GERF: A Group Event Recommendation Framework Based on Learning-to-Rank

Yulu Du, Xiangwu Meng, Yujie Zhang, and Pengtao Lv

Abstract—Event recommendation is an essential means to enable people to find attractive upcoming social events, such as party, exhibition, and concert. While growing line of research has focused on suggesting events to individuals, making event recommendation for a group of users has not been well studied. In this paper, we aim to recommend upcoming events for a group of users. We formalize group recommendation as a ranking problem and propose a group event recommendation framework GERF based on learning-to-rank technique. Specifically, we first analyze different contextual influences on user's event attendance, and extract preference of user to event considering each contextual influence. Then, the preference scores of the users in a group are taken as the features for learning-to-rank to model the preference of the group. Moreover, a fast pairwise learning-to-rank algorithm, Bayesian group ranking, is proposed to learn ranking model for each group. Our framework is easily to incorporate additional contextual influences, and can be applied to other group recommendation scenarios. Extensive experiments have been conducted to evaluate the performance of GERF on two real-world datasets and demonstrate the appealing performance of our method on both accuracy and time efficiency.

Index Terms—Group recommendation, event-based social networks, cold-start recommendation, learning-to-rank

1 Introduction

Event-based social networks (EBSNs), such as Meetup[1] and Douban Event,[2] linking online and offline social interactions among users have become increasing popular in recent years. EBSNs provide convenient online platforms for users to create, distribute and organize offline social events, which promote face-to-face social interactions among users. Due to a large volume of events distributed on EBSNs every day, it is crucial to utilize information about events and users to make personalized event recommendation, which helps users find upcoming events that attract them to attend, facilitate organizers to advertise their events and find more potential participants.

Different from other items (e.g., movies and music), event possesses very short life cycle. For example, a concert may only continues two hours. The outdated events should not be recommended to users and the recommendation candidate list only consists of new events. Besides, many new events which are published in a short time receive few responses or not even one. Therefore, event recommendation inherently faces a serious cold-start problem [1], compared to other domain recommendations. Some recent works [2], [3] utilize different contextual information (e.g., content introduction, organizer, and venue of event) to alleviate the cold-start problem for event recommendation. However, these studies focus mainly on suggesting events for individuals and ignoring making recommendations for groups of users. People are gregarious due to common nature of human beings and usually prefer to attend events in the form of groups. For instance, users are likely to attend event with their friends or family. Because the recommendation approaches designed for individuals can not make suggestions for group of users directly, group recommender systems [4], [5] have been proposed to meet the demands of a group of users by identifying the items that appeal to the group as a whole. In this paper, we aim to offer accurate event recommendations for groups by considering different contextual influences on individuals' preferences.

The key problem of group recommendation is group preference modeling, which is a challenge task due to the diversity of group members' preferences. For example, friends may not share same tastes for all movies and the preferences of family members to music may be very different. One of the most successful methods to model group preference is aggregation-based method [5], [6], which aggregates individual preferences into group preference using an aggregation strategy. The aggregation-based methods can be classified into two strategies: *preference aggregation* [7], [8] first aggregates the profiles of group members into a new profile for the group, and then employs recommendation techniques designed for individuals to make recommendations for groups; *score aggregation* [4], [9], [10] first predict relevant scores for individuals using individual recommendation algorithm, and then employs an aggregation

1. https://www.meetup.com/
2. https://beijing.douban.com/

- *The authors are with the Beijing Key Lab of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, Beijing 100876, China, and the School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 1000876, China. E-mail: {yuludu, mengxw, zhangyj}@bupt.edu.cn, pengtaolv@163.com.*

Manuscript received 19 Dec. 2017; revised 14 Nov. 2018; accepted 8 Jan. 2019. Date of publication 16 Jan. 2019; date of current version 5 Mar. 2020. (Corresponding author: Xiangwu Meng.) Recommended for acceptance by D. Cai. Digital Object Identifier no. 10.1109/TKDE.2019.2893361

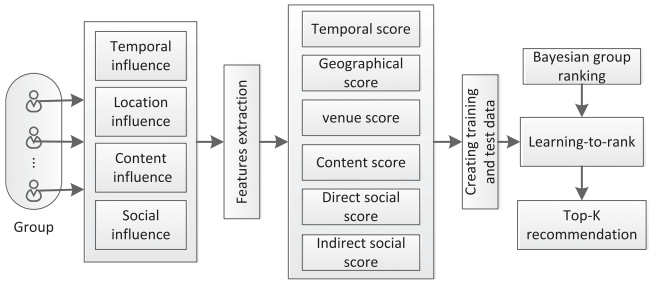


Fig. 1. Overview of group event recommendation framework.

strategy to combine individuals' scores into group score, which is further used to generate group recommendation. Masthoff et al. [11] introduce 10 heuristic aggregation strategies, most of which are based on social choice theory. However, the influence of aggregation strategy on the accuracy of the group recommendations is largely dependent on the individual recommendation algorithm and aggregation strategy [12]. Moreover, some experimental results show that the performances of aggregation strategies change a lot on different domains [13]. To the best of our knowledge, the previous studies [14] combine individuals' ratings to form group profiles and have not consider group recommendation as a learning-to-rank task, which is helpful for building more accuracy aggregation model than heuristic methods.

In addition, with increasingly contextual factors incorporated into group recommendation, the aggregation is not only required for modeling group score, but also needed for combining multiple contextual factors and the traditional heuristic methods may not achieve good performance. Although one type of content information is incorporated through prior distribution in the model proposed by [5], the cold-start problem has still not been alleviated due to the limitation of the model. Lu et al. [15] only consider geographical and social influences in their group recommendation model. Incorporating more context (e.g., content and organizer of event) into their model is difficult, since the model is too complex. As a consequence, it is crucial to design an extensible framework to model group preference with fusion of different contextual influences to alleviate the cold-start problem and achieve better performance for group event recommendation task.

In this paper, we propose a group event recommendation framework as shown in Fig. 1, called GERF, to model group preference by combining different contextual influences on individual's preference using learning-to-rank technique. We first formalize group event recommendation task as a learning-to-rank problem. Analogies to learning-to-rank problem in information retrieval, where a set of documents need to be ranked for a given query, a group is regarded as a query, an event is regarded as a document, and a training sample used to learning is created for each group-event pair. Each training sample contains a feature vector and a judgment, which represents the relevance degree of an event to the group. To adopt learning-to-rank technique to generate event recommendation for groups, GERF is split into two components: feature extraction and learning-to-rank for group event recommendation. In feature extraction part, we analyze six contextual factors including temporal, geographical, venue, content, direct and indirect social influences, based on which, we calculate each contextual score mirroring the relevance degree between users and the corresponding context. We consider above six contextual influences for group event recommendation in this paper, but our proposed GERF is quite extensible, and can incorporate more contextual influences by feature extraction.

In learning-to-rank for group event recommendation part, we take the different contextual scores obtained in feature extraction part as the features, and propose a metric to measure relevance degree of an event to a group to assign the judgment for each training sample input to learning-to-rank. To improve the efficient of learning-to-rank, we propose a temporal threshold-based method to reduce the number of training samples without loss of effectiveness. We adopt learning-to-rank algorithm to learn a ranking model for each group. The learned ranking model is used to predict ranking score of a new event to the group. The ranked recommendation list of events for a given group is generated by sorting all candidate events according to their predicted ranking scores to the group.

Although many existing learning-to-rank algorithms (e.g., RankSVM [16]) can be adopted in GERF, most of those methods are designed for information retrieval, where a single ranking model is learned and used to sort documents for any query. We believe that each group has unique preference, because of diversity of group composition, i.e., different groups have different members. It is more reasonable to learn a group-dependent ranking model for each group. In light of this, we propose a novel learning-to-rank algorithm, called Bayesian Group Ranking (BGR), for group event recommendation task. Specifically, we define the ranking model of each group is a linear function of the feature vector of each group-event pair. We estimate the parameters of ranking model based on the optimization criterion of Bayesian personalized ranking.

The main contributions of our work are summarized as:

- We formalize group recommendation problem as a learning-to-rank task, and propose a group event recommendation framework. Thus many existing learning-to-rank algorithms can be used to generate recommendation for groups.
- We propose a fast pairwise learning-to-rank algorithm for group event recommendation task, called Bayesian Group Ranking, to learn the weights of aggregation model for each group.
- We conduct extensive experiments on two real-world datasets with cross-validation to evaluate the performance of our method. The experimental results show the advantages of our proposals in both recommendation effectiveness and efficiency.

The rest of this paper is organized as follows. Section 2 introduces related work on traditional recommendation algorithms, group recommender systems and learning-to-rank techniques. Section 3 defines key concepts and formalizes group event recommendation problem. We analyze different contextual influence on user's event attendance and extract features for learning-to-rank through exploring temporal, geographical, venue, content and social influences, respectively, in Section 4. The group event recommendation framework (GERF) and a novel learning-to-rank algorithm

(BGR) are described in Section 5. We report the experimental results in Section 6 and conclude the paper in Section 7.

2 Background and Related Work

In this section, we first brief introduce personalized recommendation algorithms designed for individuals in Section 2.1. Then we introduce the recent development of general group recommendation in Section 2.2. Finally, we review the learning-to-rank technique in Section 2.3.

