Artificial Intelligence Research and Development
A. Cortés et al. (Eds.)

doi:10.3233/FAIA220324

Enabling Reproducibility in Group Recommender Systems

Joaquin Dario SILVEIRA [a], Maria SALAMÓ [b,1], and Ludovico BORATTO [c],
[a] *Universitat Politècnica de Catalunya, Barcelona, Spain*
[b] *Dept. Mathematics and Computer Science, University of Barcelona, Barcelona, Spain*
Institute of Complex Systems, University of Barcelona, Barcelona, Spain
[c] *Dept. Mathematics and Computer Science, University of Cagliari, Cagliari, Italy*

Abstract. Reproducibility is a challenging aspect that considerably affects the quality of most scientific papers. To deal with this, many open frameworks allow to build, test, and benchmark recommender systems for single users. Group recommender systems involve additional tasks w.r.t. those for single users, such as the identification of the groups, or their modeling. While this clearly amplifies the possible reproducibility issues, to date, no framework to benchmark group recommender systems exists. In this work, we enable reproducibility in group recommender systems by extending the LibRec library, which stands out as one of the richest, with more than 70 different recommender algorithms, good performance and several evaluation metrics. Specifically, we include several approaches for all the stages of group recommender systems: group formation, group modeling strategies, and evaluation. To validate our framework, we consider a use-case that compares several group building, recommendation, and group modeling approaches.

Keywords. Group Recommender Systems, Reproducibility, Algorithms

1. Introduction

Enabling reproducibility should be of paramount importance inside the research community [1]. In fact, it is hard to determine the speed of progress, or even if we are making any, when so much of the newly generated knowledge is not reproducible [2]. The existence of base libraries with known and well studied approaches and algorithms is one of the first steps in any field that seeks to advance on firm knowledge. Moreover, it is in fields in which such frameworks are missing that it is hardest to justify new ideas by benchmarking them against the existing literature.

Recommender systems (RSs) support users by suggesting items that might be of interest to them [3]. This is usually done by learning behavioral patterns from historical data, usually in the form of user-item interactions. Nearly any popular programming language has a library or framework for making single recommendations. Despite the amount of papers regarding the problem of generating recommendations for groups of users (*group recommender systems*, GRSs) [4,5], a firm ground for GRSs does not exist.

[1]Corresponding Author: Department of Mathematics and Computer Science, University of Barcelona, Barcelona, Spain; E-mail: maria.salamo@ub.edu

This is exacerbated by the difficulty of accessing to or generating datasets that gather information of actual group recommender systems [6,7,8,9]. Thus, in the RS research field, a common framework for benchmarking GRSs is a known open issue.

In contrast to single RSs, in a framework for GRSs several issues appear, mainly because a GRS provides suggestions in contexts in which more than one person is involved in the recommendation process and their aim is to provide recommendations to the whole group, considering the preferences and the characteristics of more than one user. Because of this, a great amount of researchers resort to individual recommendation datasets for offline testing and benchmarking [10,11,12,13]. As a consequence, this introduces an important issue regarding the need to *form groups* to whom propose group recommendations. In addition, it would certainly be necessary to address the issue of how to *evaluate the results for groups*. Every group recommendation study seems to tackle these questions differently. Due to these two issues and other factors, there is a whole myriad of ways in which group recommendations can be performed. Furthermore, the development of new strategies and ongoing research in several of these issues makes the task of encompassing all of them in a single framework daunting. For this reason, enabling reproducibility in group recommender systems is of central importance.

In this paper, we enable reproducibility in group recommender systems by extending the LibRec (i.e., www.librec.net) library, which is one of the most widely used recommendation frameworks. The proposed extension encompasses different interpretations of the aforementioned issues. In particular, we focus on several of the stages of group recommender systems: group formation, group modeling, and evaluation. To narrow down the scope of this paper, we tackle the following aspects: (1) *Building of synthetic groups*, where we focus on offline group recommendation with synthetic group formation, due to the lack of real group recommendation datasets; (2) *Single user prediction aggregation*, where we implement the most common approaches of aggregating user preferences [14]; (3) *Measuring members satisfaction with the group recommendation*, where we compare the individual preferences expressed in the test set with the group recommendations.

