Towards creating a non-synthetic group recommendation dataset

Matthijs Rijlaarsdam
Delft University of Technology
Delft, The Netherlands
mscrijlaarsdam@gmail.com

Sebastiaan Scholten
Delft University of Technology
Delft, The Netherlands
j.s.m.scholten@tudelft.nl

Cynthia C. S. Liem
Delft University of Technology
Delft, The Netherlands
C.C.S.Liem@tudelft.nl

ABSTRACT

Recommender systems can be useful in group settings, e.g. when choosing a movie to watch with a group. However, while considerable research in group recommendation has been performed, we still lack truly ecological datasets on group recommendations in real life consumption scenarios. Much of the existing work considers hypothetical consumption scenarios, and commonly, individual ratings are aggregated, but no actual group consumption takes place in which situational differences per group are taken into account. In this paper, we outline a vision for acquiring more realistic and ecological group consumption data, based on a crowdsourcing application that will acquire individual ratings per group consumption event. We discuss various design decisions that will allow us to gather these ratings effectively from a large group of people, and demonstrate and evaluate the viability of our approach towards reaching group consensus through rating session simulations.

CCS CONCEPTS

• **Information systems** → *Test collections*; Recommender systems.

KEYWORDS

Datasets, Group Recommendation, Crowd Sourcing

1 INTRODUCTION

Recommender systems are often tailored to the individual. However, in real life, many of the scenarios relevant to recommender systems actually occur in group contexts. At different moments, an individual user may be a part of different groups with different preferences. In such cases, not only the individual's preference should be taken into account, but also the preferences at the group level, at that particular moment, and given that particular group composition.

Currently, group recommender systems are mostly based on aggregation of information in individual user profiles, such as [12]. Various aggregation strategies, rooted in theories on group decision-making processes, exist for this [9]. Major challenges for group recommender systems involve visualizing the acceptability of a recommendation and choosing the correct preference aggregation [8].

While group recommendation is actively studied, it should be noted that no public group recommender research dataset exists. The past few years, evaluation of group recommendation systems has typically been done offline, through metrics on aggregated individual rating datasets [5] [1] [4]. Some existing works consider surveys or user assessments of hypothetical consumption scenarios, such as [11]; in other cases, such as PolyLens [10], real-life user logs were additionally studied, but the corresponding data has not been released to the community, making it hard to truly assess group-based dynamics. Similarly, studies into group recommendation strategies have been done using a dataset which uses recipe ratings provided by families of users, however this dataset likely is too small to fully capture these intra-group effects [2], while also not having been publicly disclosed [3].

Being interested in acquiring a more ecological dataset of group consumption and recommendation, in this paper, we outline a vision to acquire such a dataset through a crowdsourcing application, focusing on the problem of group recommendation for movies. Similar to common user feedback strategies in dating apps, we propose to acquire fast user feedback by collecting swipes on movie likes and dislikes through a mobile interface, and consider approval voting strategies, in which common consensus should be reached by the group on what to watch. Through a simulation study based on the Movielens 100k dataset [6], we assess the feasibility of reaching common consensus as quickly as possible for different group sizes, investigate the ratio of unrated movies suggested to reach consensus, and the necessary ratio of agreement needed within the group. With our results showing that reaching common consensus within several interactions is feasible, it will make sense to implement this crowdsourcing mechanism in real life and integrate it into group watching scenarios, thus acquiring more ecological data that can help us in better assessing and understanding group recommendation mechanisms and their impact on user preferences in real life.

2 APPLICATION SCENARIO

We choose to focus on a group recommender scenario, in which several users convene as a group, and wish to watch a movie together. Considering that a movie typically takes several hours to watch, we assume that the group jointly needs to decide on a single movie to watch. As a consequence, it makes sense to assume an approval voting strategy, in which a choice needs to be made that satisfies everyone.