2.1 Personalized Recommendation Algorithms

The existing personalized recommendation algorithms can be classified into three categories: collaborative filtering (CF) [17], content-based methods [18] and hybrid methods [19]. CF has been widely used in many recommender systems, because its good performance and it does not require additional information except for users' ratings. However, CF methods fail to handle with cold-start problem, i.e., CF methods do not perform well to recommend new items, which are not consumed or rated, to users [1]. Content-based methods utilize content information of items to bridge the gap between history items and new items. To overcome the shortcomings of the systems using only one method, some hybrid methods, which combine CF and content-based approaches, have been proposed [9]. Recent studies mostly focus on context-aware recommendation, which exploits more additional information to make more accuracy recommendations, such as time [20], location [21] and social information [3]. To alleviate data sparse and cold-start problems of event recommendation, many context-aware recommendation methods have been proposed [22], [23].

2.2 Group Recommender Systems

Group reommender systems (GRSs) aim to find the items that appeal to all members in a group. One of the most popular techniques to model group preference is based on aggregation, which aggregate group members' preference into group preference. The aggregation methods can be classified into two categories: 1) preference aggregation [7], [8], [24] aggregates group members' profiles (e.g., ratings) into a group profile using aggregation strategy, then the group recommendation is generated by traditional recommendation algorithms which take a group as an user. 2) Score aggregation [4], [5], [9], [10] first uses traditional algorithms to predict score between each candidate item and each group member. After that, group members' predicted scores are aggregated into group score using an aggregation strategy.

There are many aggregation strategies [11] used to aggregate users' preference into group preference. Three representative aggregation strategies include: Average (AVG), Least Misery (LM) and Most Pleasure (MP). Some experimental results [12] show that there are no best aggregation strategies. The performance of group recommendation is influenced by grouping strategy (i.e., preference aggregation or score aggregation) and algorithm used to generate suggestions for individuals. For example, if social relationship between group members is strength, the group decision tends to reflect maximum satisfaction of the group members [25]. Berkovsky et al. [26] assume that more activity member has larger weight and assign weights for group members according to their observed activity (e.g., the number of ratings), but this assumption may not be true. To overcome the drawback of exiting aggregation strategies, in this paper, we propose a group event recommendation framework using learning-to-rank technique to learn group members' weights in aggregation. Moreover, the weights of different contextual influences on group preference also can be trained simultaneously. The experimental results show that our method outperforms the traditional heuristic aggregation strategies.

2.3 Learning-to-Rank

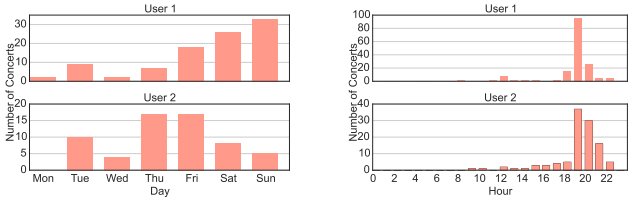
Learning-to-rank is a set of methods that use machine learning technologies to solve the problem of ranking. The goal is to automatically construct a ranking model from training data. The ranking model is used to produce a rank list of items for new data. One main difference between learning-to-rank and other machine learning problem is that the output of learning-to-rank is an ordered list of items. Learning-to-rank techniques have been applied in many areas, such as information retrieval [27], machine translation [28] and recommender systems [29], [30].

The learning-to-rank methods have been classified into three main categories [31]: pointwise, pairwise and listwise approaches. The pointwise approach treats the ranking task as a traditional regression or classification problem. The pairwise approach cares about the relative order between two items and transform the ranking problem to traditional binary classification problem. Each pair of objects (a, b) is classified to positive if a ranker higher than b, or negative otherwise. Once every object pairs have been ordered in this way, it is straightforward to construct the final ranked list. However, pairwise approach treats all rank in same. In fact, we more concern top k items in the total rank list. In the listwise approach, the ranking structure is explicitly considered by using ranking lists as instances in training and prediction. This approach measures similarity between two ranking lists when computing the loss function of ranking model. An advantage of this approach is that evaluation measures can be directly incorporated into the loss function, but listwise approaches are quite computationally expensive.

Many learning-to-rank methods have been proposed in the past decades, such as RankSVM [16], RankNet [32], RankBoost [33], Coordinate Ascent [34], ListNet [27], LambdaMART [35], some of which are compared in our experiments to evaluate their performance on our group event recommendation task. In this paper, we propose a fast pairwise learning-to-rank algorithm, BGR, which shows comparable accuracy and faster time efficiency than state-of-the-art learning-to-rank algorithms. Different from the existing methods learning same ranking model for all groups, the ranking model learned by BGR is group-dependent, i.e., different group may has different ranking model, because the weight of a user in different group may be different.

3 Problem Definition

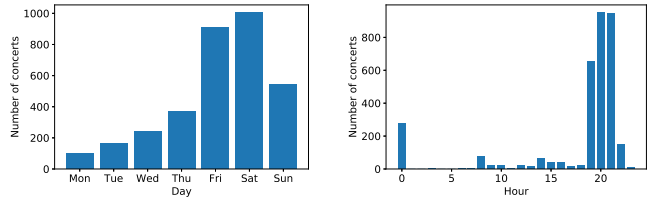
In this subsection, we first describe some key concepts in this paper. Then, we define group event recommendation



(a) Temporal distribution of user attendance in a week

(b) Temporal distribution of user attendance in a day

Fig. 2. Temporal distribution of user attendance.



(a) Temporal distribution of events in a week

(b) Temporal distribution of events in a day

Fig. 3. Temporal distribution of all events.

problem and formalize it as a learning-to-rank problem. In this paper, we use superscript to denote the id of a group.

Definition 1 (User attendance). *For a given user $u \in U$ and an event $c \in C$, a user attendance is defined as a tuple $(u, c, o_c, v_c, t_c, W_c)$, where $o_c \in O$ denotes the organizer of event c, $v_c \in V$ denotes the venue of event c, $t_c \in T$ denotes the start time of event c, $W_c \subset W$ denotes a set of words that introduce event c, W is the vocabulary containing all words in the introduction of events.*

Problem 1 (Group event recommendation). Given a target group $g \in G$, a set of history events C_{hist}, a set of upcoming events C_{cand}, and group members' attendance $\{(u, c,\ o_c, v_c, t_c, W_c) | u \in U_g\}$, where $U_g \subset U$ is the set of members in group g and $c \in C_{hist}$ is an event attended by group g, our goal is to calculate a ranking score $S_{g,c}$ for target group g and each candidate event $c \in C_{cand}$. Then, the candidate events are sorted according to their ranking scores in descending order. Finally, the top K events in the ranked list are recommended to the all members in group g.

Problem 2 (Learning-to-rank for group event recommendation). Given a set of groups $G = \{g^{(1)}, g^{(2)}, \ldots, g^{(M)}\}$, each group $g^{(i)}$ is associated with a list of events attended by members in the group $\boldsymbol{c}^{(i)} = (c_1^{(i)}, c_2^{(i)}, \ldots, c_{n^{(i)}}^{(i)})$, where $c_j^{(i)}$ denotes the jth event and $n^{(i)}$ denotes the number of events the group $g^{(i)}$ have attended. Furthermore, each list of events $c^{(i)}$ is associated with a list of judgments (scores) $\boldsymbol{y}^{(i)} = (y_1^{(i)}, y_2^{(i)}, \ldots, y_{n^{(i)}}^{(i)})$ where $y_j^{(i)}$ denotes the judgment on event $c_j^{(i)}$ with respect to group $g^{(i)}$. The judgment $y_j^{(i)}$ represents the relevance degree of $c_j^{(i)}$ to $g^{(i)}$. A feature vector $\boldsymbol{x}_j^{(i)} = \Psi(g^{(i)}, c_j^{(i)})$ is created for group-event pair $(g^{(i)}, c_j^{(i)})$. The goal of learning-to-rank for group event recommendation is to learn a ranking model $f^{(i)}$ for each group $g^{(i)}$ from training data $(\boldsymbol{y}^{(i)}, \boldsymbol{x}^{(i)})$, where $\boldsymbol{x}^{(i)} = (\boldsymbol{x}_1^{(i)}, \boldsymbol{x}_2^{(i)}, \ldots, \boldsymbol{x}_{n^{(i)}}^{(i)})$. Then ranking model is used to predict a ranking score for a new event $c_k^{(i)}$ and group $g^{(i)}$ by $S(g^{(i)}, c_k^{(i)}) = f^{(i)}(\boldsymbol{x}_k^{(i)})$.

4 Contextual Features Extraction

In this section, we analyze how the user attendance behaviors are impacted by different contextual factors (e.g., temporal, location, content and social factors) and propose methods to capture individual preference considering different contextual influences, respectively. These relevance scores will be used to create feature vector in learning-to-rank for group event recommendation.

4.1 Temporal Features

Time is one of the most important factors that impact user's decision on attending an event. Intuitively, different users may attend events at different time. For example, some users might prefer to attend events on Saturday morning, while others may prefer to attendance on Wednesday evening. To validate our intuition, we analyze the data crawled from DoubanEvent in Beijing (more details about the dataset shown in Section 6.1.1). Figs. 3a and 3b show the temporal distribution of events on each weekday and hour in a day respectively. We observe that most of the events start on weekend in a week, and at 7 PM in a day. We also analyze the temporal distribution of user attendance behaviors. Figs. 2a and 2b show the different users have different temporal pattern on attending events.

