Hindawi
Scientific Programming
Volume 2021, Article ID 1969118, 18 pages
https://doi.org/10.1155/2021/1969118

Research Article

Dynamic Group Recommendation Algorithm Based on Member Activity Level

Junjie Jia, Yewang Yao, Zhipeng Lei, and Pengtao Liu

College of Computer Science and Engineering, Northwest Normal University, Lanzhou 730071, Gansu, China

Correspondence should be addressed to Yewang Yao; 2019221823@nwnu.edu.cn

Received 9 April 2021; Revised 12 July 2021; Accepted 9 August 2021; Published 29 August 2021

Academic Editor: Sebastiano Fabio Schifano

The rapid development of social networks has led to an increased desire for group entertainment consumption, making the study of group recommender systems a hotspot. Existing group recommender systems focus too much on member preferences and ignore the impact of member activity level on recommendation results. To this end, a dynamic group recommendation algorithm based on the activity level of members is proposed. Firstly, the algorithm predicts the unknown preferences of members using a time-series-oriented rating prediction model. Secondly, considering the dynamic change of member activity level, the group profile is generated by designing a sliding time window to investigate the recent activity level of each member in the group at the recommended moment, and preference is aggregated based on the recent activity level of members. Finally, the group recommendations are generated based on the group profile. The experimental results show that the algorithm in this paper achieves a better recommendation result.

1. Introduction

Recommender systems solve the problem of information overload and help users to choose from the options in our day to day life [1]. The traditional recommender systems use mathematical tools to analyze users' historical behavior records, and then draw their preferences to recommend products and services to them. Today, the traditional recommender systems have been widely used in many fields such as e-commerce [2], social networks [3, 4], and online entertainment [5, 6]. However, traditional recommender systems only focus on single-user recommendation scenarios. In some practical scenarios, we may need to generate recommendations to a group of multiple users, such as recommending TV shows to a family group and gathering places and dishes to a colleague group. Based on these practical scenarios, group recommender systems have become a new research hotspot.

At present, researches on group recommender systems mainly focus on preference aggregation strategies, i.e., how to aggregate preferences of members to alleviate the preference conflict problem in the group as much as possible, so that the recommendations can meet the needs of all/most members. To solve this problem, various aggregation models have been proposed by scholars, such as the average strategy [7], the least misery strategy [8], the most pleasure strategy [9], the most respected strategy [10], etc. Although these studies have promoted the process of preference aggregation to different degrees from different perspectives, unfortunately, none of them has considered the issue of the degree of interaction between group members and the recommender system. In the actual recommendation scenario, there are usually significant differences in the degree of interaction between group members and the recommender system, i.e., there is a difference in the activity level of the members. In this paper, we constructed a model for calculating the activity level of members by the degree of their interaction with the recommender system, which will be described in detail in Section 4.2.2.

Meanwhile, some progress has been made in the study of group behavior in social psychology. Among them, Back [11] argues that the formation of group behavior depends on the mutual stimulation of participants, while special members such as leaders or active members of the group play a leading

role in the formation of group behavior. Also, the Symbolic Interaction Theory of Blumer [12] argues that member behaviors are constantly reflected and contagious in the group, and will eventually achieve consistency in group behavior. This coincides with the claims made by Le Bon and Allport, where Le Bon [13] argues that individuals in a group will tend to be consistent with others when making choices and judgments, and Allport [14] argues that there is a greater similarity in the thoughts of group members after a long period of interaction and bonding. Therefore, by transferring the research results of social psychology on group behavior to group recommender systems, it is natural to conclude that the thoughts and behaviors of active members in a group will spread continuously in group decision-making, and make the decision result consistent by influencing the rest of the group members, which in turn tends to favor the active members. Therefore, in the preference aggregation process of the group recommender system, we need to pay more attention to the preferences of the active members in the group, while making appropriate trade-offs for the preferences of the inactive ones.

In addition to the above theoretical aspects, from the perspective of practical application, there are two other reasons for appropriately discarding the preferences of inactive members. First, there are already fierce preference conflicts in the group, so it is inevitable to ignore the personal preferences of other members while considering the preferences of inactive members too much, which will easily lead to unfairness. Second, inactive members may not care about recommendations because of low-level interaction with the recommender system, even if the final recommendations include the items loved by them. Therefore, it is helpful to improve the quality of group recommendations by detecting the activity level of members in the group, analyzing the recent activity level of each member before preference aggregation, and accordingly increasing the influence of active members in the preference aggregation process.

We proposed a Dynamic Group Recommendation Algorithm Based on Member Activity Level (DMA) to overcome this potential problem and improve the quality of recommendations. Since the existing group recommendation algorithms generate group profiles directly based on member profiles, these methods would make the recommender system focus too much on member profiles and ignore the bias effect of member activity level on recommendations. Therefore, we firstly proposed the concept of activity level and designed a sliding time window. After that, we analyzed the activity level of members in the sliding time window at the recommended time and generated a group profile based on "the member with higher activity level plays a more important role in group decision-making" by assigning different weights to different members of the group for preference aggregation. Finally, group recommendations are generated by the Top-N recommendation method on the basis of the group profile. Compared to other aggregation strategies, our algorithm reduced the interference of inactive members on the recommendation results based on the influence of active members in the group on the global group profile.

The main contributions of this paper are as follows:

(1) We developed a method to calculate member activity levels and incorporated the obtained member activity levels into the preference aggregation process to generate a more accurate group profile.

(2) We proposed a dynamic group recommendation algorithm based on member activity level, which generates dynamic recommendations for the group by analyzing the recent activity levels of each member in the group at the recommendation moment.

The remainder of the paper is organized as follows. Section 2 briefly presents some research literature related to group recommendation algorithm. Section 3 briefly introduces the relevant techniques used in this paper. Section 4 provides a detailed description of the DMA algorithm. Section 5 presents several experimental datasets, experimental results, and analyses. Finally, we gave few future directions and concluded in Section 6.

2. Related Work

In the study of group recommendation algorithms, most of the current work focuses on preference aggregation of group members, and the main focus of the research is on designing effective preference aggregation methods and preference aggregation strategies to alleviate the preference conflicts among members as much as possible, so that the needs of each member in the group can be considered.

In the process of preference aggregation, the aggregation methods can be divided into model aggregation and recommendation aggregation depending on when the aggregation occurs, as shown in Figures 1 and 2. Model aggregation [15] refers to aggregating the preferences of group members to generate group preferences, and then using personalized recommendation methods to obtain a group recommendation list based on group preferences; recommendation aggregation [16] refers to firstly using personalized recommendation methods to generate recommendations for each group member, and then aggregating the recommendations of each member to generate a group recommendation list. The recommendation aggregation is further divided into rating aggregation [17] and ranking aggregation [18].

Both aggregation methods have their advantages in terms of recommendation effectiveness, but the recommendation aggregation method has higher flexibility compared to model aggregation and is more conducive to recommendation efficiency and interpretation of recommendation results [19]. Besides, experiments by De Pessemier et al. [20] showed that the accuracy of recommendation aggregation is higher when using singular value decomposition-based algorithms. Since the group recommendation algorithm proposed in this paper is based on the singular value decomposition method, the recommendation aggregation method is used in this paper for group recommendation.

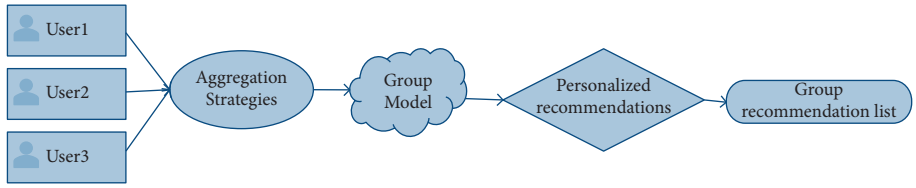

Figure 1: Model aggregation.

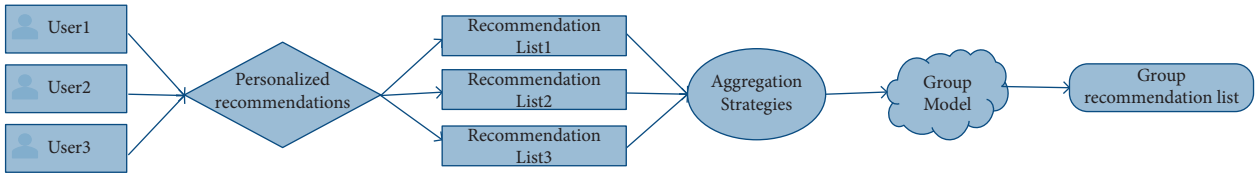

Figure 2: Recommendation aggregation.

Although there is some influence of the preferred aggregation method on the recommendation results of the group, it can be seen through Figures 1 and 2 that the choice and design of the aggregation strategy usually affects the satisfaction of the group members with the recommendation results to a greater extent [21, 22]. There are many kinds of research on aggregation strategies, and in addition to the basic aggregation strategies mentioned above [7–10], there exist various combined models and weighted models of aggregation strategies. Among the combined models of multiple aggregation strategies, Chen et al. [23] proposed a satisfaction balancing strategy, which improved the satisfaction of group members by weighting the combination of the average strategy and the least misery strategy. Jameson [24] combined the average and the median strategies in the preference aggregation process to avoid the interference of malicious users on group preferences. Tao et al. [25] used the most respected strategy and the average strategy on both sides of the disagreement threshold to generate the corresponding recommendation lists under different group features, respectively. Such methods make up for the shortcomings of a single aggregation strategy by combining multiple strategies and improve the recommendation effect. However, this kind of method lacks consideration of features and influence of group members, so some scholars proposed weighted models. Ardissono et al. [26] performed preference aggregation by analyzing the demographic information of group members and assigning different weights to different categories of members. Yuan et al. [27] argued that the process of group decision-making is more influenced by the experts in the group and therefore expert users have a higher weight in group decision-making. Wang et al. [28] assign weights to each group member based on their contribution to generating the group profile in the preference aggregation process, which in turn generates a group recommendation list. Berkovsky and Freyne [29] built a group model by weighting each member based on the role model and rating participation of each user in the group, and experiments showed that the method can effectively improve the recommendation effect. However, its calculation of member participation over the whole time series not only increases the computational effort of the recommender system but also ignores the dynamic changes of member participation, which is biased for the instantaneous recommendation requirements.