Rather than having the group discussing about this, we envision the use of an app as illustrated in Figure 1. Within this app, users are offered various movies to watch, and they can indicate their desire to currently watch this movie with a swipe, indicating a binary 'like' or 'dislike'. This way, users can give quick feedback on many possible movies.

To elicit the preferences of users in the current situational group setting, movies will firstly be advised randomly, until sufficient

RecSys '19, Sept 16–20, 2019, Copenhagen, DK

Figure 1: Mockup of a movie preference elicitation app, with a swiping interface to indicate the desire to watch.

information is gathered to make informed recommendations based on the different current individual user preference profiles.

After rating these random movies, the users will enter a new rating round, in which they partially will receive movies from a random subset of the set of movies that have not been rated by any user in that session, and partially movies from the set of movies that *have* been rated by another user in that session. From these sets, the movies that the user will like most according to the recommendation model are actually suggested. These rounds will continue until the users agree on a movie. Once all the users have reached agreement up to a certain threshold, the final recommendation will be decided and displayed to the group.

3 SIMULATION

Before developing and deploying the app as proposed in the previous section, it is important to first understand whether the proposed underlying mechanisms would indeed make sense. In particular, we are interested in investigating whether approval voting through the proposed swiping mechanism would allow for group consensus to be reached within reasonable time limits. More specifically, we focus on three main questions:

(1) How does the group size affect the amount of interaction needed until consensus?
(2) How does the ratio of unrated movies affect the amount of interaction needed until consensus?
(3) How much agreement should be required within the group, in order to be able to reach consensus?

To study these questions, we perform a simulation study, based on the Movielens 100k dataset. The implementation of our simulator was done in Python, making use of the well-documented Scikit Surprise [7] toolbox. The program is optimized with efficient data structures and multiprocessing, allowing for thousands of simulated sessions per hour on a laptop. The code can be made available on request.

In Figure 2, a schematic overview of our simulation session setup can be seen. Our simulation consists of two sides: a group recommender, trained on the data of 80% of the users in the Movielens 100k dataset, and a user simulator, trained on all Movielens data. The 20% of the users who were not considered in the group recommender form the test set, from which groups are formed, for which the group recommender should provide the right recommendations.

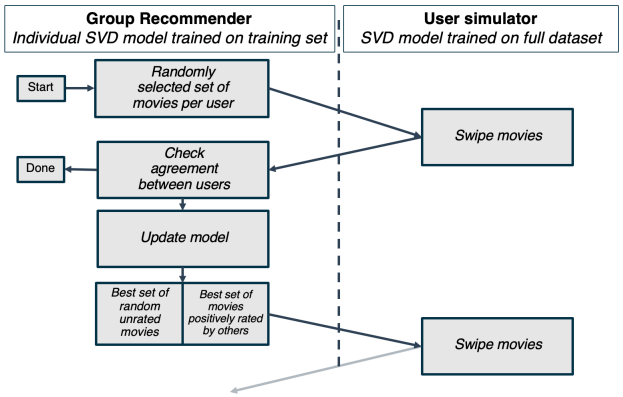

Figure 2: Schematic view of our application flow

Globally spoken, at the start of a session, a random set of users from the test set is selected to form a group. It is assumed that these users did not see any of the movies known by the group recommender, and the group recommender does not know the preferences of these users. The group recommender therefore first sends a configurable number of random movies to each user to be rated, as discussed in more detail below.

After this first round of ratings, it is highly unlikely that consensus will be reached. Therefore, more rating rounds are needed. Based on the responses of individual users on the initial random movie set, the group recommender will therefore generate new sets of movies to be rated per user. Each of these sets will comprise a personalized mix of unseen movies that are likely to be liked by the given user, and movies that have been liked by other users in the group. This process repeats until consensus is reached.

The Movielens 100k dataset has 5-point Likert ratings, whereas our proposed app uses binary relevance levels. For our current study, we assume a user explicitly likes a movie if the predicted rating for the movie is at least 4 out of 5.

