Chapter 2

Preliminaries and Literature Survey

In this chapter, we discuss preliminaries that are important to understand the concepts associated with group recommender systems. We also discuss related works on group recommendations and identify several research gaps in the existing models for auto-detecting groups. Parts of this chapter are based on our survey paper [35] and our journal articles published in [69, 70].

2.1 Preliminaries

Group recommendations work on aggregating individual user's prefered items into a recommendation vector of length k using consensus function, where k is a group budget (also known as space budget [1,2]). The main task in a group recommender system is to design a model that integrates all the preferences of the group members. This integration is done with the help of different aggregation strategies [1] chosen according to the requirements of the group.

Preference aggregation strategies, i.e., least misery [1, 77] and aggregated voting [1,77]) are used to generate a single recommendation vector that is suitable for the whole group. Aggregated voting and Least misery are consensus functions.

- Aggregated voting: Sum of individual satisfaction scores of all the users are maximized in aggregated voting [77].

  Example 1: Suppose there are *n(=4)* users in a group and *m(=5)* items with the group budget of *k(=2)*. The preferences given by the users are:

[1]Methods that combine individual preferences/recommendations are called aggregation strategies.

User1: {1, 3}

User2: {4, 2}

User3: {1, 5}

User4: {1, 2}

In this case, considering all possible permutations of size (k=2), we have ${}^{m}P_{k}$=20 possible recommendations. This strategy considers the preferences of group members intending to maximize the overall group satisfaction. Table 2.1 gives an item-satisfaction score based on user preferences where the frequency of an item in the group is its satisfaction value.

Aggregated voting (AV) picks top k(=2) items having the highest satisfaction value in the recommended list. Here, item set {1, 2} maximizes group satisfaction, and hence it is the final recommendation of aggregated voting.

- Least misery: In the least misery [77], the minimum satisfaction score among all users is maximized. Initially, every user's default satisfaction value is considered as 0. It operates in k iterations wherein the first iteration, the item providing the highest group satisfaction is selected and is added to the recommendation list. In the remaining $(k-1)$ iterations, user satisfaction values are updated before selecting a least satisfied user. Then a least satisfied user is selected, and the item with the highest satisfaction from her preference vector is added to the recommendation list. In Example 1, item 1 is selected in the first iteration and added to a recommendation list as it provides the highest satisfaction (Table 2.1). After the first iteration, the updated satisfaction score of users is shown in Table 2.2. In the second iteration, the consensus function chooses the item from the second user's preferences which has maximum group satisfaction (Table 2.1). Here, item 2 provided maximum satisfaction and is added to the final recommendation list. Hence, for a group budget (k=2), item set {1,2} is the final generated recommendation list for the least misery approach.

Table 2.1: Group satisfaction with respect to items.

Items	1	2	3	4	5
Satisfaction	3	2	1	1	1

Table 2.2: Individual user satisfaction after selecting item 1.

User	1	2	3	4
Satisfaction	1	0	1	1

2.2 Literature survey

2.2.1 Order preference model

The order preference model provides a set of items that users would like to see in the generated recommendation list along with an order in which a user likes those items to appear [1]. In the available literature, none of the group satisfaction measures consider the order of preferences; therefore, User Satisfaction with Order (USO) [1] is proposed.

USO is computed as:

$$uso_j = uso_j + f(x, a) \tag{2.1}$$

$$f(x, a) = c + \frac{k}{(x+1)^{\frac{1}{a}}} \tag{2.2}$$

Where k is the group budget, x is an absolute difference between the position the user wants the item to appear in and the position at which the item appears in the generated recommendation, i.e.,$(0 < x < k - 1)$. a is a constant, and its value is inversely proportional to the role of the order to determine the recommendation. Taking a higher value of a results in an insignificant role of order and vice-versa. c is a regularization factor and takes a constant value. The satisfaction score of any user j at any iteration depends on the currently recommended item and his position in the recommender list, so initially, the satisfaction score of all users will be zero. When the item is taken in the recommendation list, the satisfaction of every user j to that item is measured using $f(x, a)$. If an item is not listed in the preference list (pl_j) of user j, the satisfaction score will remain intact. This metric helps to find the satisfaction of each user from the generated recommendation vector. A user's satisfaction reflects the similarity of her preferences with the generated

recommendation.