Although the above group recommendation algorithms have been designed to solve the preference aggregation problem from different perspectives, few algorithms take into account the interest drift of group members. Meanwhile, time-series-oriented personalized recommendation algorithms have been widely studied in recent years due to their ability to capture the evolution of users' interests in the temporal dimension and to mine users' potential preferences, and generate recommendations from a dynamic perspective. For example, Ding and Li [30] simulated the natural forgetting pattern of the human brain by designing a temporal decay function and incorporated the function into a rating prediction formula to predict users' unknown preferences. Along with the upsurge of matrix factorization models in the field of recommender systems, numerous time-series-oriented recommendation studies have gradually used matrix factorization models to predict user preferences. Among them, Sun et al. [31] proposed a recommendation algorithm based on the user's temporal behavior, which searches the nearest neighbor relationship between users and items based on the temporal information of user ratings and adds them to the probability matrix factorization to obtain the recommendation results. Zhang et al. [32] constructed a user-item-time model by integrating temporal weights into an intensive predicted rating matrix obtained from probability matrix factorization to generate recommendations. Although the above algorithms can effectively predict user ratings, they lack consideration of item factors in the factorization process. Therefore, Koren [33] relied on the powerful scalability of matrix factorization to model the temporal parameters jointly with the SVD++ model into the timeSVD++ model based on taking into account the long-term and short-term changes of user interest and item popularity, which can effectively capture the local changes of user interest and precisely predict user preferences, and this paper mainly refers to this part of the work in predicting the unknown preferences of members.

In summary, we take into account the drift of group members' interests and dynamic changes of their activity level, combine the existing time-series-oriented recommendation algorithms, and choose a more effective recommendation aggregation method for group recommendation, expecting to generate a more precise group profile through more detailed mining of group members' preferences and activity level, to achieve a higher quality recommendation effect.

3. Relevant Technology

This section firstly provides an overview of the technologies used in the paper, then gives a formal description of the group recommender system, and finally defines the problem of this paper.

3.1. Nonnegative Matrix Factorization. Nonnegative Matrix Factorization (NMF) was proposed by Lee and Seung [34] in Nature, which decomposes the original matrix into two smaller matrices and assumes that the smaller matrices satisfy nonnegative constraints, making the decomposition results more interpretable, e.g., pixel values cannot be negative, ratings of items are also generally positive, etc. In terms of the user-item rating matrix, NMF can be expressed as follows:

$$\begin{cases} R^{m\times n} \approx U^{m\times k}V^{k\times n}, \\ \forall U_{ik} \in U^{m\times k}, \quad \text{s.t.}\, U_{ik} \geq 0, \\ \forall V_{kj} \in V^{k\times n}, \quad \text{s.t.}\, V_{kj} \geq 0, \end{cases} \tag{1}$$

where $R^{m\times n}$ denotes the original user-item rating matrix; $U^{m\times k}$ and $V^{k\times n}$ denote the decomposed user matrix and item matrix, respectively; U_{ik} and V_{kj} denote the elements in matrices $U^{m\times k}$ and $V^{k\times n}$; m and n are the number of users and items; and k is the number of latent factors.

To maximize the approximation of the product of $U^{m\times k}$ and $V^{k\times n}$ to the original rating matrix $R^{m\times n}$, the following objective function is established and solved according to the multiplicative iteration rule proposed by Lee and Seung [35].

$$\min_{U,V\geq 0} \Phi(U,V) = \frac{1}{2}\sum_{i,j}\left[R_{ij} - (UV)_{ij}\right]^2 + \gamma\left(\|U\|_2^2 + \|V\|_2^2\right), \tag{2}$$

where R_{ij} denotes the elemental values in the original rating matrix and γ is the regularization coefficient.

3.2. K-Means. K-means clustering is an unsupervised machine-learning algorithm that maps different data points into different clusters by iteratively solving for a given data, where the data points in the same cluster have similar features and the features of the data points between different clusters are different.

The specific steps of K-means clustering are as follows: (1) randomly select k points as the initial clustering centers; (2) calculate the distance between each data point to the k clustering centers and assign it to the center with the closest distance; (3) update the clustering centers and calculate the mean point of each cluster as the new clustering center; (4) judge whether the termination condition is reached: if the condition is reached, return the clustering result; otherwise, repeat Steps (2)–(4), where the termination condition can be that the clustering center no longer changes, the error sum of squares is locally minimized, or the number of iterations reaches an agreed threshold.

3.3. Group Recommender System. In this subsection, we provided a simple formal definition of a group recommender system in terms of its general steps.

Definition 1. A group G is a collection of several users with preferences, $G = \{u_i | 0 < i \leq n\}$, where u_i denotes the group members and n is the group size.

Definition 2. Group preferences $r_{Gi} = \sum_{u\in G}\omega_{ui}r_{ui}$, where r_{Gi} denotes the preference of group G on item i obtained by aggregating members ratings; r_{ui} denotes the rating of the member u on the item i; and ω_{ui} is the weight of member u in a group G when aggregating preferences on the item i. Different weights can represent different preference aggregation strategies. For example, when $\omega_{ui} \equiv 1/|G|$, the above equation can represent the average strategy; when the weight of a member is constant 1 and the weight of other members is constant 0, the above equation can represent the most respected person strategy.

Definition 3. Top-N group recommendation: The group recommender system generally adds the items with the top n highest group ratings from the candidate item set to the group recommendation list, and the candidate item set is the set of items that has not been rated by any member of the group. For a given group G, a group recommendation list I_G can be generated from the candidate item set I as follows:

$$\begin{cases} |I_G| = N, \\ \forall i \in I \\ \forall j \in I \\ \text{s.t.}\, r_{Gi} \geq r_{Gj}, \quad i \in I_G, j \notin I_G, \end{cases} \tag{3}$$

and the items in $|I_G|$ are listed in descending order by group rating.

3.4. Problem Definition. Given ratings $r_{ui}(t)$ ($r_{ui}(t) \in [1,5]$) for item i by user u at moment t, a higher $r_{ui}(t)$ indicates that user u is more interested in item i. $r_{ui}(t) =$ null indicates no rating. Then, we stored $r_{ui}(t)$ as a triple (u,i,t) in the set $K = \{(u,i,t) | (r_{ui}(t) \in [1,5])$ and (if $\exists r_{ui}(t') \in [1,5]$ and $t' > t$ then $(u,i,t) = (u,i,t'))\}$. The (u,i,t) in K stores only the ratings of user u for the most recent moment for item i.

In this paper, our task is to detect potential group G based on known ratings and obtain the predicted rating $\hat{r}_{vj}(t)$ of group member v for candidate item j. After that, we will generate a group profile at the recommended time t_{rec} by

obtaining the group rating $r_{Gi}(t_{\text{rec}})$ via aggregating the predicted ratings of members, and then the top-N group recommendations will be generated based on the group profile.

4. Proposed Model

For the problem of preferences conflict in group recommender systems, we proposed the concept of activity level and quantified the importance of different members in group decision-making with it. Based on this, we weighted the preferences of each member according to their activity level and aggregated them to generate group recommendations by simulating the group decision-making process in real scenarios. The general framework of our algorithm is shown in Figure 3.

According to Figure 3, it can be seen that the algorithm we proposed mainly consists of the following modules:

(1) *Group Mining*. The group mining module is implemented based on NMF dimension reduction and K-means clustering, and is mainly used to detect groups that are more similar internally, which is the potential basis of our research.

(2) *Group Modeling*. The group modeling module firstly uses the timeSVD++ model to fill the rating matrix to get the filled matrix, then mines the member preferences from the filled matrix, and finally uses the preference aggregation strategy proposed in this paper to aggregate the member preferences to generate the group profile, which is the core of our research.

(3) *Group Recommendation*. The group recommendation module generates recommendations for the group based on the already generated group profile using the Top-N recommendation method, which is the main purpose of our research.

The above is the main content of this paper. In the following parts, this section will be carried out in accordance with the above modules.

4.1. Group Mining. The literature [36] proposed a group mining algorithm, which detected potential groups by K-means clustering of the user rating vectors after filling the blank ratings with predicted ratings. However, the user rating vector is generally of high dimensionality, and it may affect the clustering process with the curse of dimensionality when directly input into the K-means algorithm. Therefore, in this paper, we proposed a new group mining algorithm based on the literature [36]. The algorithm firstly decomposed the user-item rating matrix with NMF, and then the decomposed user matrix was input into the K-means clustering algorithm to mine potential groups. Since NMF can achieve a low-dimensional representation of high-dimensional data while maintaining the essence of the original data, it has been widely applied in the fields of image analysis, text clustering, and data mining [37]. The user feature vectors in the user matrix decomposed by NMF

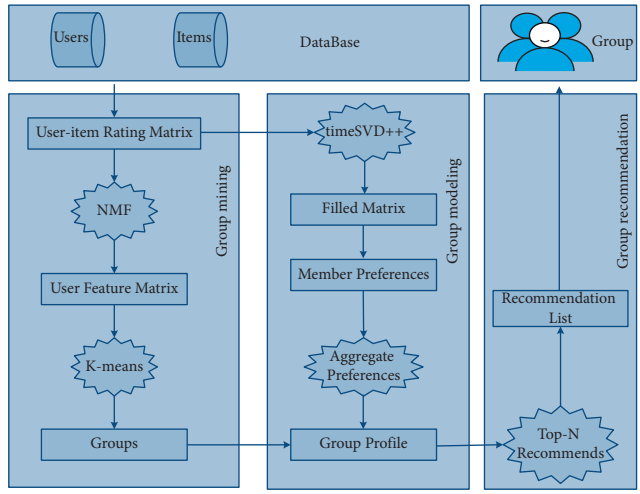

Figure 3: Dynamic group recommendation algorithm framework.

represent the preferences of users for different implicit item factors. Therefore, it can be input into K-means to cluster users with similar interests into the same group and users with different interests into different groups while alleviating the effect of the curse of dimensionality. In this paper, the cosine distance is used as a distance measure in the clustering process, and the formula for calculating the cosine distance is as follows:

$$\begin{aligned} \text{dist}(\overrightarrow{A}, \overrightarrow{B}) &= 1 - \cos\theta \\ &= 1 - \frac{\overrightarrow{A} \cdot \overrightarrow{B}}{|\overrightarrow{A}| \cdot |\overrightarrow{B}|}, \end{aligned} \tag{4}$$

where $\overrightarrow{A}$ and $\overrightarrow{B}$ denote two vectors, θ denotes the angle between the vectors, and $|\overrightarrow{A}|$ denotes the norm of the vector $\overrightarrow{A}$.