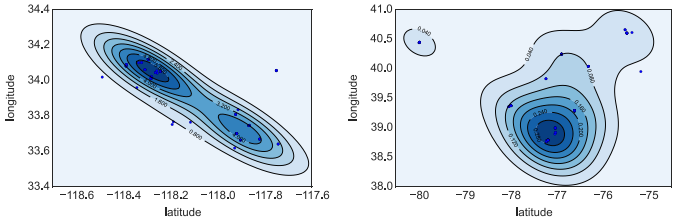
Based on above observation, we can compute the relevance between user and event by measuring the similarity between temporal distributions of user and event. Specifically, each event is represented as a binary vector $\boldsymbol{t}_c \in \{0, 1\}^{7\times 24}$ in the space of all possible days of a week and hours of a day. The one element of $\boldsymbol{t}_c$ is set to 1, if the event start at the particular day and hour. Other elements in $\boldsymbol{t}_c$ will be set to zero. Each user is represented as a normalized temporal distribution of the events attended by the user as follows:

$$\boldsymbol{t}_u = \frac{\sum_{c \in C_u} \boldsymbol{t}_c}{|C_u|}, \tag{1}$$

where C_u denotes the set of events attended by user u. The temporal influenced preference of user u to event c is cosine similarity between $\boldsymbol{t}_u$ and $\boldsymbol{t}_c$. Because the temporal distribution of events is not uniform as shown in Fig. 3, this method tends to predict higher preference of user u to the events that start at popular time period (e.g., weekend). To solve this problem, we propose a new method to calculate relevance score of user and event following the idea of TF-IDF [36], which is a classic algorithm in text analysis and aims to measure the importance of a word in a document. Given user u, $\boldsymbol{t}_u$ can be created as follows:

$$\boldsymbol{t}_u = \frac{\sum_{c \in C_u} \boldsymbol{t}_c}{|C_u|} log(\frac{|C|}{\sum_{c \in C} \boldsymbol{t}_c + 1}), \tag{2}$$

where C is the set of events. The logarithmic part in Eq. (2) is used to reduce the importance of the events that start at the popular time period. Thus, the events that start at weekends or evening have less contribution than others in user's temporal vector $\boldsymbol{t}_u$. Finally, the temporal score that represents the preference for user u to candidate event c influenced by time can be calculated as follows:



(a) user1 (b) user2

Fig. 4. Geographical distribution of user attendance.

$$S_T(u,c) = \frac{\boldsymbol{t}_u \cdot \boldsymbol{t}_c}{\|\boldsymbol{t}_u\| \times \|\boldsymbol{t}_c\|}, \tag{3}$$

where $\|\cdot\|$ denotes the L2 norm of a vector.

4.2 Location Features

Based on the first law of geography [37], is "everything is related to everything else, but near things are more related than distant things", we assume that users are more likely to attend events that held at the venue closed to users, i.e., the probability of a user attending the event near her area is higher than the probability of attending the event far from the user. To validate this intuition, we random select two users from DoubanEvent dataset and show geographical distributions of their attendance behavior in Figs. 4a and 4b respectively. We observe the attendance behaviors of both $user1$ and $user2$ can be fall into two geographical clusters. We note that this observation is very similar to the geographical influence on users' check-in behaviors in location-based social networks [38]. Therefore, we create a probabilistic model using kernel density estimate [39] (KDE) to predict the probability (called geographical score) of a user u attending the event c following [38].

$$S_L(u,c) = \frac{1}{2\pi n h^2} \sum_{\boldsymbol{l}_i \in L_u} exp\left(-\frac{(\boldsymbol{l}_c - \boldsymbol{l}_i)^T (\boldsymbol{l}_c - \boldsymbol{l}_i)}{2h^2} \right), \tag{4}$$

where $\boldsymbol{l}_c = \{lat_c, lon_c\}$ is a two-dimensional geographical coordinate vector, in which two elements represent latitude (lat_c) and longitude (lon_c) of the venue of the event c, $L_u = \{\boldsymbol{l}_c | c \in C_u\}$ is the set of geographical coordinates of the venues, where user u attended events, $n = |L_u|$ is the number of venues in L_u, h is the optimal bandwidth [38], [39]:

$$h = |L_u|^{-\frac{1}{6}} \sqrt{\frac{1}{2} \hat{\boldsymbol{h}}^T \hat{\boldsymbol{h}}}, \tag{5}$$

where $\hat{\boldsymbol{h}}$ is the marginal standard deviation vector of L_u. The numbers on contour lines in Figs. 4a and 4b represent the probabilities of users going to events calculated by KDE. The deeper color of the area in the figure represents the higher probability that the user will attend events in that area.

Except for the geographical influence, the user attending behaviors may be impacted by some inherent characters of the venue, e.g., how many users the venue can hold, the venue is outdoor space or indoor space. These characters of venue may be associated to the kind of events held at. For instance, the stadium that holds a hundred thousand people can be the venue of a big outdoor concert or a football match. The users who like concert or football will be more likely to visit the stadium. These inherent characters of venue can not be modeled by KDE, so we assume that if a user have attended more events at a venue, the user will attend the events at the venue with higher probability. Based on this assumption, we calculate the venue score between user u and event c as follows:

$$S_V(u,c) = \frac{\sum_{c' \in C_u} f(v_{c'}, v_c)}{\sum_{c'' \in C} f(v_{c''}, v_c)} \tag{6}$$

$$f(v,v') = \begin{cases} 1 & if\ v = v' \\ 0 & else \end{cases}, \tag{7}$$

where C_u is the set of events attended by user u. v and v' denote two venues.

4.3 Content Features

Content-based method has been widely used in many recommender systems. It is also a promising approach for event recommendation, due to the ability of alleviating cold-start problem [2]. Generally, the content information of an event may be in different forms, e.g., photos, video and text. In our problem, the content information is a textual introduction of the event. Thus, some text analysis approaches (e.g., TF-IDF [40] and LDA [41]) can be used to extract content features. However, TF-IDF and LDA may miss some semantic information because they are based on bag-of-words model, which ignores the sequence of the words in a document.

In this subsection, we utilize word embedding technique to extract semantic information from the text content of events by capturing the sequence of the words and use the semantic information to measure the relevance of an event to a user influenced by the content of event. Word embedding technique receives increasing attention recently and has been successful used in many domains including information retrieval, natural language processing and recommender systems. The basic idea of word embedding is to map a word from the vocabulary space to a low-dimensional space. Then each word is represented as a vector that reserves some semantic information of the word. One of the most popular methods for word embedding is word2vec [42] and we use it in this paper to map each word in the text introductions of the events to a low-dimensional vector. Specifically, we regard the introduction of an event as a document. Then, the corpus made up of all documents is input to word2vec. The output of word2vec is low-dimensional vector for each word in the corpus. Finally, we calculate the content score for user u and event c as follows:

$$S_C(u,c) = cosine(\boldsymbol{p}_u, \boldsymbol{q}_c) \tag{8}$$

$$\boldsymbol{q}_c = \frac{\sum_{w \in W_c} \boldsymbol{z}_w}{|W_c|} \tag{9}$$

$$\boldsymbol{p}_u = \frac{\sum_{c' \in C_u} \boldsymbol{q}_c}{|C_u|}, \tag{10}$$

where $\boldsymbol{p}_u, \boldsymbol{q}_c$ denote the user content profile and event content features, respectively. $\boldsymbol{z}_w$ denotes the low-dimensional vector of word w obtained by word2vec.

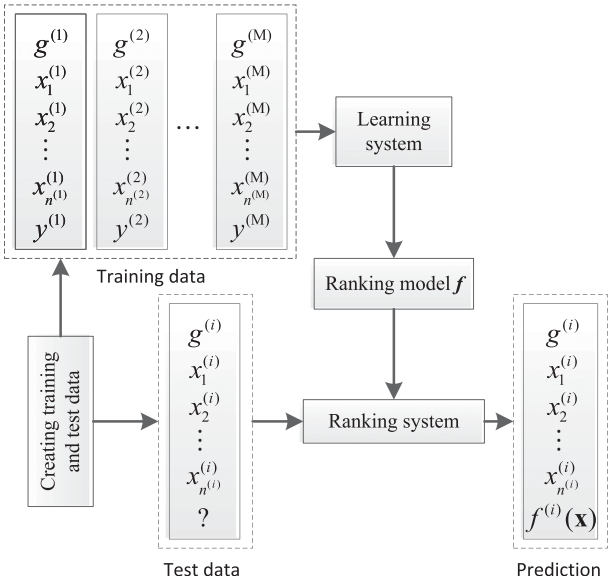


Fig. 5. Learning-to-rank flow for group event recommendation.

$cosine(\boldsymbol{a}, \boldsymbol{b})$ measures cosine similarity between two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$.