Greedy Aggregated method (GRAM) and Hungarian aggregated method (HAM) that are the extensions to the aggregated method are proposed in [1]. Greedy aggregated method (GRAM) reduces the computational cost compared to aggregated voting to improve accuracy. They also propose the least misery method (LMM) and least misery method with priority (LMMP) that are the extensions to least misery. Experimental results show that LMMP provides better group satisfaction than LMM. It achieves by choosing the least misery user in every iteration based on the priority of each user. Thus, LMMP increases the overall group satisfaction. The overall group satisfaction produced by the HAM method is also overwhelming in smaller group sizes and fewer preferred items.

2.2.2 Flexible size preference model

The objective is to design a model by including flexibility in users' preferences where the size of the preference list would be less than or equal to k, which provides a recommended list of items of group budget (k) [35].

Flexible preferences without order:

1. Aggregated voting: In this method for each user; k points are equally distributed among the items in her preference list by $\frac{k}{m}$ where k is the group budget and m is the number of items in her preference list. The points of an item to the group are calculated by adding points with respect to each interested user of the group for that item. Finally, the top-k items are taken in the recommended list.

   Example 2: Suppose there are *n(=7)* users in a group and *m(=8)* items with the group budget of *k(=5)*. Each user preferences are given as follows:
   User1: {2,6,7}
   User2: {2,3,4,6,7}
   User3: {3,8}
   User4: {1,2,6,8}
   User5: {5,7}
   User6: {1,3,4,5,8}
   User7: {7}

Table 2.3: User Preference Matrix

Position	Items							
	1	2	3	4	5	6	7	8
1	2.25	2.67	2.5	0	2.5	0	5	0
2	0	1.25	2	0	0	1.67	2.5	2.5
3	0	0	0	2	0	1.25	1.67	0
4	0	0	0	0	1	1	0	1.25
5	0	0	0	0	0	0	1	1

Table 2.4: Points Vector(P)

Item_id	1	2	3	4	5	6	7	8
Points	2.25	3.92	4.5	2	3.5	3.92	10.17	4.75

In this method for each user; k points are equally distributed among m items by $\frac{k}{m}$. The points of an item to the group is calculated by adding points regarding each interested user of group for that item. These values are stored in point vector. Table 2.4 with a corresponding item number as an index. Finally, the top-k items from points vector (Table 2.4) are taken in the recommended list. For the above example top-5 items $< 2, 3, 6, 7, 8 >$ from the Points vector(P), with total group satisfaction **27.26** are selected.

2. Least misery method with priority: The least misery method with priority (LMMP) is an extension of least misery, which uses the priority to break the ties if multiple users hold the same satisfaction. We proposed a model to compute the priority of users. Initially, the satisfaction of all users of the groups is 0. So, the user with the highest priority value is selected as the least misery user. In Example 2, user 2 is considered as the least misery user, and from his item preference set {2, 3, 5, 6, 7 } item 7 is selected (item is selected that maximizes the group satisfaction according to Points vector (P)(Table 2.4))and added to the recommendation set. The recommended list of items according to least misery with priority is $< 3, 5, 6, 7, 8 >$ with overall group satisfaction **26.84**.

Table 2.5: Priority value of users

user	1	2	3	4	5	6	7
Priority	1.58	1.66	0.77	1.17	1.08	1.1	1.03

Table 2.6: User Satisfaction Matrix

Position	Items							
	1	2	3	4	5	6	7	8
1	13.5	21.7	24.07	7.77	18.5	15.92	51.06	18.95
2	10.2	19.59	23.34	9.07	15.23	19.56	48.74	23.36
3	8.75	16.03	18.79	12	14.26	19.6	44.66	20.9
4	7.88	14.19	16.52	9.07	14.75	18.15	39.32	21.76
5	7.28	13.01	15.09	7.77	12.63	15.23	37.41	20.42

Flexible preferences with order:

In this model, each group user has a variable-size preference vector where the size is less than or equal to k. The objective is to generate a recommendation vector of length k for variable-size order preferences. Table 2.3 represents user preference matrix.