The group mining algorithm is described in detail as Algorithm 1 [36].

4.2. Group Modeling. The purpose of group modeling is to calculate a group rating for each candidate item and then decide which items should be recommended to the group. To generate a group profile, the profiles of the members in the group need to be aggregated, and since the member profiles are often very sparse, the unknown preferences in the member profiles need to be predicted before aggregation to get the corresponding predicted ratings. Therefore, in this subsection, we first outline how to use the timeSVD++ model to predict member unknown preferences, which is the basis for generating a group profile. Then, we will detail how to integrate the activity level into the preference aggregation process to generate a group profile, which is the core of this section and also the main innovation of this paper.

4.2.1. Unknown Preferences Prediction. In group recommender systems, predicting the unknown preferences of members is the basis for generating group profiles, so the unknown ratings of group members need to be predicted

Input: Rating matrix $R^{m\times n}$, Latent factor number k, Group number g.
Output: g groups.
Step 1: $U^{m\times k}\leftarrow R^{m\times n}\leftarrow$NMF;
Step 2: $U^{m\times k}\leftarrow$K-means;
Step 3: Return g groups.

Algorithm 1: Group mining algorithm.

before building the group model. Since user preferences drift dynamically over time and the popularity of items changes over time, better prediction accuracy will be achieved by considering the time factor when predicting unknown ratings of members, which will facilitate the building of a more precise group model. In this part of the study, we mainly referred to the work of the literature [33].

The literature [33] proposed the timeSVD++ model based on the consideration that preferences of users for items have a time effect, which transforms the variables into a function about time t. While alleviating data sparsity, the interest drift of users can be accurately captured, thereby achieving a better prediction accuracy. The main functions in the paper are as follows:

(1) Two main temporal effects are included in the user preference baseline predictor: one is the change in user bias over time; the other is the effect of item bias over time.

$$b_{ui}(t) = \mu + b_u(t) + b_i(t), \tag{5}$$

where $b_{ui}(t)$ denotes the baseline estimate of the user u for the item i at a time t; μ is the overall average rating; and $b_u(t)$, $b_i(t)$ are time-varying, real-valued functions that represent user bias and item bias subject to time effects, respectively.

(2) *Item Bias*. By splitting the timeline into many bins, the item bias is split into a stationary part and a time-changing part.

$$b_i(t) = b_i + b_{i,\mathrm{Bin}(t)}, \tag{6}$$

where b_i denotes the stationary part and $b_{i,\mathrm{Bin}(t)}$ denotes the time-changing part of the item bias.

(3) *User Bias*. Considering the long-term and short-term changes of user bias affected by time, i.e., user bias may change gradually over time or suddenly on a certain day, $b_u(t)$ is divided into two parts: static and dynamic, and the dynamic includes long-term changes (gradual drift) and short-term changes (sudden drift).

$$b_u(t) = b_u + \alpha_u \cdot \mathrm{dev}_u(t) + b_{u,t}, \tag{7}$$

where b_u denotes the static part of the user bias; $\alpha_u \cdot \mathrm{dev}_u(t) + b_{u,t}$ denotes the dynamic part, $b_{u,t}$ denotes the sudden drift in the dynamic change, $\alpha_u \cdot \mathrm{dev}_u(t)$ denotes the gradual drift in the dynamic change, α_u denotes the parameter assigned to the user u, and $\mathrm{dev}_u(t)$ is a time drift function as follows:

$$\mathrm{dev}_u(t) = \mathrm{sign}(t - t_u) \cdot |t - t_u|^{\beta}, \tag{8}$$

where $|t - t_u|$ measures the time distance between t and t_u; β is a parameter.

The prediction of unknown user ratings is completed by solving the following objective function to obtain the predicted ratings.

$$\min \sum_{(u,i,t)\in K} \left(r_{ui}(t) - \mu - b_u - \alpha_u \cdot \mathrm{dev}_u(t) - b_{u,t} - b_i - b_{i,\mathrm{Bin}(t)}\right)^2 + \lambda\left(b_u^2 + \alpha_u^2 + b_{u,t}^2 + b_i^2 + b_{i,\mathrm{Bin}(t)}^2\right), \tag{9}$$

where λ is the regularization coefficient.

A filled matrix $RF^{m\times n}$ is generated by filling in the blank in the original rating matrix $R^{m\times n}$. The filled matrix contains the preference information of all users. We exactly mined the member preferences (both known and predicted preferences) using the rating information of all users. Compared to using only the ratings of group members to predict unknown preferences, this has the advantage of generating more accurate predicted ratings by modeling the entire data without having to drop most of the data. Eventually, the member preferences $rf_u(t) = \{rf_{ui}(t) | i \in I\}$ are obtained by extracting the rows in the filled matrix to which the member belongs.

4.2.2. Preferences Aggregation. Member preference aggregation is a key problem in constructing a group model, and we structured the group model by integrating activity level into the preference aggregation process. Meanwhile, we treated the activity level as a counting problem within a time interval, and the time information in the member preferences is usually in the form of timestamps, so a time window is designed to process them. In the later section, this section first defines the member activity level within a time window and then describes how to structure the group model by preference aggregation based on the member activity level.

(1) Member Activity Level. In recommender systems, activity level refers to the degree of user interaction with the recommender system, which is most intuitively reflected by the number of historical ratings of users. However, for group recommender systems, it is not appropriate to analyze the activity level of group members over the entire time series when performing activity-level investigations. Because the activity level of members is dynamic, members who were active a long time ago may not be active now, and similarly, members who were inactive a long time ago may become active now. That is, the degree of interaction of group members with the recommender system becomes less and less reliable for calculating the current activity level as the time span increases. Moreover, assuming that recommendations are generated to the group at the time t_{rec}, it is also usually difficult to predict member activity level after time t_{rec} due to various possible abrupt conditions. Therefore, our investigation of the member activity level is set to a recent period at the moment of recommendation, based on which a sliding time window is designed, as shown in Figure 4.

In Figure 4, firstly, the maximum and minimum timestamps are selected from the rating data, which are denoted as $t_{\max}$ and $t_{\min}$, respectively. $t_{\max}$ indicates the time closest to the current time, while $t_{\min}$ indicates the time farthest from the current time. Then, the time interval $|t_{\max} - t_{\min}|$ is split into many small periods, each representing one day (i.e., 86400 seconds), called short windows, denoted by Δt. η short windows form a long window, denoted T (i.e., $T = \eta\Delta t$), and the long window is set to slide forward with time. Finally, suppose that items are recommended to the group at the moment t_{rec}. In order to analyze the activity level of each member in the time period closer to t_{rec}, we would simply slide the long window to the moment t_{rec} and then investigate the activity level of the group members within that window. By the way, the size of the long window depends on specific industry patterns.

Now, for the difference in the number of user ratings, the activity level of each member under the current moment is calculated by analyzing the number of ratings of each member and the total number of ratings of the group within the sliding time window at the recommended moment. For generating recommendations to the group at the time t_{rec}, the activity level of each member is defined as follows:

$$A_{uG}(t_{\text{rec}}) = \frac{|R_u(t)|}{\sum_{u \in G} |R_u(t)|}, \quad t \in [t_{\text{rec}} - T, t_{\text{rec}}], \tag{10}$$

where $A_{uG}(t_{\text{rec}})$ denotes the activity level of the member u in the group G at the recommended moment t_{rec}, $|R_u(t)|$ denotes the number of historical ratings of the member u in the period t, and T is the sliding time window length.

Since the value of t is in a time range, the specific representation of $|R_u(t)|$ is as follows:

$$|R_u(t)| = \sum_{t=t_{\text{rec}}-T}^{t_{\text{rec}}} f(|r_{ui}(t)|),$$
$$f(|r_{ui}(t)|) = \begin{cases} 1, & |r_{ui}(t)| > 0 \\ 0, & \text{else} \end{cases}, \quad t \in [t_{\text{rec}} - T, t_{\text{rec}}], \tag{11}$$

where $f(x)$ is an indicator function indicating whether the user u has a rating record in the time range $t \in [t_{\text{rec}} - T, t_{\text{rec}}]$.

To verify whether there is a difference between the number of user ratings, the number of ratings for all users in the MovieLens dataset and for members of several groups mined by the method in Section 4.1 was analyzed, with higher numbers of ratings indicating more active users. At the same time, the recommended moment t_{rec} is randomly selected to count the number of users corresponding to each number of ratings in the random period. The smaller the number of users under the same number of ratings, the greater the difference between the activity level of group members in that period; to verify the differences between the activity level of ratings of group members in the recent past at the moment of recommendation, the sliding time window length is set to 90 days, i.e., 7776000 seconds. The results are shown in Figures 5 and 6.