4.4 Social Features

Many previous works show that social information is helpful to improve recommendation performance [22]. With the popular of EBSNs, the social relationships between users and organizers become available. A user can follow an organizer on DoubanEvent or join a group that organizes events on Meetup. In this section, we use the social relationships between users and organizers to alleviate cold-start problem in event recommendation. Our method is based on two assumptions. (1) A user is more likely to attend the events created by the organizer who has social relations to the user. This is called direct social influence. (2) A user prefer to attend the event that is organized by the same organizer who has created the events that have been attended by the user before. This is called indirect social influence. Based on above two assumptions, we propose two methods considering direct and indirect social relations between users and organizers, respectively. Specifically, given user u and event c, the direct social score S_{DS} and indirect social score S_{IS} of user u to event c are calculated as follows:

$$\begin{aligned} S_{DS}(u,c) &= \sum_{h \in H_u} sim(h_c, h) \\ S_{IS}(u,c) &= \sum_{c' \in C_u} sim(h_c, h_{c'}), \end{aligned} \tag{11}$$

where H_u denotes the set of organizers with social relations to user u, h_c is the organizer who creates event c. The similarity $sim(h_1, h_2)$ between two organizers h_1 and h_2 is defined as

$$sim(h_1, h_2) = \begin{cases} 1 & h_1 = h_2 \\ 0 & h_1 \neq h_2 \end{cases}. \tag{12}$$

5 Learning-to-Rank for Group Event Recommendation

We have formalized group event recommendation as a learning-to-rank problem in Section 3. The key component of our problem is defining the feature vector $\boldsymbol{x}_j^{(i)}$ and judgment $y_j^{(i)}$ for each group-event pair $(g^{(i)}, c_j^{(i)})$ for training and test purpose respectively. Fig. 5 shows the flow of learning-to-rank for group event recommendation. We first create training data and test data for learning ranking model and prediction ranking score respectively. The training data consists of $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$ of each group. In learning phase, the learning system utilizes learning-to-rank algorithm to learn a ranking model $f^{(i)}$ for each group $g^{(i)}$ from training data. In prediction phase, the ranking score of an upcoming event $c_k^{(i)}$ to the group $g^{(i)}$ is calculated by $f^{(i)}(x_k^{(i)})$, which is finally used to generate recommendation list for group $g^{(i)}$.

5.1 Creating Training and Test Data

In this subsection, we describe how to create training data to learn a ranking model for each group using learning-to-rank technique and how to create test data to predict ranking score of an upcoming event to the group using its ranking model.

One of most important component for creating training data is how to measure the judgment $y_j^{(i)}$, which represents the relevance degree of event $c_j^{(i)}$ to group $g^{(i)}$. A simple method is that if the members in group $g^{(i)}$ have attended an event $c_j^{(i)}$, we have $y_j^{(i)} = 1$, otherwise, $y_j^{(i)} = 0$. Because the members in a group attended very few events or even no event together in history, this method may lead to that some groups have no relevant event, i.e., $\boldsymbol{y}^{(i)} = \boldsymbol{0}$. To solve this problem, we propose a method to measure the relevance degree of an event to a group. Specifically, we assume that the events attended by a part of members are also relevant to the group and the events attended by more group members are more relevant than those attended by fewer group members. For example, suppose that two members $u_1^{(i)}, u_2^{(i)}$ in group $g^{(i)}$ attended event $c_1^{(i)}$. Event $c_2^{(i)}$ was attended by $u_1^{(i)}$ and not attended by $u_2^{(i)}$. The relevance degree of event $c_1^{(i)}$ to group $g^{(i)}$ is higher than event $c_2^{(i)}$ to group $g^{(i)}$, i.e., $y_1^{(i)} > y_2^{(i)}$. We define the judgment $y_j^{(i)}$ for event $c_j^{(i)}$ to group $g^{(i)}$ as follows:

$$y_j^{(i)} = \sum_{u \in U_g} I_{C_u}(c_j^{(i)}), \tag{13}$$

where U_g denotes the set of members in group g, C_u is the set of events attended by user u, $I_A(x)$ is indicator function which returns 1 if $x \in A$, otherwise returns 0. In fact, $y_j^{(i)}$ equals to the number of group members of $g^{(i)}$ who attended event $c_j^{(i)}$. Note that we only assign judgment for each group-event pair in training data rather than those in test data, because the judgment of each group-event pair in test data needs to be predicted by ranking model.

After the judgment $y_j^{(i)}$ is defined, we describe how to create feature vector $x_j^{(i)}$ for each group-event pair $(g^{(i)}, c_j^{(i)})$ in both training and test data. For a given group $g^{(i)}$ and event $c_j^{(i)}$, the feature vector $x_j^{(i)} \in \mathbb{R}^D$, where $D = m^{(i)} \times F$, $m^{(i)}$ denotes the number of members in group $g^{(i)}$ and F

denotes the number of contextual scores. We consider six contextual scores (i.e., $S_T, S_L, S_V, S_C, S_{DS}, S_{IS}$) in this paper. The feature vector $x_j^{(i)}$ is defined as follows:

$$x_j^{(i)} = (p(u_1^{(i)}, c_j^{(i)}), p(u_2^{(i)}, c_j^{(i)}), \ldots, p(u_{m^{(i)}}^{(i)}, c_j^{(i)})), \tag{14}$$

where $\{u_1^{(i)}, u_2^{(i)}, \ldots, u_{m^{(i)}}^{(i)}\} \subset U$ are the members of group $g^{(i)}$. Given a user u and event c, $p(u, c)$ is defined as a list of contextual scores: $p(u, c) = (S_T(u, c), S_L(u, c), S_V(u, c), S_C(u, c), S_{DS}(u, c), S_{IS}(u, c))$.

Because the ranges of different contextual scores may be different, we need to normalize each contextual score to create feature vector. For example, S_L outputs a probability of the user attend an event considering geographical influence, the range of which is 0 to 1, while S_C computes cosine similarity between two real value vectors and the range of S_C is -1 to 1. We normalize the different contextual scores into same scale as follows (we only take S_T as an example):

$$S_T(u, c) = \frac{S_T(u, c)}{max(S_T(u, c))}. \tag{15}$$

For each group, we trade a pair of feature vector and judgment score $(x_j^{(i)}, y_j^{(i)})$ as a training sample. N training samples, where N is the number of events. Thus, $M \times N$ samples should be created for all groups. Because the members in a group usually have attended very limited events, most of the events are not relevant to the group. To reduce this imbalance between the number of relevance and irrelevance events, we propose a method based on scheduling conflict to select the events that are not relevant to the group for creating training samples. Intuitively, a user can not attend two events that start at the same time. Moreover, a user attends an event after very short time of attending another event with a relative low probability. Based on these intuitions, we set a time threshold to sample irrelevent events from the set of events that have not been attended by any members in a group. Specifically, given a group g and a relevant event c relevant to g, we select a event c' that are unrelated to the group g if satisfy $t'_c \in (t_c - \beta, t_c + \beta)$, where β is time threshold.

The benefit of our group event recommendation framework, i.e., GERF, is twofold. 1) Combining multiple contextual influence to achieve better recommendation performance than only considering single influence. 2) Combining group members' preferences to model group preference, which enables to make recommendation for group of users. Moreover, the feature vector for learning-to-rank consists of the group members' contextual scores. The new contextual influence can be incorporated in our feature vector easily, which makes our GERF extensible.

5.2 Bayesian Group Ranking

Our proposed GERF can adopt existing learning-to-rank algorithms to learn a ranking model for each group. According to the definition of feature vector in Eq. (14), we know that different groups[3] have different features space. Thus, the ranking model is group-dependent and we need to learn a ranking model for each group instead of for all groups. This is different with the traditional learning-to-rank task, where there is a same feature space for all queries. Although the existing learning-to-rank algorithms can be used to learn a ranking model for each group individually, the training process is inefficient. To solve this problem, we propose a novel learning-to-rank algorithm for group event recommendation task based on Bayesian Personalized Ranking (BPR) [43], called Bayesian Group Ranking (BGR). Different with the existing approaches to learning a ranking model for each group individually, BGR provides a unified optimization criterion for all groups.

Following the user specific pairwise preference between a pair of items [43], we use event pairs as training data and optimize for correctly ranking event pairs. Specifically, if the event c_j is more relevant to the group $g^{(i)}$ than event c_k c_k, i.e., $y_j^{(i)} > y_k^{(i)}$, the group $g^{(i)}$ prefers c_j to c_k. Based on this assumption, we can create training data $D_S : G \times C_{hist} \times C_{hist}$ as follows:

$$D_S = \{(g^{(i)}, x_j^{(i)}, x_k^{(i)}) | y_j^{(i)} > y_k^{(i)} \wedge g^{(i)} \in G\}. \tag{16}$$

We adopt two methods to ensure enough training data to train. First, we assume that the events attended by a part of members are also relevant to the group and the events attended by more group members are more relevant to the group. Based on this assumption, we can find more relevant events for a group than assuming that the relevant events must be attended users by all members. Second, we create training data according to Eq. (16). Assume that there are N events for training and M relevant events for group $g^{(i)}$. We can get $N \times (N - M)$ training samples to train $\theta^{(i)}$. That is enough for training $\theta^{(i)}$ of the group.

We assume that the ranking model $f^{(i)}$ for group $g^{(i)}$ is a linear model, i.e., $f^{(i)}(c_j^{(i)}) = \theta^{(i)} x_j^{(i)}$, where $\theta^{(i)}$ denotes the parameter vector of ranking model $f^{(i)}$ with same dimension as $x_j^{(i)}$. In BPR, the prior density $p(\theta)$ is a normal distribution with zero mean and covariance matrix λI, where I denotes identity matrix. To ensure enough training data for a group that has few positive feedbacks, we utilize the groups with common members to the current group to help learn the model parameters for current group. Specifically, we assume that the prior density $p(\theta)$ is a normal distribution as follows.

$$p(\theta^{(i)}) \sim N\left(\sum_{g^{(q)} \in F_{g^{(i)}}} Y_{i,q}^* \theta^{(q)}, \lambda I\right), \tag{17}$$

where $F_{g^{(i)}}$ denotes the set of groups that have common members to the current group $g^{(i)}$, and Y denotes group-group matrix, where $Y_{i,q}$ denotes the number of common members in group $g^{(i)}$ and $g^{(q)}$. Each of the rows of the matrix Y is normalized to 1, resulting in the new matrix Y^* with $Y_{i,q}^* \propto Y_{i,q}$ and $\sum_q Y_{i,q}^* = 1$ for each group $g^{(i)}$. The goal of BGR is to maximize posterior probability as follows.

$$\max_{\Theta} \sum_{(g^{(i)}, x_j^{(i)}, x_k^{(i)}) \in D_S} \log \sigma(x_{j,k}^{(i)}) - \lambda \sum_{i=1}^{M} \|\theta^{(i)}\|^2, \tag{18}$$

where $\Theta = \{\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(M)}\}$ is the set of parameters of ranking models. $x_{j,k}^{(i)} = \theta^{(i)} x_j^{(i)} - \theta^{(i)} x_k^{(i)}$ is the difference

3. If two groups have different members, we regard the two groups as different groups.

between two feature vectors $x_j^{(i)}$ and $x_k^{(i)}$. λ is the model specific regularization parameters. σ is the Logistic function defined as follows:

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{19}$$

The learning algorithm of BGR is shown in Algorithm 1. In each iteration, we choose a triple $(g^{(i)}, x_j^{(i)}, x_k^{(i)})$ randomly (uniformly distribution) from training data D_S and update the model parameter vector $\theta^{(i)}$ for group $g^{(i)}$ until convergence. α denotes the learning rate in Algorithm 1.