Our contributions are summarized as follows: **(1)** We propose a framework[2] to enable reproducibility in group recommender systems; **(2)** We elevate some reproducibility questions often not regarded or ignored, which are relevant for understanding and comparing group recommendation experiments; **(3)** In the scope of collaborative filtering approaches, we enable the use of several combinations in three main stages (group building, group modeling, and evaluation) of group recommender systems; and **(4)** We present a use-case that compares group building, recommendation, and group modeling approaches, both in terms of effectiveness and efficiency.

2. Framework for Benchmarking Group Recommender Systems

2.1. Introduction to the group recommendation pipeline

The group recommendation framework, called GroupLibRec, is an extension of the LibRec library. We have chosen LibRec as base library because it is under the GPL license, it has already implemented more than 70 recommendation algorithms, has good performance and is widely used in the recommender system research community. Note

[2] https://github.com/panserbjorn/librec/tree/3.0.0/RecSys

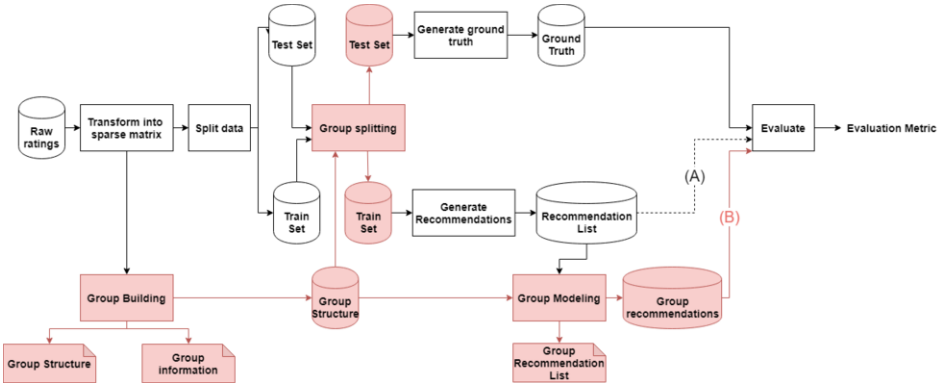


Figure 1. GroupLibRec recommendation pipeline. In red the extensions to the data flow pipeline made to LibRec to enable group recommendations.

that the usage of a base library for building the framework permits us to reuse existing and tested solutions to the data processing and single users recommendation scenarios. This is highly desirable, since most group recommender systems include in their process single user recommendations. The library extension could be pursued in different ways, i.e., as an external project, as another project inside the library or integrating the group recommendations to the core library. We have opted for the last one, mainly because our goal has been to include the group recommendation capabilities with as few changes as possible to LibRec. We consider it will increase the chances of the group capabilities being introduced to stable future versions of the library.

Figure 1 depicts the recommendation pipeline for GroupLibRec, which includes the LibRec pipeline (elements in white color). Note that the LibRec library pipeline performs the steps of data ingestion, splitting it into train and test sets, training a recommender, generating recommendations and, finally, evaluating the results. Configuration files enable the setup of all the required parameters in a run, such as the recommendation algorithm selection, the dataset location, and the evaluation metrics. Thus, this process facilitates the reproducibility of experiments just by sharing and exchanging a single configuration file.

The red components in Figure 1 show the extension of the pipeline for enabling group recommendations. The additional steps are: *group building*, *group modeling*, and *splitting*. Specifically, the Group Recommender delegates the responsibility of generating the individual recommendations in the pipeline to the individual user recommender algorithm and then it builds the recommendations for the groups. Specific additions to the recommendation process for enabling group recommendations will be discussed in the following sections.

2.2. *Group Building*

Due to the lack in datasets that contain natural groups information, most of the research work on group recommendations starts with the formation of synthetic groups inside datasets of individual users ratings. We consider three aspects associated to synthetic groups: (*i*) if they are overlapping, (*ii*) if they use internal or external information, and (*iii*) if the groups cover the entire user base.