Figure 5 shows the statistics of the number of user ratings in MovieLens. It can be seen that the number of ratings varies significantly among users, with some users having only a few rating records and some users having more than 700 rating records. Among all users, the number of ratings of users follows long-tailed distribution, indicating that a few users have a high number of ratings, while most users have a low number of ratings. In different groups, it can be seen that the number of users corresponding to different numbers of ratings is generally lower, which indicates that the number of ratings varies among members in the group, i.e., there is a difference between the activity level of different members. Figure 6 presents the statistics of the number of user ratings over 90 days, and it can be seen that the number of ratings for most group members is completely different over time, except for a small number of very inactive users who have a small number of rating records. This shows that there is a significant difference between the activity level of each member in the group within a period. Therefore, it is necessary to analyze the activity level of each member before preference aggregation. Furthermore, the vertical combination of Figures 5 and 6 shows that compared to Figure 5, where some members have the same number of ratings, Figure 6 completely shows that the number of ratings varies among group members, so it is realistic and logical to investigate the recent activity level of members.

(2) Preferences Aggregation. The activity-level formula indicates that the higher the number of ratings a member has within a long window, the higher is his or her activity level in the group, and the more important is he or she in the

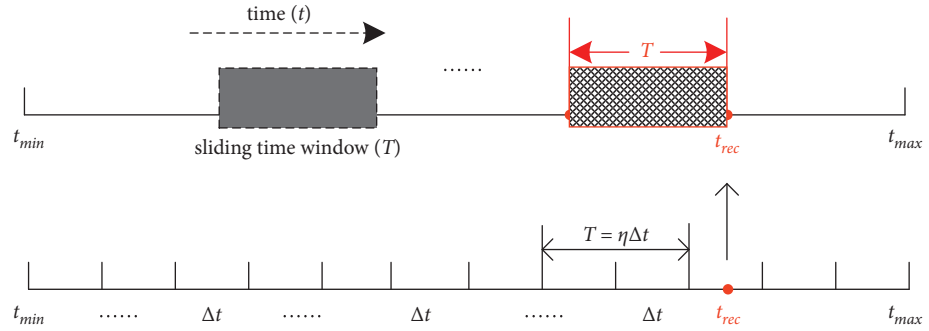

Figure 4: Sliding time window.

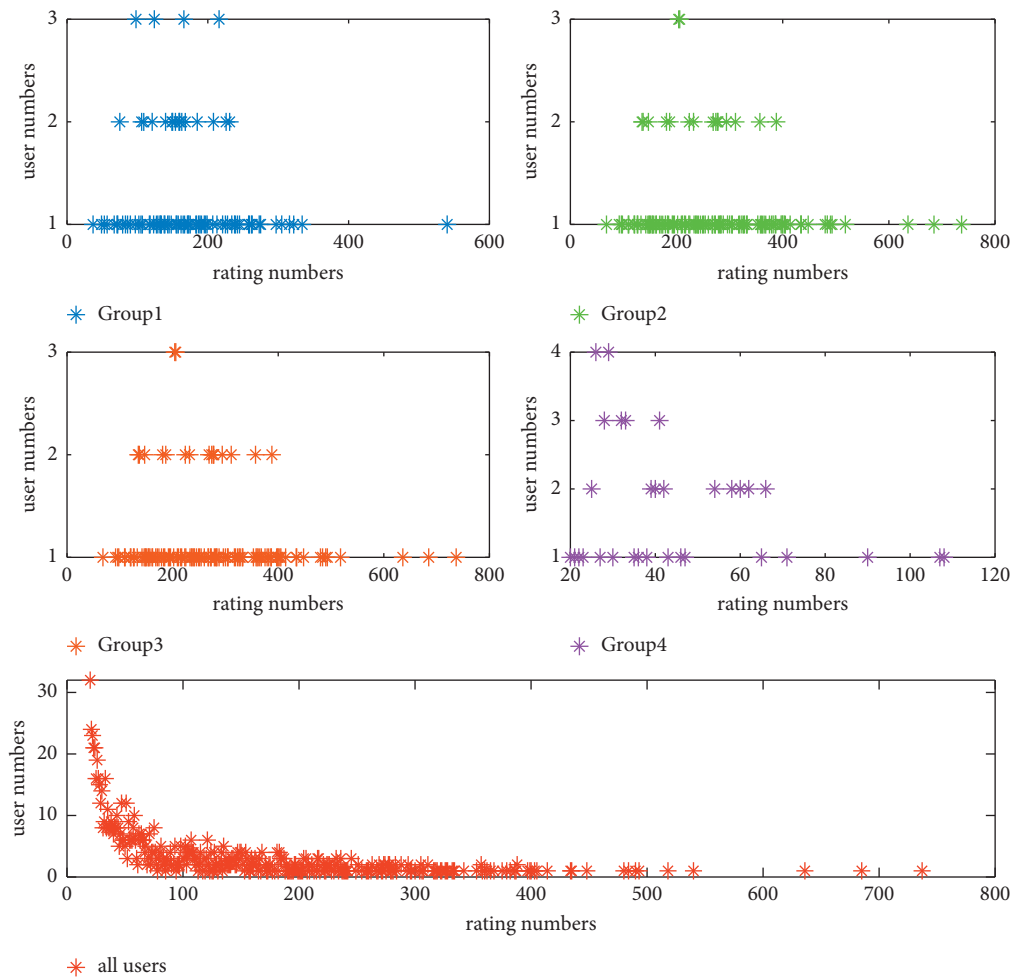

Figure 5: Statistics of the number of users under each rating number in MovieLens.

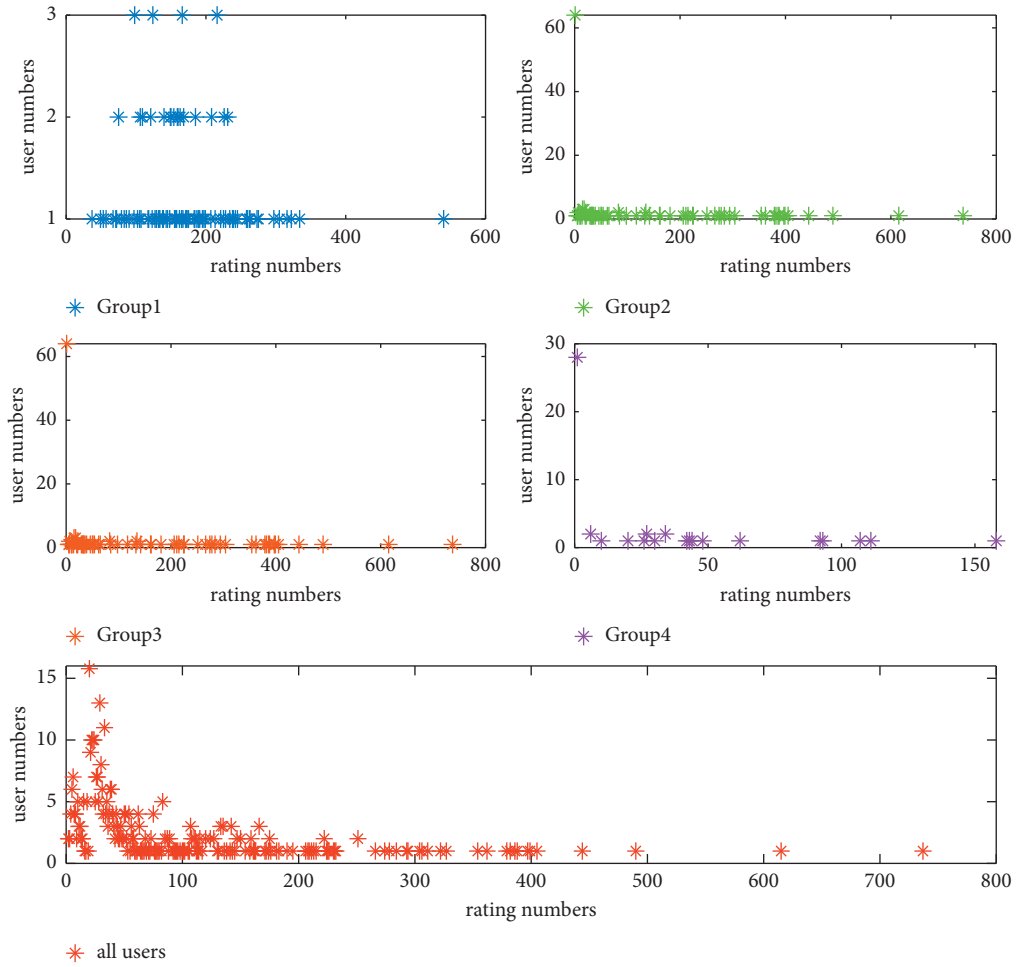

Figure 6: Statistics of the number of users corresponding to the number of ratings within 90 days in MovieLens.

preference aggregation process. Therefore, after the activity level of each member in the group is calculated, the activity level of each member can be regarded as his or her weight for preference aggregation to generate group preferences, as follows:

$$r_{Gi}(t_{\text{rec}}) = \sum_{u \in G} A_{uG}(t_{\text{rec}}) \cdot rf_{ui}(t), \tag{12}$$

where $r_{Gi}(t_{\text{rec}})$ denotes the rating for the item i by the group G at the recommended moment t_{rec}, i.e., the group preference.

According to equation (12), the modeling of the group G is accomplished in the process of calculating the ratings of the group for all candidate items.

4.3. Group Recommendation. The final implementation of the group recommender system is to generate a group-oriented recommendation list. After calculating the ratings of the group for all candidate items through Section 4.2, the Top-N recommendation algorithm can be used to select the N candidates with the highest group ratings to be added to the group's recommendation list. The dynamic group recommendation algorithm based on member activity level (DMA) is described as Algorithm 2 [28].

5. Experiment and Analysis

This section first introduces the datasets and evaluation metrics we used in the experiment, then introduces the baseline algorithms and experimental setup, and finally analyses the results as well as the parameters.