1. Modified least misery method with priority (MLMMP): Modified Least Misery Method with Priority (MLMMP) is an extension of the least misery method and considers flexible preferences. This method examines similarity among users' preferences to resolve the ties in the least misery values. The priority of a user is computed from the preference vectors of the group. Priority[i] represents the similarity of a user i to the other users. A user's higher similarity value gives him a higher priority in a group. The procedure to compute the priority of a user is given in COMPUTE_PRIORITY procedure (Algorithm 2). In Example 2, to compute priority score for User1, we calculate the similarity of preference vector lis_1 to the vectors lis_2,lis_3.....lis_7 pairwise by using either Jaccard similarity or overlap similarity method. After summing up these values, we get a priority[1]=**31.29**. Priority values for all users are given in Table 2.7. In this method, the user with the highest priority is considered as the least misery user for the current state, and

Algorithm 1: Computing satisfaction of individual user when flexibility introduced with user preferences

Input: i, **lis[i]**, **r**

Output: satisfaction score as t_{sum}

```
t_sum = 0;
for 0 ≤ p < len(r) do
    item = r[p];
    if (item in lis[i]);
     then
        idx = lis[i].index(item);
        t_sum = t_sum + s[abs(p − idx)];
    end
end
```

the corresponding item that provides the highest group satisfaction is added in the recommendation vector. To avoid conflicts in the item selection in the further iterations, after adding an item j at the position i in the recommendation vector, the corresponding row and column of user satisfaction matrix (S) are reset to 0. This process is iterative and continues until the group budget k is full. In Example 2, this method generates the output $< 7, 8, 6, 3, 5 >$ with overall group satisfaction=**65.2**. The overall group satisfaction is obtained by adding up the individual satisfaction of all users where COMPUTE_SATISFACTION(Algorithm 1) method calculates individual satisfaction.

2. Modified GReedy Aggregation Method (MGRAM): This algorithm uses a greedy approach to select an item for the recommendation. It operates in k iterations; in each iteration, it selects the best item greedily using user satisfaction score matrix (Table 3.4). The MGRAM model reduces computation time as well as increases total group satisfaction by using a greedy approach. Like MLMMP, after selecting an item, this method places zero to corresponding i^{th} row and j^{th} column of S to avoid conflict. For Example 2, this method generates the output $< 7, 8, 6, 3, 2 >$ with total group satisfaction=**67.36**.

Algorithm 2: Procedure for computing priority in MLMMP

Input: **lis**, j, l, $\mathbf{r}_1$

Output: priority of user j as sum

```
1 sum = 0;
2 r_1 = sizeof(lis_j);
3 for 0 ≤ l < len(lis) do
4     if (l ≠ j);
5      then
6         sum = sum+COMPUTE_SATISFACTION(l, lis[l], r_1) ;
7     end
8 end
```

Table 2.7: Priority values for the users in case of modified least misery with priority(MLMMP)

User	1	2	3	4	5	6	7
Priority	31.29	46.24	16.47	32.59	16.08	34.08	11.67

3. Modified Hungarian Aggregated Method (MHAM): A modified hungarian aggregated method is an alternative to the aggregated voting method for flexibility in users' order preferences. The Hungarian method [67] is used in a job assignment problem to assign jobs to an individual, minimizing overall operation cost. It provides better group satisfaction scores compared to other consensus function-based approaches, when smaller group sizes and items are considered. For Example 2, this method generates the output $< 7, 3, 4, 6, 8 >$ with total group satisfaction=**74.65**.

4. Modified most pleasure method with priority(MPMP): Most Pleasure Method with Priority(MPMP) [35] is an add-on of Most Pleasure with Priority(MPP). MPP approach takes into account the order preferences and MPMP method incorporates flexibility in user preferences. MPMP looks for the most satisfied user at each iteration and appends an item to the recommendation vector according to the preferences of the most satisfied user. In each iteration; an item and position are selected from user satisfaction score matrix, so that, adding a product in the recommendation

vector maximizes group satisfaction.