First, in relation to the overlaps, *groups that do not overlap* could be stable or occasional [15], e.g., family, colleagues, or close friends. These groups gather to share a common experience and recommendations about such experiences might be sought. This is the case of research studies such as [14] and [16] that propose a system for generating group recommendations regarding trips, or [13] that proposes group recommendation in the context of movies. In these groups, we assume that the similarity between users is high, either because they are gathered together due to those similarities or because their interactions as a group shape their responses to certain items. For the study of this kind of groups, we implemented two approaches, capable of detecting them in individual user datasets, namely *similar users group identification* and a *k-means clustering* technique. On the other extreme, *groups that have big overlaps* could be non established or random groups such as people gathered in the same physical place for an event like a club [15]. Members in these groups might not share much in common and they do not necessarily present a defined inner structure for deciding between different options [17]. The same users that are present in a group might behave differently in another group and therefore could be considered as a different user altogether [7]. That is why we believe that the study of these groups is the most interesting, not because of the overlapping nature, but mainly because of the diversity of the members inside each group. To generate diverse groups we have included a *random group generation* strategy, described below.

Second, considering the usage of *internal or external information*, most of the state-of-the-art approaches use internal information, such as rating matrix. On the other hand, other group strategies use external information such as demographics or social network information [7] for building groups. Currently, the framework focuses on building groups using internal information and similarities between users. The usage of external information will be addressed in the future work.

Finally, the last aspect we consider for group building is the *coverage of users when groups are formed*. The coverage determines if every user inside the dataset belongs to a group or not. The coverage does not rise when the origin of the data has already groups built in, because the assumption of groups being analyzed excludes single users that do not fit into any group. However, when generating synthetic groups it gains more relevance. This arises the problem of what do do when users do not belong to any group, by either removing or maintaining them in the dataset. Our decision was to maintain them since their ratings may contribute to the collaborative filtering process. Thus, we decided to allow formation of groups for a dataset that did not cover all users. We will return on the coverage aspect when discussing the evaluation and splitting.

Overall, the three strategies (i.e, k-means, similar users, and random) in our framework produce non-overlapping groups, which reduces the number of possible groups, but simplifies the evaluation of the group recommendations. To reproduce experiments, both the assignation of users to groups and group information are stored in files. Next, we detail the group building strategies.

K-means based group builder. This strategy assigns a group to every user based on the K-means algorithm applied over the rating matrix. However, since the rating matrix contains sparse data, the Euclidean distance between the centroid and the users is computed only in the common items. For the non shared items, the maximum distance is used. This maximum distance depends on the rating scale present in the dataset. An interesting aspect of the K-means approach is that it can be used to identify a certain number of target groups, independently of their size. Note that the group size depends on the content of

the rating matrix. In contrast, the two approaches we present in what follows are defined based on the size of the groups. These approaches are more useful to study scenarios that are meant to be used by well defined groups.

Random group builder. The random group builder assigns each user to a group randomly, until the desired group size is reached.

Similar users group builder. This strategy is inspired on the work described by Baltrunas et al. [11], which discussed a group formation of any size with the use of the Pearson Correlation Coefficient (PCC) as a measure of similarity. Basically, they define that groups should have a member-to-member high correlation in their preferences. Thus, group members must have a PCC higher than a specified threshold with all other members of the group. However, finding these groups was not an easy task. By its very nature the method cannot assign a group to every user, since not all users in the dataset necessarily have $n-1$ other users they are similar enough with. As a result, it does not generate overlapping groups and does not cover all users in the dataset.

Our approach to deal with this problem is as follows. We maintain a list of available users (i.e., users that do not belong to any group). Then, the strategy retrieves for each user the other similar users, which are still available, sorted by similarity. An initial group is formed with the firstly selected user as the center. New possible group members are verified for compatibility with the current group. If they have a PCC higher than the predefined threshold with all current members, they are added to the group and the next user is verified. This is performed until the specified group size is reached or all available users have been verified. Once the group reaches the desired size, all its members were removed from the available user list and the process continues with the next available user. If the group do not reach the desired size with the first user as its center, the method will continue with the next available user as center. This method generates groups surrounding an initial user as center and seeks in a greedy fashion the closest group w.r.t. the center. The correlation threshold can be adjusted to generate more or less groups up to a certain point, since groups that have correlations close to 0 are not considered "similar".

2.3. Group Modeling

Group modeling refers to how individual preferences can be combined to express the group preference, either to rate an item or to generate a ranking. The strategies included in the framework are: *Additive Utilitarian*, *Most Pleasure*, *Least Misery*, *Multiplicative Utilitarian*, *Borda Count*, *Approval Voting*, *Average Without Misery*, *Fairness*, *Plurality Voting* and *Copeland Rule*. All implemented as detailed in [4], except for Borda Count.