5.1. Datasets. In the experimental part, we choose the MovieLens dataset (https://grouplens.org/datasets/movielens) and the Netflix dataset (https://www.kaggle.com/netflix-inc/netflix-prize-data) as the experimental datasets. The MovieLens dataset and the Netflix dataset contain information about ratings given by users to movies, the time of ratings, and the tags of users and movies, which are common datasets for relevant scientific experiments in recommendation system studies, and their specific information is shown in Table 1. In the experiment, both datasets are randomly divided into a training set and a test set in the ratio of 8 : 2, and the algorithm model is first trained on the training set, and then the trained model is verified and analyzed on the test set.

Input: Group G, Rating matrix $R^{m\times n}$, Number of latent factors k, Candidate item set I, Recommended time t_{rec}, Sliding time window length η, Number of recommended items N.
Output: Group recommendation list I_G.
Step 1: $RF^{m\times n} \leftarrow R^{m\times n} \leftarrow$ timeSVD++;
Step 2: $RF_G^{|G|\times n} \leftarrow RF^{m\times n}$;
Step 3: For u in G:
Step 4: $A_{uG}(t_{\text{rec}}) = (|R_u(t)|/\sum_{u\in G}|R_u(t)|), \quad t \in [t_{\text{rec}} - T, t_{\text{rec}}]$;
Step 5: End For;
Step 6: For i in I:
Step 7: $r_{Gi}(t_{\text{rec}}) = \sum_{u\in G} A_{uG}(t_{\text{rec}}) \cdot rf_{ui}(t)$;
Step 8: End For;
Step 9: sorted $r_{Gi}(t_{\text{rec}})$ by descending order;
Step 10: select N highest items;
Step 11: Return I_G.

Algorithm 2: DMA algorithm.

5.2. Evaluation Metrics. We use the normalized discounted cumulative gain (nDCG) [20] as an evaluation metric for the accuracy of the group recommendation algorithm, which fully takes into account the influence of item ranking on the recommendation results. The formula is as follows:

$$\begin{aligned} \text{DCG@N} &= \text{rel}_1 + \sum_{i=2}^{N} \frac{\text{rel}_i}{\log_2(i)}, \\ n\text{DCG@N} &= \frac{\text{DCG@N}}{\text{IDCG@N}}, \end{aligned} \tag{13}$$

where DCG@N denotes the satisfaction of a user with the true ratings of the N items in the recommendation list relative to their positions in the list, and rel_i denotes the rating of the user for the item ranked i in the recommendation list. IDCG@N is the value of DCG@N in the best scenario, i.e., the recommendation list is ranked in the descending order of ratings for the individual user.

nDCG@N is the ratio of DCG@N and IDCG@N, which is between 0 and 1. And, the higher the nDCG@N, the higher the user satisfaction with the recommendation list and the better the recommendation performance. In the experiments, the value of nDCG@N is calculated for each member of the group, and the nDCG@N value of the group is expressed as the mean value.

F-measure is used to evaluate the missing prediction and group rating classification for members [28], and it is another metric we use to measure the performance of group recommendations. F-measure is calculated by the following formula:

$$F = \frac{2\text{TP}}{2\text{TP} + \text{FN} + \text{FP}}, \tag{14}$$

where TP denotes the number of items in the recommendations where all member ratings are greater than the threshold and the group ratings are also greater than the threshold, FN denotes the number of items in the recommendations where all member ratings are greater than the threshold and the group ratings are less than the threshold, and FP denotes the number of items in the recommendations where some member ratings are below the threshold and the group ratings are above the threshold.

5.3. Baselines and Experimental Setup. To verify the effectiveness of the DMA, we selected eight of the more successful and popular methods as baselines for comparison, as shown in Figure 7. Among them, in the predicting user rating (i.e., rating matrix filled) stage, the comparison algorithms are the timeSVD++ model and the SVD model; in the phase of generating group ratings, the comparison algorithms are the average strategy (AVG) [7], the least misery strategy (LM) [8], and the most pleasure strategy (MP) [9]. Also, for comparison with the latest group recommendation algorithms, we selected the WBF algorithm proposed by Ortega et al. [38] and the MCS algorithm proposed by Wang et al. [28]. Below are the labels and descriptions we use to denote each of these baselines.

(i) AVG: Use the average strategy to generate a group profile where the group rating for a candidate item is equal to the average of the ratings of the members.

(ii) LM: Use the least misery strategy to generate a group profile where the group rating for a candidate item is equal to the minimum value of the ratings of the members.

(iii) MP: Use the most pleasure strategy to generate a group profile where the group rating for a candidate item is equal to the maximum of the ratings of the members.

(iv) WBF [38]: Add weight to each item based on the number of times each item has been rated by group members, and use a weighted ridge regression method to solve for the group rating.

(v) MCS [28]: Add weight to each member based on their contribution to the group, and the group rating is equal to the weighted average of the member ratings.

In summary, the baseline algorithms we set up are SVD_AVG, SVD_LM, and SVD_MP algorithms based on SVD; timeSVD++_AVG, timeSVD++_LM, and timeSVD++_MP algorithms based on timeSVD++; and the latest group recommendation algorithms WBF and MCS in the literature. For the DMA and SVD_Act algorithms, they

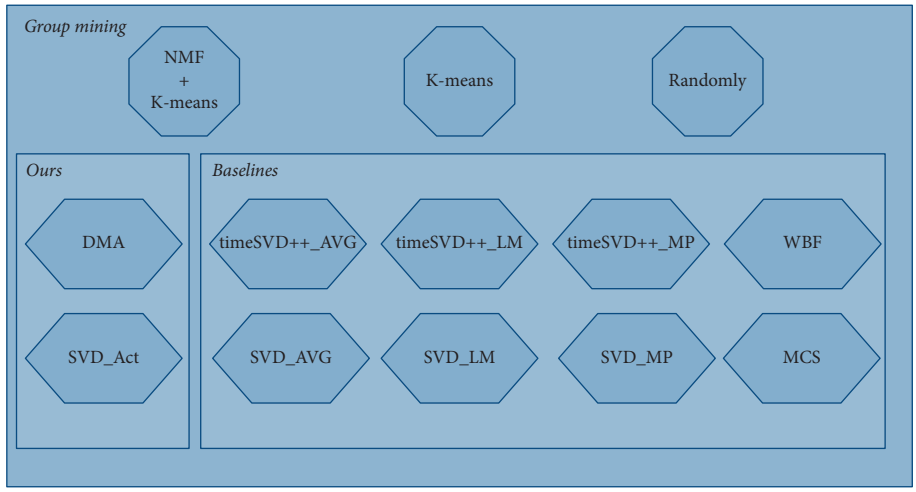

Figure 7: Experimental setup.

correspond to the group recommendation algorithm based on member activity level after the rating matrix is filled by timeSVD++ and SVD, respectively.

In terms of the experimental setup, we set up three group mining algorithms in our experiments, so that the above group recommendation methods have experimented under different group sizes mined by different group mining algorithms to verify their robustness and the effectiveness of the group mining algorithms proposed in this paper.

5.4. Experimental Results and Parameter Analysis

5.4.1. Experimental Results Analysis. To verify the improvement of the DMA algorithm, we conducted experiments at each group size corresponding to different group mining algorithms in the MovieLens dataset and the Netflix dataset and calculated their corresponding *n*DCG@10 and F-measure. To facilitate comparison, a consistent parameter was set in the SVD model and the timeSVD++ model, and Table 2 shows the details of the parameters we set in our experiments. As for the WBF and MCS algorithms, we referred to the experimental parameters in the relevant papers.

Tables 3 and 4, respectively, show the experimental results of *n*DCG@10 under different group sizes corresponding to different group mining algorithms in the MovieLens and Netflix datasets. The group size is set to 5, 10, and 20, respectively, the recommended moment t_{rec} is selected randomly, and the number of recommendations N is set to 10.

In the results shown in Tables 3 and 4, DMA and SVD_Act are our proposed group recommendation algorithms. The following contents can be analyzed from the experimental results in Tables 3 and 4. Firstly, our proposed algorithm achieves better accuracy in group recommendations, further revealing the effectiveness of considering the activity level of group members when generating group recommendations. Secondly, the accuracy of the group recommendations generated by using the timeSVD++ model for rating prediction and combining with the corresponding rating aggregation strategies is significantly higher than that of the recommendations generated by using the SVD model. This is because the timeSVD++ model can accurately capture the interest drift of users and predict their preferences more precisely. Again, among the groups detected by different group mining algorithms, the group mining algorithm proposed in this paper presents better recommendation accuracy for the corresponding group size. This is because the proposed group mining algorithm can detect groups that are more similar internally, which results in fewer preference conflicts inside them, thus achieving higher recommendation accuracy. Finally, it can be observed that as the group size increases, the recommendation accuracy roughly presents a decreasing trend. The possible reason is that as the number of users in the group increases, the conflict within the group becomes all the more fierce, and the preferences become more difficult to aggregate, thus leading to a decrease in accuracy. But at the same time, as the number of members increases, the difference in activity level is bound to be exposed. Our algorithm slows down the decreasing trend of recommendation accuracy by analyzing the recent activity level of group members and aggregating the ratings based on it, which indicates the effectiveness of our algorithm from the side.

Table 1: Statistics of the datasets.

Dataset	#Users	#Items	#Ratings	Rating range	Sparsity (%)
ML-100K	943	1682	100000	[1, 5]	93.7
Netflix	480189	17770	100480507	[1, 5]	98.8

Table 2: Details of experimental parameters.

Parameter	MovieLens	Netflix
k (number of factors)	20	10
λ (regularization coefficient)	0.004	0.002
Max. iterations	500	500
Threshold (in F-measure)	3	3
η (sliding window length)	90	90

Tables 5 and 6, respectively, show the experimental results of F-measure under different group sizes corresponding to different group mining algorithms in the MovieLens and the Netflix datasets. The group size is set to 5, 10, and 20, respectively; the recommended moment t_{rec} is selected randomly; and the number of recommendations N is set to 10.