2.2.3 Group Recommender Systems

In [35], we discuss popular group recommender systems available in different domains. Of late, group recommendations have spread to many real-life scenarios viz. a group of researchers discuss online on Mendeley[2], a group of tourists can visit breath-taking sites using mafengwo[3], etc. Meetup[4] helps users to organise and participate in group activities where all the group members share the same interest. The popularity of social activities played a role in developing group recommender systems. The group recommender systems are used in various fields, i.e., Jukola [84] and PartyVote [112] music systems play music at a party or social event. MusicFX [79] is one of the earliest group recommenders, which analyzes a user's personal preferences, and finds the appropriate music for a group of users in a fitness center. These recommendation systems are associated with similar kind of activities and form an established group in group recommendations. Pocket Restaurant Finder [78] provides restaurants for a group to dine together. The formed group is an example of an occasional group. PolyLens [86], an extension of the MovieLens recommender system, recommends a movie to a group of users. It uses the least misery strategy to model a group and collaborative filtering approach to give the preferences for unrated items of the group. After aggregating user preferences, it generates a recommendation for each group user. In Polylens, formed groups are permanent groups, in which 95% users were satisfied with group recommendations using the least misery approach, and 77% of the users were more satisfied with group recommendations than with the personal ones. HappyMovie [97] presents a system that recommends movies to a group of users on Facebook. This system exploits the social and behavioural information about the users to improve the recommendations. In general, too many questionnaires have to be filled to get the social and behavioural information about the user. But, as the system is implemented on a social networking platform, the data is deduced easily. While social and trust profiles of a user are calculated automatically from the user's Facebook profile, personality profile is created through a questionnaire and also she needs to rate some movies.

[2]https://www.mendeley.com/community/researcher-academy/

[3]http://www.mafengwo.cn/

[4]https://www.meetup.com/

Later, collaborative filtering is applied to recommend items based on her profiles.

Top-N Rec [62] works on generating recommendations of Top-N items to the groups which are important mostly in the fields of entertainment, travel, etc. This model generates recommendations to the users using content-based and collaborative filtering approaches individually to each user initially. Later, aggregation strategies are used separately to generate group recommendations. Finally, items recommended by collaborative filtering are reordered based on the content-based results. This model is based on an intuition that items recommended using both methods at the same time could be considered more appropriate than those items recommended by individual techniques. When the proposed model is evaluated on a standard dataset, it is found that it performs better than the content-based, collaborative filtering and many traditional group recommendation techniques. Traditional group recommendation techniques fail to produce better recommendations when the user-item matrix is sparse and the user's preferences are unknown. The data sparsity problem is handled [42], [8], [82] in group recommender systems in different ways. The model in [42] employs item-item similarity to fill the user-item rating matrix. In contrast, Kemeny-ordering and expected rating algorithms are used to rank the items in [8] and [82], respectively. In [130], a variant of average strategies is used to merge user profiles, and it recommends TV programs for many viewers. The proposed system provides one TV program at a time, and most of the viewers were satisfied with the recommendations in homogeneous groups. GRec-OC [64] proposes a book recommender system for online communities. The proposed system works in two stages. The first stage uses a typical collaborative filtering method by merging the profiles of its members to form a group profile. In the second stage, irrelevant items in the recommendation list obtained from the first phase are removed to improve the satisfaction of each group member. Existing works presented in [46, 86, 130] provide evidence that homogeneity or heterogeneity levels affect the quality of recommendation to some more extent.

2.2.4 Memory-based approaches in group recommendation

Memory-based approaches aggregates members' preferences without considering the interactions among members of a users' group. In [2], consensus function is introduced, which comprises two semantics: group relevance and group disagreement. Group relevance maximizes the satisfaction of the group members. In contrast, group disagreement minimizes

the dispute among them. Authors claim that an item can be recommended for the whole group if the group disagreement among the group members is much less for that item. The model can use either the least misery method or aggregated voting strategy to calculate group relevance. In [24], authors applied game theory in group recommendation by considering each group event as a non-cooperative game and addressed recommendation problem by finding the game's Nash equilibrium. Authors in [107] generated recommendations by considering deviation as a vital part of the aggregation method. They introduced a new variant of aggregation method that provides group recommendation by considering deviation in users' rating for an item.