The Borda Count strategy expects that the whole group expresses a rating for all the items that are being considered for the group. This was not always the case in the framework since every member of the groups may have different items for which it is being tested and therefore recommendations for these items are predicted but not for others. To solve this issue, we use a *Partial Voting Borda Count* strategy [18] in which the value of the votes expressed by each member depends on the number of votes and in the order they form. Note that those strategies that produce a rating value can be applied both for ranking and rating recommendation, whereas those that generate ranking cannot be directly mapped to rating predictions in groups. For rating recommendations, both the predicted and expressed preferences were used for the modeling, assuming the

recommendations do not have to be novel for the entire group. However, for ranking strategies, the individual ratings could not always be considered for the formation of the group ranking, since a direct matching between the rating and a ranking for that user was not possible.

2.4. Splitting

The splitting of rating datasets can be performed in different ways. The most popular strategies include a percentage of the ratings expressed, a percentage of the items rated by each user, a percentage of the ratings for each item, a Leave-One-Out or Hold-On, and Folds strategies. All these strategies are currently offered by LibRec and by our extension. In Figure 1, the group recommendation pipeline includes the split of data, which comes from LibRec, and later a group splitting. This is necessary because in group recommender systems additional considerations should be taken into account: 1) *Should the split be independent of the group structures?*, 2) *What means a percentage of the items "rated by the group"?*. These questions remain unclear in the literature.

Regarding the *first question*, we wanted to allow splitting strategies to be either dependent or independent of the groups. Thus, we introduced a new splitting dependent on the groups and maintained the possibility of using the splittings already existing in LibRec, which are independent of the group structures. After the splitting, it is necessary to review test samples. When the group building strategy covers all users in the dataset, the test set is not reviewed. In contrast, others may not cover all users, so the users that do not belong to any group will not get recommendations. In this case, the test set is fixed. That is, test samples that are of users that do not belong to any group are moved back to the train set. By doing this, all users in the test set get group recommendations and can be evaluated. Even though this changes slightly the number of test samples, it only affects group building strategies that do not cover all users.

In relation to our *second question*, Najjar and Wilson [19] defined a group splitting strategy that mimics the rating percentage of users but for groups. We have included it, with a relaxation on the constraints. Instead of considering only items rated by all members in a group, which would not be viable in most datasets, we consider as candidate test item any item rated by a member of a group. All ratings of group members that belong to the excluded items are then moved to the test set. This manner of splitting can be used to test recommender systems for groups that have the restriction of exclusively novel recommendations. This can be applied in situations in which the usage of an item by any member of a group prevents that item for being considered by the group.

2.5. Effectiveness Evaluation for Groups

In order to evaluate the effectiveness for groups of users, we adopted an approach that compares the recommendations generated for the groups against the expressed preferences by each member in the test dataset. Specifically, when considering a rating prediction setting, we compare the ratings of each user in the test dataset against the ratings predicted for the group. In case a ranking is generated, we evaluate the utility of the ranking generated for the group using the information of the test set of the user.

3. Analysis of the GroupLibRec framework

3.1. Setup

For evaluating the framework, we study the performance in *rating* and *ranking* recommendations for groups in the MovieLens1M dataset, which contains 1M ratings, given by 6000 users to 4000 movies. The specific setup of the framework is as follows.

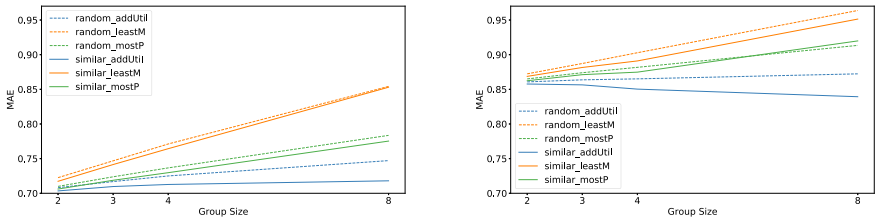
Groups were built using the *random* and the *similar group building* approaches with group sizes of 2, 3, 4, and 8. As group modeling strategies, we considered *Additive Utilitarian*, *Least Misery*, and *Most Pleasure* for the rating experiments, and all the modeling techniques for ranking experiments. The splitting between train and test was performed using the *ratio percentage splitter* included in the LibRec library with 80% of ratings for training and 20% of ratings in the dataset for testing. Groups were generated and stored previously to recommendations predictions. The number of groups generated depended on the size of the groups being tested and the approach. For the *similar group building* strategy 0.27 was used as threshold for the similarity between members.