Algorithm 1. Parameter Estimation for BGR

Input: Training data D_S.
Output: The parameters of each ranking model $\Theta = \{\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(M)}\}$.
Initialize Θ randomly.
while *not convergence* **do**
  Sample $(g^{(i)}, x_j^{(i)}, x_k^{(i)})$ from D_S.
  $\theta^{(i)} = \theta^{(i)} + \alpha((1 + e^{x_{j,k}^{(i)}})^{-1}(x_j^{(i)} - x_k^{(i)}) - \lambda\theta^{(i)})$
end

Because all members' contextual scores are regarded as the features for training the ranking model of the group, the group preference aggregation and combination of different context influences can be processed concurrently. The learned ranking model $f^{(i)}$ of group $g^{(i)}$ is used to predict the ranking score for group $g^{(i)}$ and an upcoming event c. For each group, we compute the ranking score for each upcoming event and sort these events by their ranking scores in descending order. Then the top K events in the ranked list are suggested to the group. The whole process of group event recommendation using GERF is summarized in Algorithm 2.

Algorithm 2. Group Event Recommendation Using GERF

Input: a set of users U, a set of groups G, history events C_{hist}, candidate events C_{cand}.
Output: Ranking score $S(g, c)$ of each group $g \in G$ and candidate event $c \in C_{cand}$.
for *each u in U* **do**
  for *each c in* C_{hist} **do**
    Compute $S_T(u, c)$ by Eq. (3).
    Compute $S_L(u, c)$ by Eq. (4).
    Compute $S_V(u, c)$ by Eq. (6).
    Compute $S_C(u, c)$ by Eq. (8).
    Compute $S_{DS}(u, c)$ and $S_{IS}(u, c)$ by Eq. (11).
  end
end
Create training data according to Eqs. (13) and (14).
for *each* $g^{(i)}$ *in G* **do**
  Learn the parameter of ranking model $\theta^{(i)}$ for group $g^{(i)}$ utilizing BGR.
for *each* $g^{(i)}$ *in G* **do**
  for *each* c_j *in* C_{cand} **do**
    compute $S(g^{(i)}, c_j^{(i)})$ by $\theta^{(i)} \cdot x_j^{(i)}$.
  end
end

Time Complexity Analysis. The group event recommendation based on GERF can be split into offline modeling and online recommendation. Offline modeling consists of features extraction and training the ranking models and takes much more cost than online recommendation, which uses the learned ranking models to predict ranking score for each group and candidate event and the ranked recommendation list is generated based on the ranking scores. We assume that the time complexity of each contextual feature extraction method from Sections 4.1, 4.2, 4.3, and 4.4 is O(1). Thus, predicting each context-aware score for user $u \in U$ and event $c \in C_{hist}$ requires $O(D)$ operations, where D is the number of ratings in the dataset. Creating training data for learning-to-rank needs $O(M)$ operations, where M is the number of groups. Learning the ranking models for all groups by BGR requires $O(XF)$ operations, where X is the number of training samples for each group and F is the dimension of feature vector (i.e., $x_j^{(i)}$ for pair of group $g^{(i)}$ and $c_j^{(i)}$), the dimension of which depends on the size of group $g^{(i)}$. Thus, the time complexity of offline modeling is $O(D + MXF)$. Online recommendation requires $O(F)$ operation (i.e., $\theta^{(i)} \cdot x_j^{(i)}$) and it is linear to the size of group.

6 Experiments

In this section, we first describe our experimental settings, and then illustrate the experimental results.

6.1 Experimental Settings

6.1.1 Dataset

We adopt two real-world datasets crawled from Douban-Event and Meetup, two popular event-based social networks (EBSNs), respectively. EBSN is an online platform where users can create, find and share social events (e.g., parties and exhibitions). A user in EBSNs can RSVP[4] to upcoming event, i.e., "Yes" denotes user will participate the event and "No" denotes user will not participate. We crawl events in Beijing from DoubanEvent during Sep 2016 and Dec 2016. The Meetup dataset is collected and released by [22]. We extract the events held in Chicago during Jan 2013 and June 2014 from Meetup dataset for our experiments. For each event in both datasets, we obtain its organizer, participants, content information, geographical venue and start time. Because there is no group information in both datasets, we have to generate some synthetic groups for our group event recommendation task. Before that, we first remove the users who attended less than 5 events to avoid noise and select the events that have at least two participants to facilitate group formation. Finally, the statistical information of the two datasets we used is shown in Table 1.

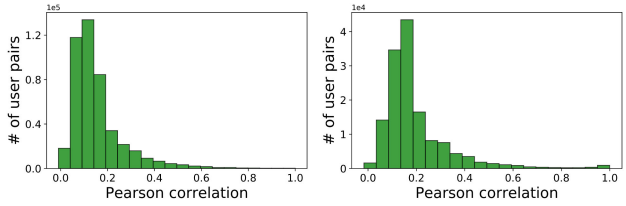
6.1.2 Evaluation Method

In order to simulate the realistic scenario of group event recommendation, where the events required to be recommended are cold-start, we propose a modified time-dependent cross-validation method [44]. For each dataset, we first sort the events chronologically according to their start time. Because the Meetup dataset contains the events held in 16 months, i.e., from Jan 2013 to April 2014, we perform 8-fold cross-validation on this dataset. Specifically, the

4. RSVP stands for "please response".

TABLE 1
Statistical Information of Our Datasets

Datasets	Users	Events	RSVPs	Venues	Organizers	Sparsity
Meetup	4982	9291	59196	1669	672	99.87%
Douban	1146	1050	10697	539	390	99.11%



(a) Meetup (b) DoubanEvent

Fig. 6. User-to-user similarity.

first experiment (fold) takes the events held in first two months for training and remaining events as candidates for testing. The second experiment takes the events in first four months for training and remaining for testing, and so on. Because each experiment have more events for training than the previous experiment, the data sparse of each fold is different. For example, the training data of first fold is more sparse than that of second fold. Thus, the effectiveness of recommendation methods on different degrees of data sparsity can be evaluated. Similarly, we perform 3-fold cross-validation on DoubanEvent dataset, which containing the events held between Sep 2016 and Dec 2016. The first experiment uses the events in first month for training, the second experiment exploit the events in first two months for training, and so on.

Note that both DoubanEvent and Meetup datasets do not contain any predefined groups, so we generate some synthetic groups for evaluating the performance of group event recommenders. Because there are few datasets containing group information, generating synthetic groups is a common protocol for the evaluation of group recommender systems [5], [7], [9], [12], [15]. We generate groups based on user similarity between group members following the approach in [45]. Specifically, the groups are defined as those containing users with user-to-user similarity higher than certain threshold. We set the threshold as average of user-to-user similarity following [9]. Figs. 6a and 6b shows the user-to-user similarity distribution on Meetup and DoubanEvent datasets, respectively. The similarity is calculated by Pearson correlation coefficient (PCC). We find that the average user-to-user similarity of DoubanEvent dataset is 0.197 and average user-to-user similarity of Meetup dataset is 0.155. Therefore, the thresholds for group generation of DoubanEvent and Meetup datasets are set to 0.197 and 0.155, respectively. For each dataset, we generate 2000 groups for each group size from 2 to 6.

To evaluate the recommendation performance of GERF, we adopt precision, recall and F1 score, which are standard metrics in information retrieval and have been widely used to measure accuracy of group recommender systems [5], [7], [24]. Specifically, for each group attendance record $(g, c, o_c, v_c, t_c, W_c)$ in $\mathcal{D}_{test}$:

TABLE 2
Features of Different Methods in Our Experiments

Methods \ Features	Temporal Effect	Geographical Effect	Venue Effect	Content Effect	Social Effect	Group recommendation
BPR						
GERF	•	•	•	•	•	•
GLFM	•	•	•		•	
CBPF			•	•	•	
COM		•				•
HBGG		•				•

1) We first compute the ranking scores for all events in candidate set C_{cand} and each group g.
2) We generate a ranked list of events by ordering all events in C_{cand} according to their ranking scores.
3) We form a Top-K recommendation list by selecting the K top ranked events from the list for group g. If an event in the Top-K list of a group is contained in $\mathcal{D}_{test}$ corresponding to the group, we have a hit. Otherwise, we have a miss.
4) The Precision@K and Recall@K are defined as follows:

$$P@K = \frac{\#hit}{K \times |G|} \tag{20}$$

$$R@K = \frac{\#hit}{|\mathcal{D}_{test}|}, \tag{21}$$

where $\#hit$ denotes the number of hits for all groups, $|G|$ is the number of groups, and $|\mathcal{D}_{test}|$ denotes the number of all test cases. Based on the definition of Recall and Precision, we can calculate F1 score as follows:

$$F1@K = \frac{2 \times P@K \times R@K}{P@K + R@K}. \tag{22}$$

The higher of Precision@K, Recall@K and F1@K scores, the better recommendation method performs.