2.2.5 Model-based approaches in group recommendation

The goal of a model based approach is to model group's decision making processes and generating the most suitable items for a group of users. Some popular works on model-based approach for group recommendations are discussed in [75, 128, 131]. COM(COnsensus Model) [131] is based on *Latent Dirichlet Allocation* (LDA) model to simulate the generative process of group events. Here, the authors observed that the recommendation decisions are dominated by the users who have expert skills in the topics related to the group and users' behaviour change in the group and content information that affects the group recommendation. The authors incorporated all these different decision points in the model. Two-step Gibb's sampling method is used for the probabilistic parameter estimation, and experiments performed on four different datasets reflect the effectiveness of the proposed model to a great extent. The model in [43] calculates the item-item similarity using *Support Vector Machines* (SVM), and the computed similarities are used to find similar users, based on the similar items that the user had rated earlier. A memory-based collaborative filtering approach is used to make the predictions, and the resultant improvement was functional. The proposed model outperformed other state-of-the-art baseline models. In [85], the authors aggregate the preferences of the members of the group via the aggregated voting strategy and perform group recommendations using matrix factorization [103] technique.

In [58], a topic-based model named *GIST* not only considers the individual user's interest but also considers the subgroup's interest to make a recommendation. The connectivity of group member who participates in more subgroups gets a higher weight. Such

a member will have more influence in the group. In [21], neural collaborative filtering framework was extended to recommend items for a group of users by combining standard attention neural network with neural collaborative filtering [50]. The framework dynamically updates the weights of the group members and learns the aggregation strategy from historical data. In [38], authors attempt to design a comprehensive approach, random walks based on topic model *(RTM)*. Concisely, a probabilistic transferring graph-based model is defined by integrating *User Topic Model* (UTM) with the *Random Walk with Restart* (RWR). Here interactions among users, groups, and items are modelled as the probabilistic weights. The model makes recommendations by aggregating user preferences and detecting latent relationships among users, items, and groups. Authors in [135] introduce a recommendation model based on Markov Decision Process (MDP). They employ a deep reinforcement learning model (Deep-Q-Network (DQN)). The numbers of skipped items refer to negative feedback. The proposed *DEERS* framework updates the recommendation strategies when a user misses some items. The model continuously improves its approach by modeling the interactions with users.

In [118] *Medley of Sub-Attention Networks* (MoSAN), a novel neural architecture is proposed for group recommendations by adopting the attention mechanism to learn attentive and dynamic weights of each member of the group. The intuition behind this model is that predefined aggregation strategies are inefficient to model the intricacy and tendency of the group during weight adjustment of the members of a group. In contrast, the proposed model can adjust the user's weight, and the user with a higher weight will have more influence in the final group's decision. This model also provides user-user interaction using sub-attention networks.

In [29], the authors propose a system that considers group members' interactions. They assume that in group recommendations, ratings are not just provided by an individual but also by subgroups. This assumption offers different influences on recommendation aggregation. The authors generated an improved genetic algorithm by combining collaborative filtering and genetic algorithm (GA) to get fair weights of group users. This method improves the quality of the system. In [13], the authors propose the Louvain algorithm by using modularity [45] based community detection algorithm for automatic group detection having a similar taste. This method works on a greedy optimization principle that attempts to optimize modularity gain. A group preference for a given item was predicted

by computing an average of members' ratings or calculating the weighted sum of ratings provided by the group on similar items.

In group recommender systems, solving user preference conflicts is a major challenge to be handled. So the method proposed in [102] mainly concentrates on building a consensus aggregating users' preferences in group recommendations as an additional feature to the model presented in CATS [80]. As an extension to the CATS recommender system, consensus negotiation is added to the model to produce better recommendations. Nine consensus negotiation strategies based on statistical, content and collaborative ideas are evaluated and analysed using the live-user preference data of the ski-holiday group recommender system. These strategies are studied by varying group sizes and on different types of user groups. The benefits of each consensus strategy are analysed, which helps in choosing a consensus to be followed in generating group recommendations based on the group requirements and the data available. As not much attention was given to consensus negotiation, this work acted as a starting step to tackle this problem. In [89], the proposed model uses a similar learning model using Bayesian networks to recommend restaurants to user groups.