In addition, we have chosen three recommendation algorithms for individual user predictions, namely *UserKnn* (User K-Nearest Neighbor) and *BiasedMF* (Biased Matrix Factorization) for rating and ranking prediction, and *BPR* (Bayesian Personalized Ranking [20]) solely for ranking. The *BiasedMF* recommender was used with a learning rate of 0.01 and 20 factors. The *UserKnn* recommender used 20 nearest neighbors and PCC as similarity measure. *BPR* was used with a learning rate of 0.1, 10 factors and a decay of 1.0. We use two metrics for the evaluation, MAE (*Mean Absolute Error*) is used for the rating prediction results and NDCG (*Normalized Discounted Cumulative Gain*) for the ranking predictions. For the ranking experiments, we generated a top-20 for each group.

3.2. Analysis of Results

In this section we analyze the effectiveness of our framework.

Rating prediction. We begin by comparing the effectiveness of each group modeling strategy in the rating prediction task. Figure 2 depicts the results for the different prediction algorithms and group building strategies, in relation to the group modeling strategies analyzed. It is important to highlight that in all cases the *BiasedMF* algorithm outperformed *UserKnn*. Moreover, the performance of the recommendation algorithms was slightly better in the similar groups than in the random ones (see dotted lines). Baltrunas et al. [11] did a similar comparison of group building strategies with Movielens 100k, with results comparable to ours. Indeed, they established 0.27 as threshold for user similarity, because of the distribution among the similarity pairs, and we verified that in Movielens 1M it was the same. In addition, in Figure 2 we compared the rating performance of Additive Utilitarian, Least Misery, and Most Pleasure with respect to the group size. Our results denote that Additive Utilitarian performed the best in spite of the base recommendation algorithm. Another remarkable observation is that independently of the group building strategy, all algorithm's performance decreased as group size increases. Boratto and Carta [21] also noted this effect. The only exception to this being the *Additive Utilitarian* model with the *UserKNN* recommender in the similar groups, which maintained itself almost stable. Indeed, this combination improves with the size of the groups because the bigger the groups are, the more users end up forming part of the



(a) Biased Matrix Factorization (b) User K-Nearest Neighbors

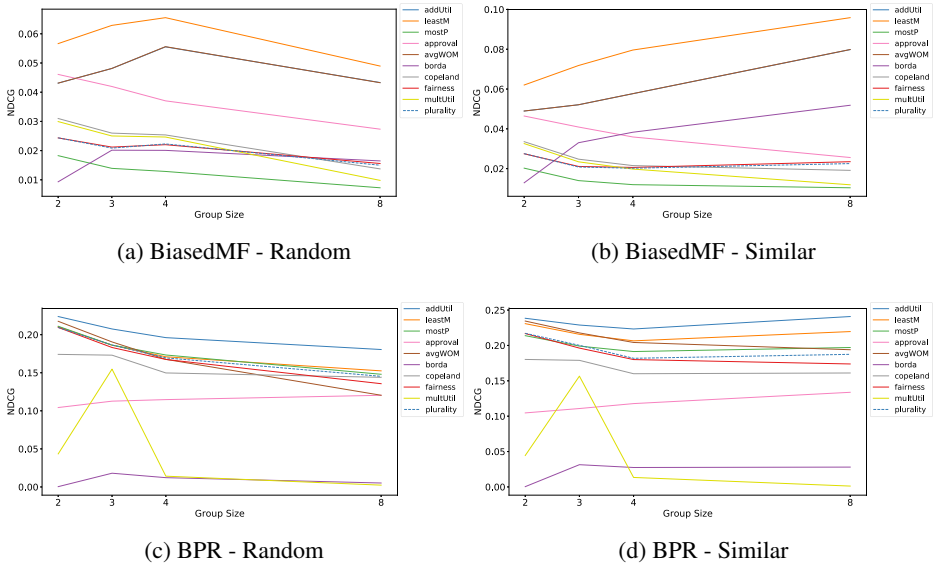
Figure 2. Rating performance comparison by base recommender algorithm. Base single user recommendation algorithm performance in groups of size 2, 3, 4 and 8 for random and similar group building strategies. Performance is being measured with the Mean Absolute Error metric.

neighbors considered for the recommendations. Despite that, the value of the MAE for this configuration is still higher than that of *BiasedMF*.