The experimental results of the F-measure are presented in Tables 5 and 6. Firstly, compared with other baselines, our proposed algorithm obtains higher F-measures for different group sizes in different datasets in general, indicating that our algorithm performs better compared with other algorithms. Secondly, the F-measure obtained by combining the timeSVD++ model with the baseline algorithms is higher than those of the corresponding SVD model. One possible reason is that the timeSVD++ model can generate more accurate user ratings, which provides a better basis for subsequent rating aggregation, illustrating that predicting more accurate user preferences is an essential prerequisite for group preference aggregation. Again, compared to the MovieLens dataset, the F-measure computed in the Netflix dataset is lower overall, which may be caused by the Netflix dataset being too sparse. On the one hand, data sparsity leads to lower prediction accuracy; on the other hand, data sparsity also leads to poor performance of the group mining algorithm. Finally, it can be seen that among the three different group generation strategies, the value of F-measure under the groups detected by the group mining algorithm in this paper is higher, which further demonstrates the effectiveness of our proposed group mining algorithm. Therefore, in order to conduct subsequent experiments under more ideal conditions, all the groups we have used in the subsequent experiments are generated by our proposed group mining algorithm.

5.4.2. Parameter Analysis

(1) Number of Latent Factors k. In the singular value decomposition, the value of k affects the accuracy of the prediction ratings. A small set of the k value will result in the latent features not being adequately represented, while a large set will increase the computational complexity. Therefore, we fixed the number of recommendations N as 10, the group size as 20, the sliding window length η as 90, and set $k = \{5, 10, 15, 20, 25, 30\}$, to explore the specific effect of k on each group recommendation algorithm. Because the MCS algorithm is not based on singular value decomposition, it is out of the scope of this experiment. The experimental results are shown in Figure 8.

We can draw the following conclusions based on the experimental results of Figure 8 in both datasets. Firstly, the recommendation performance of the group recommendation algorithm based on the latent factor decomposition is significantly influenced by the value of k. This is reflected in the fact that the recommended performance becomes better when the value of k is gradually increased in the case of a smaller value than the optimal value, and it reaches the best value when k takes the optimal value at the same time. When it continues to increase with a value of k greater than the optimal value, the recommended performance starts to decrease. This may be due to the difference in the accuracy of the predicted ratings generated from different k values. The reason for this phenomenon may be due to the differences in data features in the two datasets. Again, it can be observed from the experimental results that the algorithm proposed in this paper achieves a better recommendation performance when all k values are optimal. Finally, considering the above analysis, we set k to 20 in the MovieLens dataset and k to 10 in the Netflix dataset in the subsequent experiments.

(2) Number of Recommendations N. In this experiment, the number of recommendations is set to $N = \{10, 20, 30, 40\}$, and the remaining parameters are the same as above, to explore the specific effect of the number of recommendations on the performance of the group recommendation algorithm, and the experimental results are shown in Figure 9.

According to the experimental results in Figure 9, firstly, the algorithm performance shows an inverse ratio with the number of recommendations in both datasets. When the number of recommendations increases, the performance of the algorithms gradually decreases without exception. The possible reason is that the number of items liked by all members in the group is small, and when the number of recommendations increases, the items recommended afterward do not meet the needs of most members, thus leading to a decrease in recommendation performance. Secondly, it can also be seen from Figure 9 that the rate of decline of nDCG@10 is faster when the number of recommendations is between 10 and 20, while the rate of decline becomes slower after exceeding 20, which may be due to the small difference in the degree of preference of group members for subsequent recommended items. Finally, the algorithm of this paper shows better recommendation performance than other algorithms under the corresponding different recommendation quantities, which further verifies the effectiveness of the algorithm of this paper.

(3) Sliding Time Window Length η. This experiment sets the sliding time window length to $\eta = \{10, 30, 60, 90, 180, 360\}$, and the rest of the parameters are kept the same as above to explore the effect of the sliding time window length on our algorithm, and the experimental results are shown in Figure 10.

According to the experimental results in Figure 10, it can be seen that the algorithm in this paper is significantly influenced by the sliding long window. Firstly, when the sliding time window is small, the recommendation performance of the algorithm in this paper is poor. The possible reason is that the activity level calculated by the algorithm in this paper is more one-sided in the short sliding time window, which cannot capture the real activity level of the group members, thus making the recommended items more inclined to the very small number of recently active members and ignoring most of the rest members, which leads to the poor recommendation performance. Secondly, as the sliding time window gradually increases, the recommendation performance of the algorithm in this paper increases significantly and outperforms other baseline algorithms at a sliding time window size of 90. The baseline

Table 3: Experimental results of nDCG@10 in MovieLens.

Algorithm	NMF + K-means			K-means			Randomly		
	5	10	20	5	10	20	5	10	20
SVD_AVG	0.897	0.878	0.859	0.801	0.807	0.791	0.153	0.176	0.159
SVD_LM	0.911	0.881	0.839	0.830	0.839	0.816	0.213	0.228	0.174
SVD_MP	0.883	0.823	0.771	0.811	0.785	0.738	0.164	0.157	0.102
SVD_Act	0.881	0.871	0.915	0.823	0.820	0.837	0.160	0.181	0.154
timeSVD++_AVG	0.9	0.891	0.868	0.826	0.831	0.806	0.172	0.236	0.201
timeSVD++_LM	0.915	0.893	0.863	0.843	0.853	0.840	0.196	0.244	0.183
timeSVD++_MP	0.896	0.854	0.779	0.836	0.815	0.768	0.192	0.167	0.125
WBF [34]	0.893	0.897	0.878	0.878	0.889	0.850	0.182	0.216	0.185
MCS [24]	0.901	0.884	0.867	0.885	0.876	0.853	0.209	0.215	0.158
DMA	0.892	0.906	0.929	0.851	0.877	0.875	0.197	0.228	0.176

Table 4: Experimental results of nDCG@10 in Netflix.

Algorithm	NMF + K-means			K-means			Randomly		
	5	10	20	5	10	20	5	10	20
SVD_AVG	0.875	0.854	0.816	0.786	0.801	0.768	0.145	0.165	0.116
SVD_LM	0.892	0.887	0.856	0.820	0.798	0.786	0.224	0.215	0.185
SVD_MP	0.835	0.814	0.776	0.714	0.701	0.718	0.153	0.153	0.118
SVD_Act	0.864	0.858	0.874	0.804	0.793	0.782	0.173	0.151	0.158
timeSVD++_AVG	0.884	0.873	0.853	0.790	0.804	0.772	0.172	0.179	0.126
timeSVD++_LM	0.907	0.885	0.873	0.837	0.818	0.790	0.216	0.239	0.179
timeSVD++_MP	0.843	0.826	0.788	0.752	0.737	0.728	0.186	0.163	0.120
WBF [34]	0.897	0.890	0.862	0.857	0.827	0.816	0.179	0.202	0.169
MCS [24]	0.915	0.897	0.854	0.874	0.826	0.804	0.214	0.205	0.163
DMA	0.885	0.915	0.903	0.844	0.854	0.826	0.189	0.214	0.172

Table 5: Experimental results of F-measure in MovieLens.

Algorithm	NMF + K-means			K-means			Randomly		
	5	10	20	5	10	20	5	10	20
SVD_AVG	0.711	0.653	0.516	0.612	0.558	0.432	0.154	0.128	0.101
SVD_LM	0.702	0.522	0.462	0.579	0.536	0.413	0.105	0.082	0.079
SVD_MP	0.683	0.517	0.472	0.617	0.544	0.502	0.169	0.134	0.119
SVD_Act	0.698	0.538	0.496	0.583	0.560	0.513	0.135	0.127	0.092
timeSVD++_AVG	0.729	0.672	0.519	0.625	0.562	0.453	0.191	0.169	0.139
timeSVD++_LM	0.708	0.543	0.470	0.585	0.531	0.422	0.190	0.154	0.118
timeSVD++_MP	0.693	0.547	0.484	0.622	0.573	0.496	0.204	0.169	0.115
WBF [34]	0.742	0.652	0.578	0.623	0.562	0.499	0.342	0.323	0.272
MCS [24]	0.732	0.678	0.606	0.646	0.585	0.516	0.352	0.311	0.264
DMA	0.744	0.664	0.628	0.613	0.582	0.548	0.339	0.326	0.289

Table 6: Experimental results of F-measure in Netflix.

Algorithm	NMF + K-means			K-means			Randomly		
	5	10	20	5	10	20	5	10	20
SVD_AVG	0.512	0.467	0.428	0.432	0.388	0.328	0.130	0.134	0.114
SVD_LM	0.484	0.426	0.379	0.376	0.314	0.275	0.147	0.141	0.117
SVD_MP	0.476	0.418	0.386	0.335	0.301	0.284	0.109	0.107	0.098
SVD_Act	0.462	0.438	0.401	0.352	0.317	0.295	0.131	0.122	0.101
timeSVD++_AVG	0.538	0.519	0.475	0.463	0.415	0.367	0.139	0.142	0.125
timeSVD++_LM	0.528	0.487	0.443	0.424	0.373	0.339	0.145	0.136	0.130
timeSVD++_MP	0.522	0.463	0.416	0.397	0.352	0.316	0.119	0.109	0.098
WBF [34]	0.604	0.536	0.483	0.473	0.425	0.372	0.244	0.205	0.156
MCS [24]	0.622	0.561	0.501	0.495	0.431	0.395	0.265	0.199	0.164
DMA	0.617	0.572	0.527	0.462	0.436	0.414	0.260	0.236	0.191