Some popular works on model-based approach for group recommendations are discussed in [75, 128, 131]. COM(COnsensus Model) [131] is based on *Latent Dirichlet Allocation*(LDA) model to simulate the generative process of group events. Here, the authors observe that the recommendation decisions are dominated by the users with expert skills in the topics related to the group. Also, the users' behaviour change in the group and content information that affect the group recommendation. The authors incorporate all these different decision points in the model. Two-step Gibb's sampling method is used for the probabilistic parameter estimation, and experiments performed on four different datasets greatly reflect the proposed model's effectiveness.

2.2.6 Group formation in group recommendations

Groups were composed for the first time by combining the concept of group relevance and group disagreement [9]. The groups were delighted with the top-k recommendations. A group recommendation approach proposed in [36] is based on the Bayesian network. This system helps a group of people to decide (like watching a movie). The authors assume that the structure of a group is known apriori. Furthermore, in [41], the authors propose a rule-

based group consensus function that considers different factors among group members. This model uses social information to have a better social descriptor. This descriptor incorporates three factors: dissimilarity factor, social factor, and expertise factor. The consensus function quantifies and merges these factors to a single value and aggregates individual recommendations efficiently.

The model proposed in [83] exploits hierarchical agglomerative clustering to divide all users into clusters; each cluster member contains similar preferences. A collaborative filtering approach is applied over the user clusters for an expert recommendation. In [83], the authors propose the model to recommend items to a group based on the past users' experience, i.e., similar users who liked items in the past are likely to give the same preferences in the present. Boratto et al. [15–17] proposed a set of group recommender systems by using the K-means clustering algorithm that automatically detects groups of users. The introduced predict & cluster model firstly predict the unknown rating using collaborative filtering, and the k-means algorithm is employed to build the user group.

In our lab's work [1], we propose order preference models by introducing the concept of the ordering of items in group recommendations. To establish this concept, variants of aggregated voting and least misery methods are proposed in order to maximize the overall group satisfaction with the generated recommendation vector. The authors propose the hungarian aggregated method and the least misery with priority method by adapting the aggregated voting method and the least-misery method, respectively. Groups were composed based on applying the overlap similarity procedure to all user preferences. Partitions of users having similar total overlap similarity scores merged into a particular cluster. Forming a group using this method is computationally expensive and a cumbersome task from the users' point of view. We introduce flexibility in user preferences by using similar consensus functions of order preferences [1] in [35]. Authors in [85] use a latent factor model (a matrix factorization method for collaborative filtering) for group recommendation. The authors compare this method in three different kinds of group sizes: small, medium and large. A dynamic connection-based group recommendation framework proposed in [94] divides large groups into some subgroups by considering similar contents.

In [107], the authors generate recommendations by considering deviation as a vital part of the aggregation method. They introduce a new variant of the aggregation method that provides group recommendation by considering deviation in users' rating for

an item. In [58], a topic-based model named *GIST* is proposed, which not only considers the individual user's interest but also considers the subgroup's interest in making a recommendation. The connectivity of group member who participates in more subgroups gets a higher weight. Such a member will have more influence in the group. AGREE (Attentive Group REcommEndation) framework in [21] uses an attention mechanism that combines attention network and neural collaborative filtering (NCF) [50] to adopt representation of groups for learning interaction between groups as well as items to improve the group recommendation for cold-start users. In [11], the authors introduce frameworks which can merge trust and similarity among users. The proposed clustering collaborative frameworks are based on fuzzy c-mean clustering and trust-aware. Item-based fuzzy c-means collaborative filtering improves the group recommendation. In [33], the authors use the k-medoid clustering approach to form the group and propose a collaborative filtering approach (which is based on the preferences of the members of the group) to increase the accuracy of the recommendation.

In [1], the authors introduce the concept of the ordering of items in group recommender systems. To justify the importance of order in a group recommendation, the authors introduce different variants of aggregated voting [1, 77] and least misery [1, 77] method to maximize the overall group satisfaction with the generated recommendation vector. The authors form groups using overlap similarity. Since it needs to compute correlations among all users when identifying the group, the group formation technique is computationally inefficient and not preferable. In [34], the authors introduce flexibility in user preferences where the size of the preference list would be less than or equal to group budget[5]. To establish this concept, the authors adopt similar consensus functions of order preferences [1].