Ranking generation. In Figure 3 we first analyzed the performance of all group models for each recommendation algorithm separately. Similarly to the rating scenario, the results show that in the random group formation the performance decreases with the size of the groups. This seems an expected behavior since the bigger the group, the less probable it is that the right items for each user will be present in the top 20. However in the similar groups, in some situations, the performance minimally improves or is stable with respect to the group size. This may be due to the fact that similar groups are formed based on their expressed ratings and the bigger the groups, the more items in common all members have and thus increasing the chances of them being present in the group top 20 items.

Execution time. It is also an important factor for a framework, we have compared the time required to execute each modeling strategy in the approaches optimized for the ranking. This choice was made because rating prediction strategies are few in our comparison and they required only a few seconds in the MovieLens 1M dataset, while ranking recommendations take more time due to the number of items being considered for each member in the group ranking process. It is important to highlight that all the strategies obtained good results (from 12 to 50 seconds), with the exception of Copeland Rule. This is largely due to the complexity of this strategy, which needs to consider all the pairwise victories of items in the rankings of each group member. Such number of combinations explodes with the number of items and the number of group members. As a result, the execution of Copeland Rule takes approximately one hour for each experiment, whereas the remaining strategies spend less than a minute.

Discussion. The first aspect that comes out of our evaluation is that we have proved again some of the assumptions already expressed in previous papers regarding group recommendations and group sizes. Indeed, in Figs. 2 and 3, the results show that the smaller the group size, the better the performance. Additionally, we have also shown how the framework allows for reproducibility and quick testing of a great variety of group recommendation algorithms. An important aspect in a framework is the execution time. With the exception of the Copeland Rule group modeling strategy, the results clearly show that the remaining strategies are under reasonable limits, allowing the usage of the framework in small and medium sized datasets. One great advantage of our way of extending the LibRec library is that previous users can simply change configuration files and execute



(a) BiasedMF - Random (b) BiasedMF - Similar

(c) BPR - Random (d) BPR - Similar

Figure 3. Ranking performance comparison between group modeling strategies. (a) and (c) show the performance in the random group building strategy for the BiasedMF and BPR. (b) and (d) show the same algorithms in the similar group building approach. Performance is measured with NDCG in the top20 rank.

the same experiments used in individual recommendations but for group recommendations. Finally, regarding the limitations of our work, despite allowing for a large number of combinations of strategies, general architectures for building the recommendations other than aggregating individual user predictions are not covered by the framework. For example, the usage of the groups structure for building the individual member recommendations before aggregating them into the group recommendations (such as [19] for memory-based group recommendations). Our expectation is to cover this in the future.

4. Conclusions and Future Work

In this paper, we tackled the issue of enabling reproducibility in group recommender systems due to the lack of frameworks in the field to build, test and benchmark this type of recommenders. We have proposed GroupLibrec, a framework on the scope of collaborative filtering algorithm, that concentrates on three of the main stages of a group recommender system: *group formation*, *group modeling*, and *evaluation*. In any of these three stages, we have integrated several strategies in order to enable the reproducibility of the most used strategies in the field. Indeed, we have shown how the structure of the framework allows for a great number of combinations of base recommendation algorithms, group modeling strategies, and group building strategies. The proposed framework can be easily used for reproducing, testing and helping with the development of new group recommendation systems.

As future work, we plan to include even more features, such as the ability to use other group recommendation architectures or the use of external information in the group building stage.

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 860843.

References

[1] Beel J, Breitinger C, Langer S, Lommatzsch A, Gipp B. Towards Reproducibility in Recommender-Systems Research. User Modeling and User-Adapted Interaction. 2016 Mar;26(1):69–101. Available from: https://doi.org/10.1007/s11257-016-9174-x. doi:10.1007/s11257-016-9174-x.

[2] Ferrari Dacrema M, Cremonesi P, Jannach D. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In: Proc. of the 13th ACM Conference on Recommender Systems; 2019. p. 101–109. doi:10.1145/3298689.3347058.