6.1.3 Comparative Approaches

Recommendation Effectiveness. To the best of our knowledge, there are few studies on recommending events for group of users. Most of existing group recommendation methods may not perform well on group event recommendation task due to the cold-start problem, where the events that need to be recommended have not interacted with users. We compare our GERF with the following five state-of-the-art approaches. Because three of them are designed to generate event recommendations for individuals, we utilize aggregation strategy to combine their results for individuals into group recommendation. Table 2 lists the characteristics of these methods.

BPR [43] is a widely used pairwise optimization framework for many models, such as matrix factorization and k-nearest-neighbor. Our proposed BGR follows the idea of BPR to learn the ranking model. We compare our method with BPR-MF, i.e., matrix factorization optimized by BPR. Because BPR-MF does not designed to generate recommendation for groups, we adopt average strategy to combine the

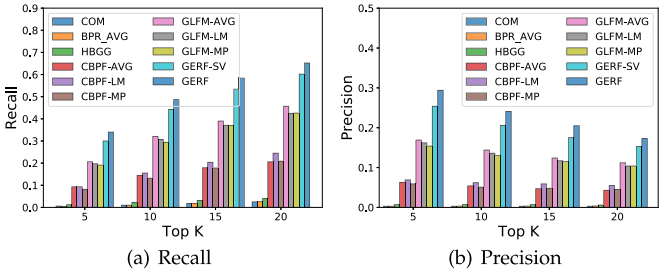


Fig. 7. Top-K performance on Meetup dataset.

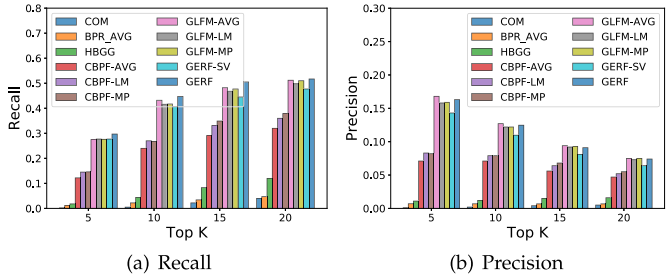


Fig. 8. Top-K performance on DoubanEvent dataset.

individual's score produced by BPR-MF into group's score. We refer this method as BPR-AVG in our experiments.

CBPF [2] is a collective Bayesian Poisson factorization model for cold-start local event recommendation. CBPF can capture social (interaction between users and organizer), location and content influence in a unified framework. Because CBPF is designed for recommending events to individuals and can not generate suggestions for groups directly, we aggregate individuals' scores produced by CBPF into ranking score for group with three aggregation strategies: AVG, LM and MP. We finally get three variant versions of CBPF, i.e., CBPF-AVG, CBPF-LM, CBPF-MP, respectively.

GLFM [3] is dual-perspective latent factor model for group-aware event recommendation, where the group is the organizer of event. The temporal, venue, popularity and social influences are considered. GLFM is not specially designed for suggesting events to groups. Similarly, we use three aggregation strategies to combine group members's scores generated by GLFM, and get three variant versions of GLFM, i.e., GLFM-AVG, GLFM-LM, GLFM-MP, respectively.

COM [5] makes group recommendations by modeling the generative process of group activities. COM can incorporate both users' feedback and personal considerations of content factors. There are two kinds of content factors, i.e., geographical distance and movie cast, considered in [5] for venue recommendation and movie recommendation, respectively. In our experiments, we only incorporate geographical distance into COM for our group events recommendation task.

HBGG [15] jointly models group geographical topic, group mobility regions and social relations among groups, users and items. HBGG is designed to recommend venues to groups of users, but we use it to recommend events to groups in our experiments.

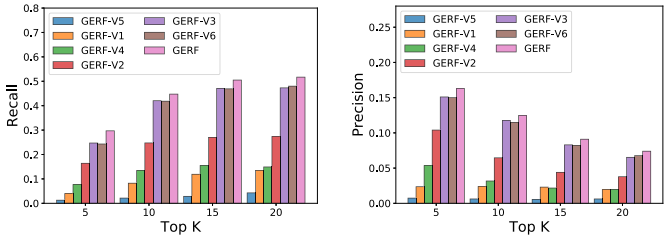
Note that GLFM does not consider content influence, which can be incorporated in our method and CBPF. Some studies show that temporal influence is not effectiveness for event recommendation. Therefore, to ensure fair comparison, we implement GLFM and CBPF only considering indirect social influence and venue (location) influence and compare with our method GERF-SV, which only uses indirect social and venue features. Moreover, in order to evaluate the benefit of different contextual factors, we design six variant versions of GERF. GERF-V1 is the first variant version of GERF where we only consider temporal factor according to Section 4.1. GERF-V2 only considers geographical influence according to Eq. (4). GERF-V3 only incorporates the venue features. GERF-V4 only utilizes content features for learning-to-rank. GERF-V5 and GERF-V6 only consider direct social influence and indirect social influence according to Eq. (11), respectively. We set regularization parameter λ to 0.025 and the learning rate is $\alpha = 0.05$ following [3].

6.2 Recommendation Effectiveness

In this subsection, we report the comparison results between our methods and other state-of-the-art methods with well-tuned parameters. In this experiment, GERF-SV and GERF employs our proposed BGR algorithm to learn a ranking model for each group. Figs. 7 and 8 show the Top-K recommendation performance on Meetup and DoubanEvent datasets, respectively. We only report precision and recall scores when K is set to 5, 10, 15, 20, as a greater value of K is usually ignored for a typical Top-K recommendation task.

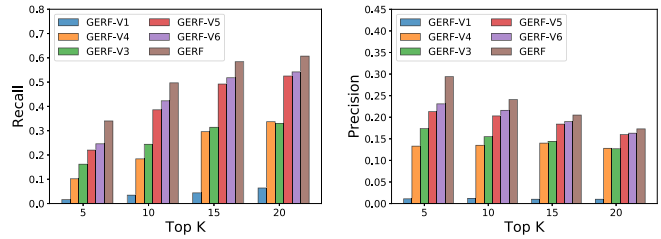
From Figs. 7a and 7b, we observe that both GERF-SV and GERF outperform other baseline approaches significantly in terms of the precision and recall scores on the Meetup dataset. Fig. 8 show that GLFM-based methods achieve higher precision and recall than GERF-SV on DoubanEvent dataset. The possible reason is that our proposed venue and social feature extraction methods does not perform well on sparse data (i.e., DoubanEvent dataset). The possible reason is that the latent factor model adopted by GLFM is more suitable to handle with sparse data and our method benefits from relative low sparsity of event-venue matrix on Meetup dataset: the average number of events held at each venue is 35.47 on Meetup dataset, while that is 19.84 on DoubanEvent dataset. Moreover, the sparsity of event-organizer matrix on Meetup dataset is also lower than that on DoubanEvent dataset, i.e., the average number of events held by a organizer is 88.09 on Meetup dataset, while that is 27.43 on DoubanEvent dataset. Among the baseline methods, we find that GLFM-based methods outperforms than CBPF-based methods on both Meetup and DoubanEvent datasets. We find that HBGG performs better than COM and BPR-AVG. This is because the HBGG can explicitly exploit geographical information to alleviate cold-start problem, while the geographical influence is only incorporated as prior distribution. BPR-AVG achieves very low Precision and Recall values on the both two datasets, because it only exploiting the users event attendance records and does not consider any additional information to alleviate cold-start problem.

We also find that there are no best aggregation strategy for GLFM-based and CBPF-based methods. As the conclusion in [12], performance of aggregation strategy depends on the individual recommendation algorithms (GLFM and CBPF in our case) and specific dataset. Fig. 7 shows that the



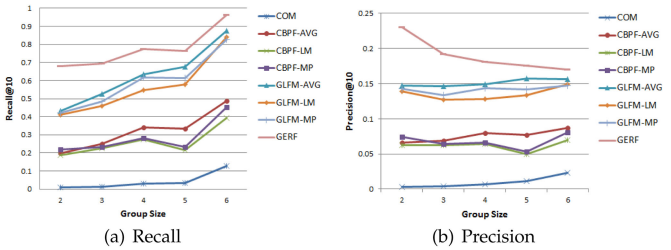
(a) Recall (b) Precision

Fig. 9. Impact of different factors on DoubanEvent dataset.



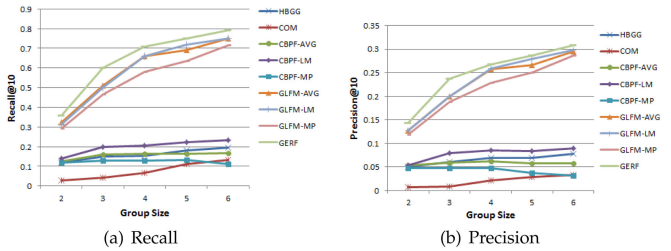
(a) Recall (b) Precision

Fig. 10. Impact of different factors on Meetup dataset.



(a) Recall (b) Precision

Fig. 11. Impact of group size on Meetup dataset.



