Dong Qin received the Bachelor degree and Master degree in School of Computer Science and Technology from Xidian University, China, in 2011 and 2014, respectively. He is currently working toward the Ph.D. degree at RMIT University, Australia. His research interests include context-aware recommendation, social network analysis and big data processing.

Xiangmin Zhou received the PhD degree in computer science from The University of Queensland, Australia, in 2008. Currently, she is a is a Lecturer in Computer Science and Information Technology at RMIT University, Australia. Her research interests include social network analysis and mining, recommender systems and big data processing, multimedia databases and streams, query processing and query optimization.

Lei Chen received the BS degree in computer science and engineering from Tianjin University, China, in 1994, the MA degree from the Asian Institute of Technology, Thailand, in 1997, and the PhD degree in computer science from the University of Waterloo, Canada, in 2005. He is now an assistant professor in the Department of Computer Science and Engineering at Hong Kong University of Science and Technology. His research interests include multimedia and time series databases, sensor and peer-to-peer databases, and stream and probabilistic databases. He is a member of the IEEE.

Guangyan Huang Dr. Guangyan Huang is a senior lecturer in the School of Information Technology, Deakin University since 2014. She is also an Australia Research Council (ARC) Discovery Early Career Researcher Award (DECRA) fellow. She was awarded a PhD degree in Computer Science from Victoria University (VU) in 2012. She was a senior lecturer (2014) and postdoctoral research fellow (2012-2013) in VU. She was an assistant professor in the Institute of Software, Chinese Academy of Sciences from 2007 to 2009. She was a research assistant from 2003 to 2007 in the Institute of Computing Technology, Chinese Academy of Sciences. Between June 2006 and Dec. 2006, she visited Platforms and Devices Centre in Microsoft Research Asia. She has 70 publications mainly in data mining, sensor networks, image/video analysis and parallel algorithms for big data.

Yanchun Zhang Yanchun Zhang is a Professor and the Director of Centre for Applied Informatics at Victoria University. He received the National thousand Talent Program Award from China in 2010, and is currently a director on the Australia-China Joint Lab on Social Computing and E-Health, a joint initiative from Graduate University of Chinese Academy of Science and Victoria University. He obtained a PhD degree in Computer Science from The University of Queensland in 1991.