[3] Ricci F, Rokach L, Shapira B. Recommender Systems Handbook. 3rd ed. Springer; 2022.

[4] Boratto L, Felfernig A. Group Recommendations. In: Collaborative Recommendations - Algorithms, Practical Challenges and Applications. WorldScientific; 2018. p. 203–232. doi:10.1142/9789813275355_0006.

[5] Jameson A, Willemsen MC, Felfernig A. In: Individual and Group Decision Making and Recommender Systems. Springer US; 2022. p. 789–832. doi:10.1007/978-1-0716-2197-4_21.

[6] Garcia I, Sebastia L, Onaindia E, Guzman C. A Group Recommender System for Tourist Activities. In: Proc. 10th Int. Conf. on E-Commerce and Web Technologies,; 2009. p. 26–37. doi:10.1007/978-3-642-03964-5_4.

[7] Sánchez LQ, Recio-García JA, Díaz-Agudo B, Jiménez-Díaz G. Social factors in group recommender systems. ACM Trans Intell Syst Technol. 2013;4(1):8:1–8:30. doi:10.1145/2414425.2414433.

[8] Contreras D, Salamó M, Pascual J. A Web-Based Environment to Support Online and Collaborative Group Recommendation Scenarios. Applied Artificial Intelligence. 2015;29(5):480–499. doi:10.1080/08839514.2015.1026661.

[9] Contreras D, Salamó M, Boratto L. Integrating Collaboration and Leadership in Conversational Group Recommender Systems. ACM Trans Inf Syst. 2021;39(4). Available from: https://doi.org/10.1145/3462759.

[10] Amer-Yahia S, Roy SB, Chawla A, Das G, Yu C. Group Recommendation: Semantics and Efficiency. Proceedings of the VLDB Endowment. 2009;2(1):754–765.

[11] Baltrunas L, Makcinskas T, Ricci F. Group Recommendations with Rank Aggregation and Collaborative Filtering. In: Proc. 4th ACM Conference on Recommender Systems. RecSys '10. Association for Computing Machinery; 2010. p. 119–126. doi:10.1145/1864708.1864733.

[12] Salamó M, Mccarthy K, Smyth B. Generating Recommendations for Consensus Negotiation in Group Personalization Services. Personal Ubiquitous Computing. 2012 Jun;16(5):597–610.

[13] Recio-García JA, Jiménez-Díaz G, Sánchez-Ruiz-Granados AA, Díaz-Agudo B. Personality aware recommendations to groups. In: Proc. ACM Conference on Recommender Systems. ACM; 2009. p. 325–328. Available from: https://doi.acm.org/10.1145/1639714.1639779.

[14] Jameson A, Smyth B. Recommendation to groups. In: Brusilovsky P, Kobsa A, Nejdl W, editors. The adaptive web. Berlin, Heidelberg: Springer-Verlag; 2007. p. 596–627.

[15] Boratto L, Carta S. State-of-the-Art in Group Recommendation and New Approaches for Automatic Identification of Groups. In: Information Retrieval and Mining in Distributed Environments; 2011. p. 1–20. doi:10.1007/978-3-642-16089-9_1.

[16] McCarthy K, Salamó M, Coyle L, McGinty L, Nixon BSP. CATS: A Synchronous Approach to Collaborative Group Recommendation. In: Proc. of the FLAIRS 2006 Conf. Springer; 2006. p. 1–16. Florida.

[17] Masthoff J, Delić A. In: Group Recommender Systems: Beyond Preference Aggregation. New York, NY: Springer US; 2022. p. 381–420. doi:10.1007/978-1-0716-2197-4_10.

[18] Koffi C. Exploring a generalized partial Borda count voting system. Senior Projects. 2015.

[19] Najjar NA, Wilson DC. Differential Neighborhood Selection In Memory-Based Group Recommender Systems. In: Proc. 27th FLAIRS Conf.; 2014. p. 69–74.

[20] Rendle S, Freudenthaler C, Gantner Z, Schmidt-Thieme L. BPR: Bayesian Personalized Ranking from Implicit Feedback. In: Proc. 25th Conf. on Uncertainty in Artificial Intelligence; 2009. p. 452–461.

[21] Boratto L, Carta S. ART: group recommendation approaches for automatically detected groups. Int J Mach Learn Cybern. 2015;6(6):953–980. doi:10.1007/s13042-015-0371-4.