(a) Recall (b) Precision

Fig. 12. Impact of group size on DoubanEvent dataset.

effectiveness rank of aggregation strategy for GLFM is $AVG > MP > LM$, while the order of aggregation strategies for CBPF is $MP > AVG > LM$. Fig. 8 reports the performance order of aggregation strategy on GLFM is $AVG > LM > MP$ and $LM > AVG > MP$ on CBPF.

6.3 Influence of Different Contextual Factors

In this subsection, we carry out study showing the benefits of different contextual factors considered in GERF, i.e., temporal (F1), geographical (F2), venue (F3), content (F4), direct social (F5) and indirect social (F6) factors. We compare GERF with its six variant versions proposed in Section 6.1.3, and comparison results on DoubanEvent and Meetup datasets are shown in Figs. 9 and 10, respectively. Note that we do not report the performance of GERF-V2 on the Meetup dataset since the geographical address (i.e., latitude and longitude) of event is not available in this dataset.

From the results, we first observe that GERF consistently outperforms the six variant versions in terms of both precision and recall, indicting the benefits brought by combining multiple contextual influences. Moreover, we find that the contribution of each factor on improving recommendation accuracy is different. Specifically, according to the importance of the six contextual factors on the DoubanEvent dataset, they can be ranked as follows: $F6 >= F3 > F2 > F4 > F1 > F5$, while on the Meetup dataset they can be ranked as: $F6 > F5 > F3 > F4 > F1$. We observe that $F6$ is the most important factor on both datasets, but the importance of $F5$ is very different on DoubanEvent and Meetup datasets. This is because the social relations between users and organizers in DoubanEvent is more sparse than those in Meetup dataset: 1) Only 26.5 percent users have social relations to the organizers of events in the DoubanEvent, while 99 percent users have the relations to organizers in the Meetup dataset. 2) Each user has 5.54 relations on average to organizers in Meetup dataset, while each user has 1.89 relations on average in DoubanEvent dataset. Furthermore, we find that the performances of GERF-V1 are relatively low on both datasts, indicting that temporal factor is not very effective for group event recommendation task.

6.4 Influence of the Group Size

Group size is an important factor that impacts the performance of group recommendation [5], [12]. In this subsection, we compare the performance of different group recommendation methods for groups with different sizes. We conduct performance study of all methods that make recommends for groups with 2-6 members. We fix the length of recommendation list as 10 and only report the Recall@10 and Precision@10 scores. Figs. 11 and 12 show the impact of the group size on Meetup dataset and DoubanEvent dataset, respectively. We first observe that our method outperforms other baseline methods irrespective of the group sizes on both dataset. Among baseline approaches, we find that CLFM-based methods outperforms CBPF-based methods and HBGG is superior against COM for all groups. Moreover, the recall scores of most methods are improved with the increase of group size. The possible reason is that the number of the ground truth of larger groups is less than small groups, i.e., the denominator of Eq. (21) becomes smaller with the size of group increased.

6.5 Effectiveness and Efficiency of BGR

In this subsection, we evaluate the effectiveness of our proposed BGR algorithm by comparing BGR with AVG method that takes ranking score as the average of feature values and five state-of-the-art learning-to-rank algorithms (i.e., RankSVM (RS) [16], [46], LambdaMart (LBM) [35], AdaRank (AR) [47], Coordinate Ascent (CA) [34], ListNet (LN) [27]). RankSVM is implemented by SVM^{rank}[5] and other learning-to-rank algorithms are provided by RankLib.[6] Table 3 shows Precision@10, Recall@10 and F1 scores on the Meetup and DoubanEvent datasets. From the

5. https://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html
6. https://sourceforge.net/p/lemur/wiki/RankLib/

TABLE 3
Comparison of Learning-to-Rank Algorithms

	Meetup dataset			Douban dataset		
	P@10	R@10	F1	P@10	R@10	F1
AVG	0.2	0.433	0.274	0.103	0.384	0.162
AR	0.189	0.42	0.261	0.093	0.367	0.148
LN	0.207	0.444	0.282	0.111	0.411	0.174
RS	0.205	0.442	0.281	0.111	0.413	0.175
LBM	0.201	0.437	0.276	0.109	0.404	0.172
CA	0.206	0.442	0.281	0.111	0.412	0.175
BGR	0.206	0.442	0.281	0.11	0.407	0.173

TABLE 4
Memory Consumption of Learning-to-Rank Algorithms on the Meetup Dataset

Methods	Memory Consumption (MB)
AdaRank	644
RankSVM	86
LambdaMart	757
ListNet	755
Coordinate Ascent	760
BGR	605

TABLE 5
Efficiency of Learning-to-Rank Algorithms on DoubanEvent Dataset

Methods	Training Time (s)	Prediction Time (s)	Total Time (s)
BGR	1.78	10.05	11.83
RS	2.33	102.51	104.84
AR	5.22	17.57	22.79
LBM	20.35	18.79	39.14
CA	24.43	12.57	37
LN	150.26	14.6	164.86

TABLE 6
Running Time on the DoubanEvent Dataset

Methods	Training Time (s)	Prediction Time (s)
GERF	62.11	0.159
GLFM-AVG	1.64	0.564
GLFM-LM	1.54	0.577
GLFM-MP	1.54	0.573
CBPF-AVG	291	0.562
CBPF-LM	353.52	0.513
CBPF-MP	340.87	0.55
HBGG	111.56	564.801
COM	2.94	0.341

results, we observe that ListNet achieves highest F1 score on the Meetup dataset and RankSVM and Coordinate Ascent outperform other algorithms on the DoubanEvent dataset. Moreover, the performance gaps between different learning-to-rank algorithms are small and all learning-to-rank algorithms, except for AdaRank, perform better than AVG, which takes the average of feature values as the ranking score. This indicates the superiority of incorporating learning-to-rank techniques into group event recommendation task.

We also compare the efficiency of BGR and other state-of-the-art learning-to-rank algorithms. All the methods run on a Windows 7 with 8G RAM and Core i5 CPU. Table 5 reports the training, prediction and total time consumption of learning-to-rank algorithms. We observe that BGR spends less training and prediction time than other five state-of-the-art algorithms. There is a very small gap between training time of BGR and RankSVM (1.78 versus 2.33 seconds), but our model is much more efficient in terms of the prediction time (10.05 versus 102.51 seconds). Moreover, we find that listwise methods (i.e., LBM, CA, LN) require much more time than pairwise methods (i.e., BGR, RS, AR) for training their models, because listwise methods aim to optimize listwise loss function which is more complexity than pairwise loss function minimized by pairwise approaches. Furthermore, we also evaluate the space efficiency of methods. Table 4 reports the memory consumption during the model training. We observe that BGR consumes least memories except for RankSVM, which only uses very few memories compared to other learning-to-rank algorithms. The possible reason is that RankSVM is implemented with C and other algorithms are implemented in Java, which usually consumes more memories than C. Moreover, we find that listwise learning-to-rank algorithms, i.e., LambdaMart, ListNet and Coordinate Ascent, requires more memories than pairwise algorithms, i.e., AdaRank, RankSVM and BGR. This is because a list of objects (e.g., events in our task) is taken as instance for training in listwise approaches, while pairwise methods take object pairs as instance in learning.

6.6 Recommendation Efficiency

In this subsection, we evaluate the whole group recommendation efficiency of GERF, i.e., the total running time of feature extraction, creation of training and test data, and learning-to-rank. Table 6 shows the running time of different group event recommendation methods. We observe that GLFM-based methods and COM outperform others. This is because they do not consider the content information of events and modeling content influence is very costly due to the large number of words in the text content. Though the whole training process of GERF is slower than GLFM-based methods, this process can be split into several parts that can be parallel and incremental implemented in a very simple way. Specifically, six contextual scores (i.e., $S_T, S_L, S_V, S_C, S_{DS}, S_{IS}$) can be distributed implemented on 6 nodes. Moreover, we find that the most time costly part in feature extraction process of GERF is learning the embedding represents of words for calculating S_C using word2vec, which costs 58.36 seconds. In fact, word2vec only need to perform once when the number of words in vocabulary is large enough. In this case, the content feature extraction can be incremental. For instance, if a new event comes and each word in its text content exists in the vocabulary, then the embedding represent of the word can be directly obtained from the saved results produced by word2vec in history.

We also report prediction time of GERF and other methods in Table 6. The results show that GERF consumes less prediction time than other methods. Compared to GLFM-

based methods, GERF saves about 72 percent prediction time. This indicates that GERF provides more efficiency online recommendation for group event recommendation task. Moreover, we observe that GLFM-based methods and CBPF-based methods require similar prediction time, because they are latent factor models where user and event are represented by latent vectors and ranking score is inner product of vectors. HBGG consumes very large time to make prediction, because there are two latent topics are coupled together.

7 Conclusion

In this paper, we propose a novel group event recommendation framework GERF based on learning-to-rank technique to incorporate different contextual influences and members' preferences in a group. We formalize group event recommendation task as a learning-to-rank problem, and then analyze different contextual influences on user behavior of attending events. Based on our analysis, six contextual features are extracted for capturing individual preference considering different contextual influences, respectively. We propose the method to create training data for learning-to-rank. Different with the existing learning-to-rank algorithms, a fast learning-to-rank algorithm, called Bayesian Group Ranking (BGR), is proposed to learn a group-dependent model for each group. We conduct an empirical study on two real-world datasets and the experimental results show that GERF achieves higher recommendation accuracy than state-of-the-art methods and BGR outperforms existing learning-to-rank methods for group event recommendation in terms of effectiveness and efficiency. Our future work aims to jointly model contextual features extraction and BGR.