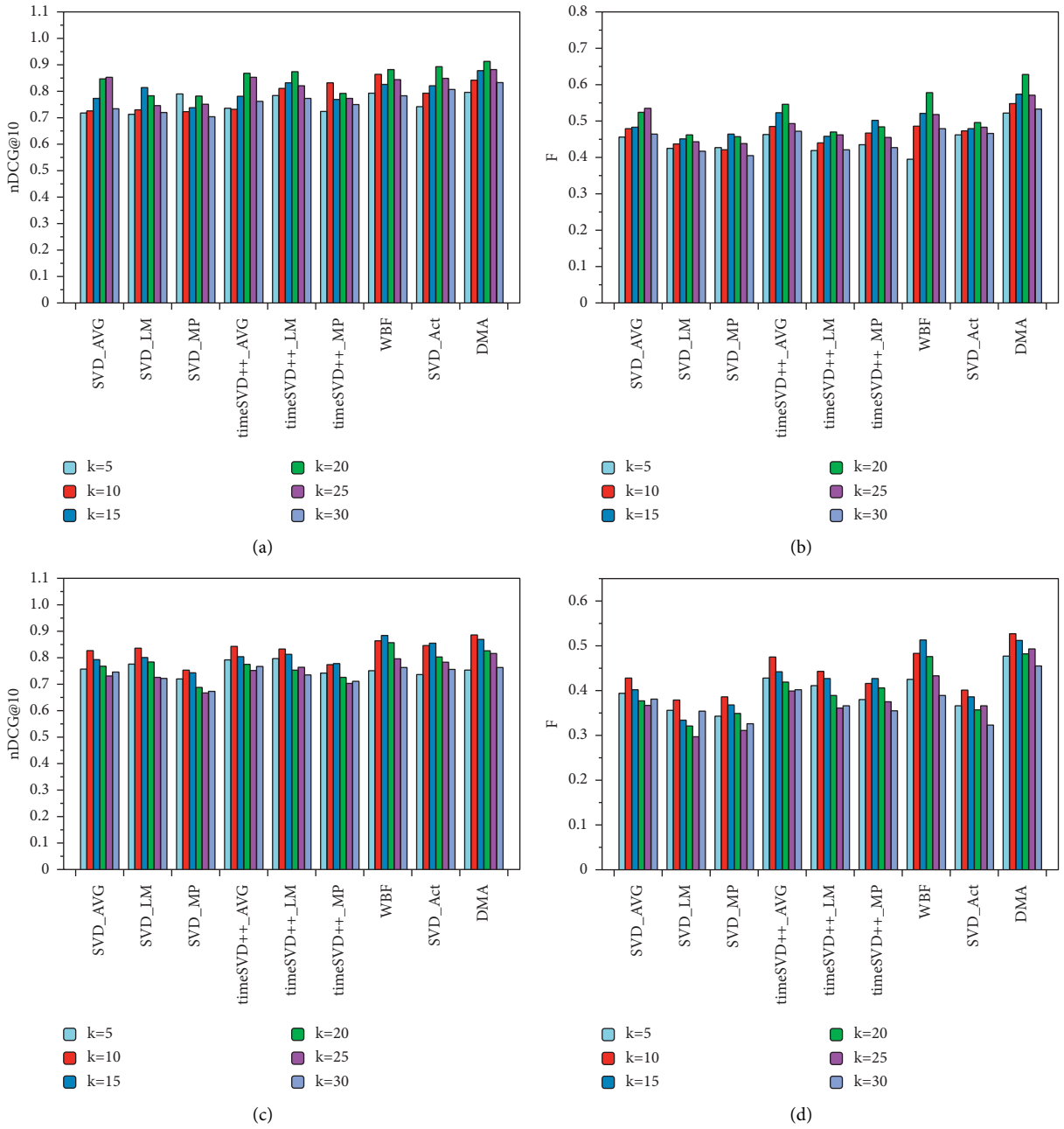

Figure 8: Performance comparison of different algorithms at different values of k in two datasets. (a) MovieLens-nDCG@10, (b) MovieLens-F, (c) Netflix-nDCG@10, and (d) Netflix-F.

algorithms are not affected by the sliding time window because they do not take the time factor into account. Finally, as the sliding time window continues to increase, the recommendation performance of the algorithm in this paper gradually decreases, probably because after the sliding time window is set too large, the algorithm mines the activity level of group members too far away from the current moment, which has a biased impact on the investigation of the recent activity level, resulting in the inability to distinguish the difference of the recent activity level, thus leading to a decrease in the recommendation effect. To summarize the results of this experiment, the algorithm in this paper can achieve better recommendation results when the appropriate sliding time window size is set.

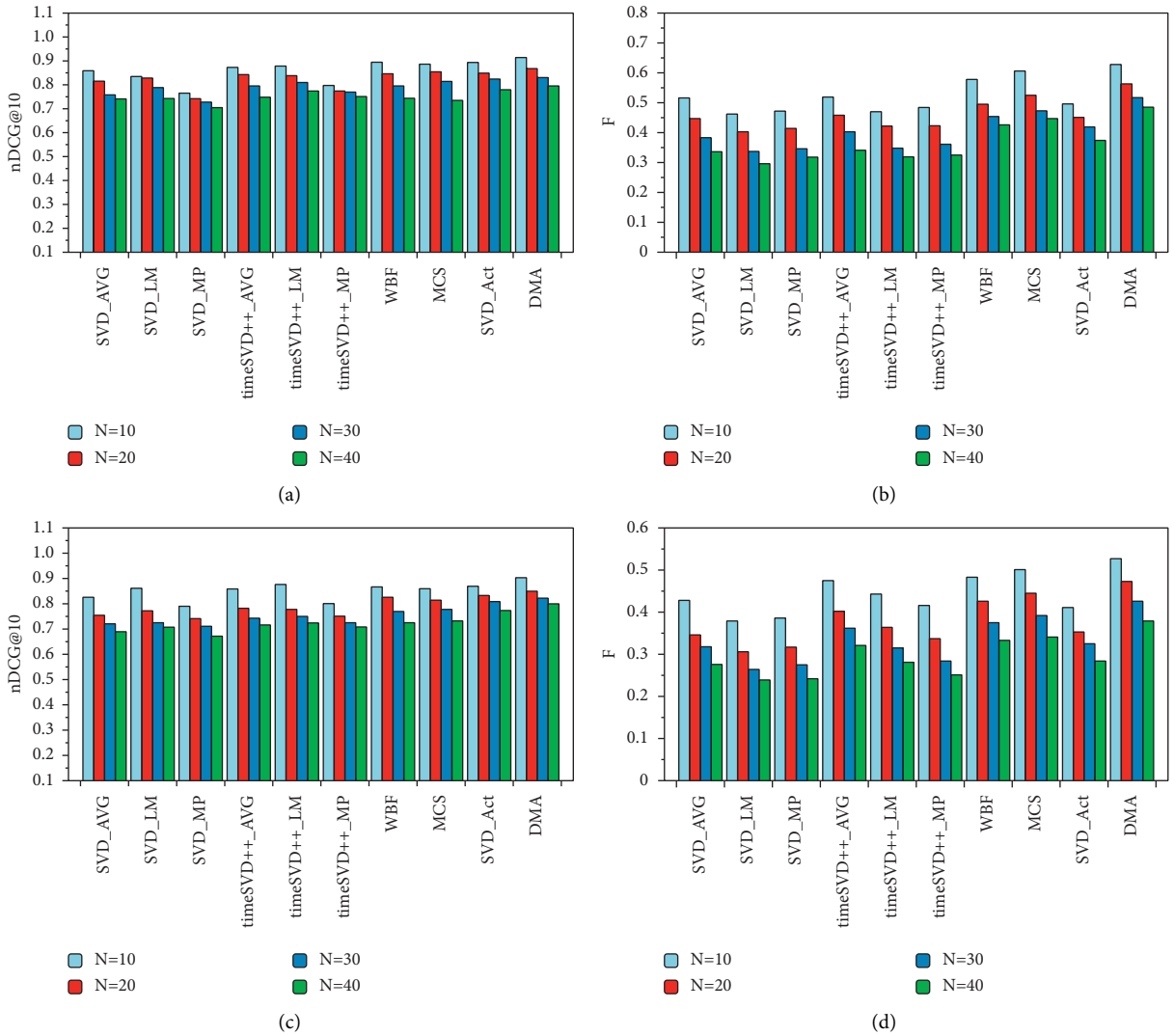

Figure 9: Performance comparison of different algorithms on different number of recommendations in two datasets. (a) MovieLens-nDCG@10, (b) MovieLens-F, (c) Netflix-nDCG@10, and (d) Netflix-F.