In [126], the authors introduce a new aggregation technique based on entropy and evaluate the group recommender systems using the evaluation metric NDCG. Authors also use the same traditional clustering approach, i.e., K-means, to detect an automatically identified group. Afterwards, In [125], the authors also auto-detect the group for recommendation by using bisecting k-means algorithm. Recently, the framework introduced in [105] can efficiently make item recommendations to ephemeral groups. Authors in [7]

[5]Group budget or space budget is the maximum number of items that can be recommended in a group recommendation.

form groups using a conventional clustering approach, i.e., fuzzy-c means(FCM) clustering, and users became potential members of the composed group based on the pearson correlation coefficient measure. Models in [129] analyze the social factors and multi-agent system based negotiation strategy [31] to improve the quality of group recommendations. In [15, 17], the authors proposed a set of group recommender systems by using the K-means clustering algorithm that automatically detects groups of users. The introduced *Predict & Cluster* model firstly predicts the unknown rating using collaborative filtering, and the K-means algorithm is employed over the utility matrix to form a group.

In [126], the authors introduce a new aggregation technique based on the concept of entropy and use *K-means* to detect an automatically identified group. Further, authors in [125] use the bisecting K-means algorithm for automatic group identification. In [105], the authors introduce a model that efficiently makes item recommendations to an ephemeral group. In [113], the authors form the groups based on the similarity shared between the group members using *Pearson Correlation* similarity on the starting samples of the dataset only. In [7], the authors compose groups using fuzzy-c means (FCM) clustering, and the users are potential members of the formed group based on the *Pearson Correlation* coefficient measure. The Recent advancement of hypergraph convolutional neural network [59] also emerges its importance in improving the quality of group recommendations. In [122], the authors introduce a novel method named as *SEAGR* (Social-dual Effect driven Attentive Group Recommendation), which is based on an attention mechanism to locate users in the representative group who holds higher influence instead of less influential members.

Though the above-discussed models consider different categories of groups and employ only traditional clustering approaches to auto-detect the group in the case of an automatically identified group, none of these models deals with the curse of the dimensionality problem. None of the existing models uses locality-sensitive hashing (LSH) [54] for "automatically identified groups". In [70], we have proposed models by taking into account the LSH technique to auto-detect the group and adopt our previous existing models of order [1] and variable-size preferences [35] to produce the group recommendation.

Although existing literature shows that the group formation technique is based on traditional clustering approaches and a dimensionality reduction technique. The proposed group formation model (OPHAencoder) [68] is promising because that deals with the

curse of dimensionality and improves time complexity. The proposed model is also used for auto-detecting groups.

Many existing group recommender systems focus on improving the recommendation quality by learning the representation of users and items and modelling their interaction [22, 23, 53, 105, 119]. However, these approaches pay little attention to comparing users in a group. It is advantageous to compose a group before generating a recommendation in order-based user preference models. Users who rated the same items in a utility matrix become members of an automatically identified group. We assume that users with semantically similar reviews on the same products could be in an auto-detected group. Therefore, each group member would play an equal role in the decision-making process and improve the overall group satisfaction score. In review-based recommender systems, textual reviews of a dataset play a pivotal role. Some review-based recommender systems employ popular deep neural network architectures in traditional recommender systems. The Neural Attentional Regression model with Review-level Explanations (NARRE) [28] considers two parallel CNNs to get a review's contextual semantic representations. Afterwards, an attention mechanism employs to choose critical reviews for modelling user and item representations. In [108], the authors introduce an aspect aware Dual graph convolutional network (Dual GCN) model to learn user preferences and item characteristics at the aspect level, which improves user and item representation. This review-based recommender system provides additional information about user preferences to alleviate the data sparsity problem. In [136], the authors use two parallel convolutional neural networks in the last dense layer for user and item embeddings.

Although existing literature presents the group formation using traditional clustering approaches over the utility matrix, none of the methods considers semantic similarity in the textual information to form the groups. Therefore, our proposed group formation models are promising for auto-detecting groups and improving the quality of group recommendations. The proposed models are based on the order preference model [1]. Also, None of the existing models considers the role of commonly uninterested items in producing the group recommendations.