Acknowledgments

This work was supported by the Mutual Project of Beijing Municipal Education Commission of China.

References

[1] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *Proc. 25th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2002, pp. 253–260.

[2] W. Zhang and J. Wang, "A collective bayesian poisson factorization model for cold-start local event recommendation," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 1455–1464.

[3] Y. Jhamb and Y. Fang, "A dual-perspective latent factor model for group-aware social event recommendation," *Inf. Process. Manag.*, vol. 53, no. 3, pp. 559–576, 2017.

[4] S. Amer-Yahia, S. B. Roy, A. Chawlat, G. Das, and C. Yu, "Group recommendation: Semantics and efficiency," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 754–765, 2009.

[5] Q. Yuan, G. Cong, and C.-Y. Lin, "COM: A generative model for group recommendation," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 163–172.

[6] J. Wang, Y. Jiang, J. Sun, Y. Liu, and X. Liu, "Group recommendation based on a bidirectional tensor factorization model," *World Wide Web*, vol. 21, no. 4, pp. 961–984, Jul. 2018.

[7] M. S. Pera and Y.-K. Ng, "A group recommender for movies based on content similarity and popularity," *Inf. Process. Manag.*, vol. 49, no. 3, pp. 673–687, 2013.

[8] Z. Yu, X. Zhou, Y. Hao, and J. Gu, "TV program recommendation for multiple viewers based on user profile merging," *User Modeling User-Adapted Interaction*, vol. 16, no. 1, pp. 63–82, 2006.

[9] O. Kaššák, M. Kompan, and M. Bieliková, "Personalized hybrid recommendation for group of users: Top-N multimedia recommender," *Inf. Process. Manag.*, vol. 52, no. 3, pp. 459–477, 2016.

[10] I. Christensen, S. Schiaffino, and M. Armentano, "Social group recommendation in the tourism domain," *J. Intell. Inf. Syst.*, vol. 47, no. 2, pp. 209–231, 2016.

[11] J. Masthoff, "Group modeling: Selecting a sequence of television items to suit a group of viewers," in *Personalized Digital Television*. Berlin, Germany: Springer, 2004, pp. 93–141.

[12] T. De Pessemier, S. Dooms, and L. Martens, "Comparison of group recommendation algorithms," *Multimedia Tools Appl.*, vol. 72, no. 3, pp. 2497–2541, 2014.

[13] I. Garcia, S. Pajares, L. Sebastia, and E. Onaindia, "Preference elicitation techniques for group recommender systems," *Inf. Sci.*, vol. 189, pp. 155–175, 2012.

[14] A. Van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," in *Proc. Advances Neural Inf. Process. Syst.*, 2013, pp. 2643–2651.

[15] Z. Lu, H. Li, N. Mamoulis, and D. W. Cheung, "Hbgg: a hierarchical bayesian geographical model for group recommendation," in *Proc. SIAM Int. Conf. Data Mining*, 2017, pp. 372–380.

[16] T. Joachims, "Optimizing search engines using clickthrough data," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2002, pp. 133–142.

[17] Y. Cai, H.-f. Leung, Q. Li, H. Min, J. Tang, and J. Li, "Typicality-based collaborative filtering recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 3, pp. 766–779, Mar. 2014.

[18] M. J. Pazzani and D. Billsus, "Content-based recommendation systems," in *The Adaptive Web*. Berlin, Germany: Springer, 2007, pp. 325–341.

[19] W. Chen, Z. Niu, X. Zhao, and Y. Li, "A hybrid recommendation algorithm adapted in e-learning environments," *World Wide Web*, vol. 17, no. 2, pp. 271–284, 2014.

[20] X. Li, M. Jiang, H. Hong, and L. Liao, "A time-aware personalized point-of-interest recommendation via high-order tensor factorization," *ACM Trans. Inf. Syst.*, vol. 35, no. 4, p. 31, 2017.

[21] H. Wang, Y. Fu, Q. Wang, H. Yin, C. Du, and H. Xiong, "A location-sentiment-aware recommender system for both home-town and out-of-town users," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 1135–1143.

[22] A. Q. Macedo, L. B. Marinho, and R. L. Santos, "Context-aware event recommendation in event-based social networks," in *Proc. 9th ACM Conf. Recommender Syst.*, 2015, pp. 123–130.

[23] W. Yao, J. He, G. Huang, J. Cao, and Y. Zhang, "A graph-based model for context-aware recommendation using implicit feedback data," *World Wide Web*, vol. 18, no. 5, pp. 1351–1371, 2015.

[24] V. R. Kagita, A. K. Pujari, and V. Padmanabhan, "Virtual user approach for group recommender systems using precedence relations," *Inf. Sci.*, vol. 294, pp. 15–30, 2015.

[25] M. Gartrell, X. Xing, Q. Lv, A. Beach, R. Han, S. Mishra, and K. Seada, "Enhancing group recommendation by incorporating social relationship interactions," in *Proc. 16th ACM Int. Conf. Supporting Group Work*, 2010, pp. 97–106.

[26] S. Berkovsky and J. Freyne, "Group-based recipe recommendations: analysis of data aggregation strategies," in *Proc. 4th ACM Conf. Recommender Syst.*, 2010, pp. 111–118.

[27] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li, "Learning to rank: From pairwise approach to listwise approach," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 129–136.

[28] T. Watanabe, "Optimized online rank learning for machine translation," in *Proc. Conf. North American Chapter Association Comput. Linguistics: Human Language Technol.*, 2012, pp. 253–262.

[29] B. Shi, G. Poghosyan, G. Ifrim, and N. Hurley, "Hashtagger+: Efficient high-coverage social tagging of streaming news," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 1, pp. 43–58, Jan. 2018.

[30] N. X. Bach, N. Do Hai, and T. M. Phuong, "Personalized recommendation of stories for commenting in forum-based social media," *Inf. Sci.*, vol. 352, pp. 48–60, 2016.

[31] T.-Y. Liu, et al., "Learning to rank for information retrieval," *Foundations Trends® Inf. Retrieval*, vol. 3, no. 3, pp. 225–331, 2009.

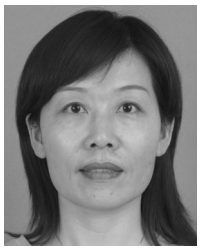
[32] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proc. 22nd Int. Conf. Mach. Learn.*, 2005, pp. 89–96.

[33] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer, "An efficient boosting algorithm for combining preferences," *J. Mach. Learn. Res.*, vol. 4, no. Nov., pp. 933–969, 2003.

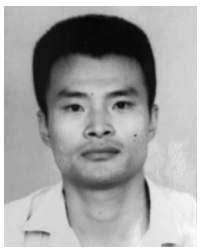
[34] D. Metzler and W. B. Croft, "Linear feature-based models for information retrieval," *Inf. Retrieval*, vol. 10, no. 3, pp. 257–274, 2007.
[35] Q. Wu, C. J. Burges, K. M. Svore, and J. Gao, "Adapting boosting for information retrieval measures," *Inf. Retrieval*, vol. 13, no. 3, pp. 254–270, 2010.
[36] D. Billsus and M. J. Pazzani, "A personal news agent that talks, learns and explains," in *Proc. 3rd Annu. Conf. Auton. Agents*, 1999, pp. 268–275.
[37] W. R. Tobler, "A computer movie simulating urban growth in the detroit region," *Econ. Geography*, vol. 46, no. sup1, pp. 234–240, 1970.
[38] J.-D. Zhang and C.-Y. Chow, "Core: Exploiting the personalized influence of two-dimensional geographic coordinates for location recommendations," *Inf. Sci.*, vol. 293, pp. 163–181, 2015.
[39] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Boca Raton, FL, USA: CRC Press, 1986, vol. 26.
[40] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Processing Manage.*, vol. 24, no. 5, pp. 513–523, 1988.
[41] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, no. Jan., pp. 993–1022, 2003.
[42] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," CoRR, vol. abs/1301.3781, 2013.
[43] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. 25th Conf. Uncertainty Artif. Intell.*, 2009, pp. 452–461.
[44] P. G. Campos, F. Díez, and I. Cantador, "Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols," *User Model. User-Adapted Interaction*, vol. 24, no. 1–2, pp. 67–119, 2014.
[45] L. Baltrunas, T. Makcinskas, and F. Ricci, "Group recommendations with rank aggregation and collaborative filtering," in *Proc. 4th ACM Conf. Recommender Syst.*, 2010, pp. 119–126.
[46] T. Joachims, "Training linear svms in linear time," in *Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2006, pp. 217–226.
[47] J. Xu and H. Li, "Adarank: A boosting algorithm for information retrieval," in *Proc. 30th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2007, pp. 391–398.

Yulu Du is currently working toward the PhD degree in the School of Computer Science, Beijing University of Posts and Telecommunications. His research interests include recommender systems and intelligent information processing.

Xiangwu Meng received the PhD degree from the Institute of Software Chinese Academy of Sciences, in 1997. He is a full professor of the Beijing University of Posts and Telecommunications. His research interests include recommender systems, network services, and artificial intelligence.



Yujie Zhang is an associate professor of the Beijing University of Posts and Telecommunications. Her research interests include recommender systems, network services, and intelligent information processing.



Pengtao Lv received the master's degree in 2015. He is currently working toward the PhD degree in the School of Computer Science, Beijing University of Posts and Telecommunications. His research interests include data mining, user behavior analytics, recommender systems, and intelligent information processing.