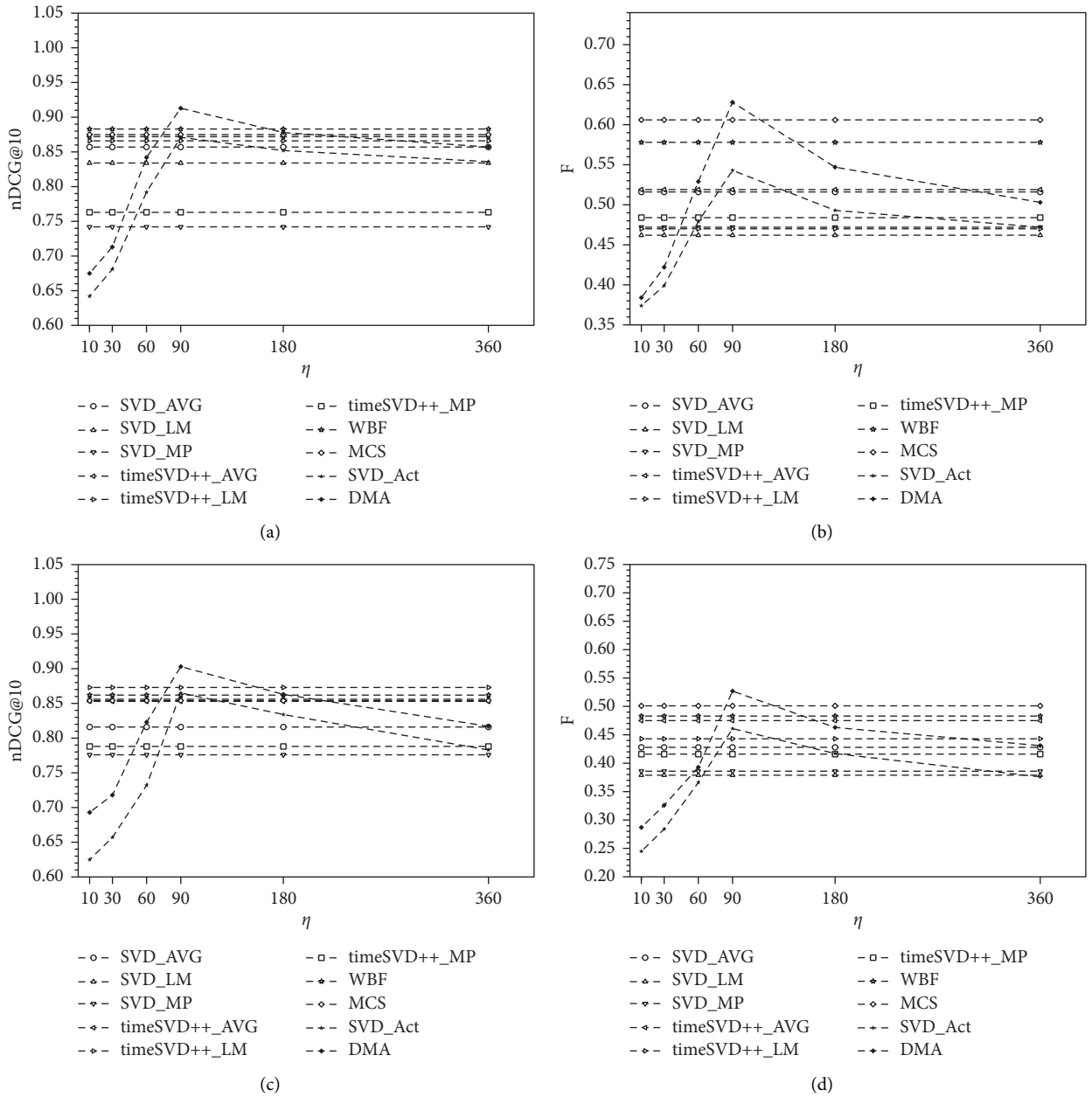

Figure 10: Performance comparison of different algorithms at different lengths of sliding windows in two datasets. (a) MovieLens-nDCG@10, (b) MovieLens-F, (c) Netflix-nDCG@10, (d) Netflix-F.

6. Conclusion

We proposed a dynamic group recommendation algorithm based on member activity level. The algorithm first captured the interest drift of users by using the existing time-series-oriented personalized recommendation algorithm, mined the recent activity level of group members by designing a sliding time window based on improving the accuracy of user preference prediction, and constructed a group model by combining the recent activity level of members for preference aggregation to generate a group recommendation list. The effectiveness of our algorithm is verified by performing experiments on publicly available datasets.

Although our research can effectively improve the performance of group recommendations by mining the recent activity level of members, deeper exploration is still needed in future work. On the one hand, the concept of recent is vague, and it relies too much on real factors such as industry rules and data distribution, which need to be determined by repeated experiments in practical application scenarios. Therefore, it is worthwhile to explore how to summarize the general rules of different industries and fields based on the distribution and trends of data in the next step. On the other hand, new group recommendation algorithms also require further exploration by researchers.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 61967013). This work was also supported by the Higher Education Innovation Ability Enhancement Project of Gansu Province (Grant no. 2019A-006).

References

[1] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE Internet computing*, vol. 7, no. 1, pp. 76–80, 2003.

[2] H. Ma, H. Yang, M. R. Lyu et al., "Sorec: social recommendation using probabilistic matrix factorization," in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pp. 931–940, Napa Valley, CA, USA, October 2008.

[3] J. Tang, X. Hu, and H. Liu, "Social recommendation: a review," *Social Network Analysis and Mining*, vol. 3, no. 4, pp. 1113–1133, 2013.

[4] W. Song, Z. Xiao, Y. Wang et al., "Session-based social recommendation via dynamic graph attention networks," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 555–563, Melbourne, Australia, February 2019.

[5] Y. Huang, B. Cui, J. Jiang et al., "Real-time video recommendation exploration," in *Proceedings of the 2016 International Conference on Management of Data*, pp. 35–46, San Francisco, CA, USA, July 2016.

[6] A. Van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," *Advances in Neural Information Processing Systems*, vol. 26, pp. 2643–2651, 2013.

[7] H. Mahyar, K. E. Ghalebi, S. M. Morshedi et al., "Centrality-based group formation in group recommender systems," in *Proceedings of the 26th International Conference on World Wide Web Companion*, pp. 1187–1196, Perth, Australia, April 2017.

[8] A. Agarwal, M. Chakraborty, and C. R. Chowdary, "Does order matter? effect of order in group recommendation," *Expert Systems with Applications*, vol. 82, pp. 115–127, 2017.

[9] L. Boratto, S. Carta, and G. Fenu, "Discovery and representation of the preferences of automatically detected groups: exploiting the link between group modeling and clustering," *Future Generation Computer Systems*, vol. 64, pp. 165–174, 2016.

[10] J. Masthoff, "Group recommender systems: aggregation, satisfaction and group attributes," in *Recommender systems handbook*, pp. 743–776, Springer, Boston, MA, USA, 2015.

[11] K. W. Back, "Influence through social communication," *Journal of Abnormal and Social Psychology*, vol. 46, no. 1, pp. 9–23, 1951.

[12] H. Blumer, "Social problems as collective behavior," *Social Problems*, vol. 18, no. 3, pp. 298–306, 1971.

[13] G. Le Bon, *The Crowd: A Study of the Popular mind*, TF Unwin, London, UK, 1897.

[14] F. H. Allport, "The group fallacy in relation to social science," *American Journal of Sociology*, vol. 29, no. 6, pp. 688–706, 1924.

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

[16] L. M. De Campos, J. M. Fernández-Luna, J. F. Huete et al., "Managing uncertainty in group recommending processes," *User Modeling and User-Adapted Interaction*, vol. 19, no. 3, pp. 207–242, 2009.

[17] I. Garcia, L. Sebastia, and E. Onaindia, "On the design of individual and group recommender systems for tourism," *Expert Systems with Applications*, vol. 38, no. 6, pp. 7683–7692, 2011.

[18] L. Baltrunas, T. Makcinskas, and F. Ricci, "Group recommendations with rank aggregation and collaborative filtering," in *Proceedings of the Fourth ACM Conference on Recommender Systems*, pp. 119–126, Barcelona, Spain, September 2010.

[19] Y.-J. Zhang, D. Yu-Lu, and M. Xiang-Wu, "Research on group recommender systems and their applications," *Chinese Journal of Computers*, vol. 39, no. 4, pp. 745–764, 2016.

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

[21] J. Masthoff, "Group recommender systems: combining individual models," in *Recommender Systems Handbook*, pp. 677–702, Springer, Boston, MA, USA, 2011.

[22] L. Quijano-Sanchez, J. A. Recio-Garcia, B. Diaz-Agudo et al., "Social factors in group recommender systems," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 4, no. 1, pp. 1–30, 2013.

[23] J.-T. Chen, T.-L. Gu, L. Chang et al., "A tourist group recommendation method combining collaborative filtering and user preferences," *CAAI transactions on intelligent systems*, vol. 13, no. 6, pp. 999–1005, 2018.

[24] A. Jameson, "More than the sum of its members: challenges for group recommender systems," in *Proceedings of the Working Conference on Advanced Visual Interfaces*, pp. 48–54, Gallipoli, Italy, May 2004.

[25] Y.-C. Tao, X. Ding, L. Shi et al., "Group POI recommendation model based on the user check-in behavior," *Journal of Chinese Computer Systems*, vol. 39, no. 10, pp. 2260–2265, 2018.

[26] L. Ardissono, A. Goy, G. Petrone et al., "Intrigue: personalized recommendation of tourist attractions for desktop and hand held devices," *Applied Artificial Intelligence*, vol. 17, no. 8-9, pp. 687–714, 2003.

[27] Q. Yuan, G. Cong, and C. Y. Lin, "COM: a generative model for group recommendation," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 163–172, New York, NY, USA, August 2014.

[28] W. Wang, G. Zhang, and J. Lu, "Member contribution-based group recommender system," *Decision Support Systems*, vol. 87, pp. 80–93, 2016.

[29] S. Berkovsky and J. Freyne, "Group-based recipe recommendations: analysis of data aggregation strategies," in

Proceedings of the Fourth ACM Conference on Recommender Systems, pp. 111–118, Barcelona, Spain, September 2010.

[30] Y. Ding and X. Li, "Time weight collaborative filtering," in *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pp. 485–492, Bremen Germany, October 2005.

[31] G.-F. Sun, L. Wu, L. Qi et al., "Recommendations based on collaborative filtering by exploiting sequential behaviors," *Journal of Software*, vol. 24, no. 11, pp. 2721–2733, 2013.

[32] P. Zhang, Z. Zhang, T. Tian et al., "Collaborative filtering recommendation algorithm integrating time windows and rating predictions," *Applied Intelligence*, vol. 49, no. 8, pp. 3146–3157, 2019.

[33] Y. Koren, "Collaborative filtering with temporal dynamics," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 447–456, Paris France, July 2009.

[34] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[35] D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," *Advances in Neural Information Processing Systems*, vol. 13, pp. 556–562, 2000.

[36] L. Boratto, S. Carta, and M. Satta, "Groups identification and individual recommendations in group recommendation algorithms," in *Proceedings of the Practical Use of Recommender Systems, Algorithms and Technologies 2010*, pp. 27–34, Barcelona, Spain, September 2010.

[37] G. Zhou, A. Cichocki, and S. Xie, "Fast nonnegative matrix/tensor factorization based on low-rank approximation," *IEEE Transactions on Signal Processing*, vol. 60, no. 6, pp. 2928–2940, 2012.

[38] F. Ortega, A. Hernando, J. Bobadilla et al., "Recommending items to group of users using matrix factorization based collaborative filtering," *Information Sciences*, vol. 345, pp. 313–324, 2016.