3.1 User simulator

For a given movie and a given user, the user simulator should predict how a user would rate this movie. For this, we employ an SVD model trained on the full Movielens 100k dataset. Using grid-search, we optimize the various parameters of our model such as the learning rate and regularization parameters. Our final model achieves a final RMSE of 0.8706, 5-fold cross-validated on the Movielens 100k dataset.

We want the user simulator to predict ratings as accurately as possible. If a ground-truth rating from the original Movielens 100k dataset is available for a certain (user, movie) pair, the user simulator returns that as its rating, as opposed to the SVD prediction. As the user simulator is a separate system from the system we are trying to test, we are able to do this without violating the assumption that all users have not watched any movie; the ground-truth rating is simply a more accurate description of the user's preference.

3.2 Group recommendation model

During a session, movies are shown to users for them to rate through a swipe. How the users rate these movies reflects their preference profile. Therefore, after a round of swiping, the group recommendation model should be updated with information resulting from these swipes, before generating new movies to swipe in the next round.

Because we need to update our recommendation model between swipe rounds, having to fully train the model every round becomes unfeasible. For this reason, a matrix-based SVD model is used for the recommendation model, as well as for the user simulator. A rating consists of the mean μ, the bias for the user b_u, the bias for the item b_i, and the matrix product of the item and user factor matrices q and p.

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u$$

By using such a model, we are able to do partial training for a user. We do this by only updating the biases for the users and movies for whom ratings are added, as well as the user factors p_u corresponding to newly added users.

4 RESULTS

Following our three main questions as presented in Section 3, we present our results in this section. Each plotted data point reflects the averaged result of 100 simulation sessions.

4.1 Amount of interaction needed for different group sizes

For this experiment, we want to have the smallest possible granularity in the amount of swipes needed per user. As a session can only end after a round, the amount of movies sent to a user per swipe round should be as small as possible. Because we need to send at least one movie that is rated by others in the same group, and one unrated by users in the group, we send 2 movies to each user per swipe round in this experiment, thus having a rated/unrated ratio of 0.5. For this first experiment, we set the required agreement ratio at 1: a movie is only selected if everyone in the group approves this movie. However, it may happen that a group may not actually reach consensus, even after many swipes. To assess this in more detail, we cap the maximum amount of ratings to 500 per user, and consider a run to be failed in case no positive consensus is reached yet at this capping moment.

For all succeeded runs in this experiment, the average amount of swipes needed before reaching positive consensus for different group sizes is shown in Figure 3.

From these results, it can be seen that on average, users find agreement on movies quite quickly, regardless of group size. Failed run ratios for different group sizes are shown in Figure 4.

As can be seen in Figure 4, mostly the small groups seem to fail, whereas all groups larger than 11 find agreement. This seems counter-intuitive, as for larger groups it should be harder to find a movie everyone agrees with. This suggests that the problem for smaller groups is not that a movie that pleases everyone does not exist, but that this movie simply is not found. Since all users in a group swipe different unrated movies, the search space of swiped movies is larger for larger groups. They are therefore more likely to find a movie that everyone in the group is willing to watch.

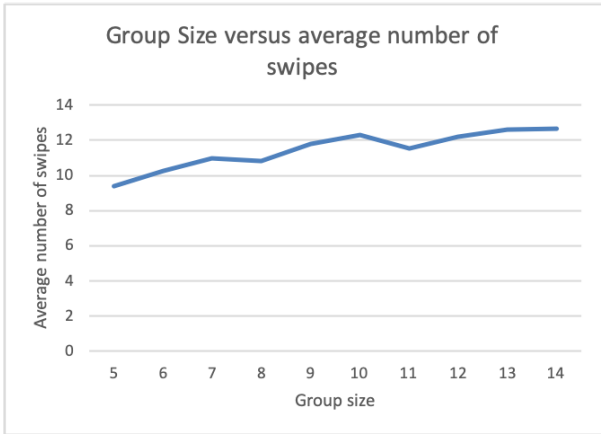

Figure 3: Average amount of swipes vs. group size for successful runs

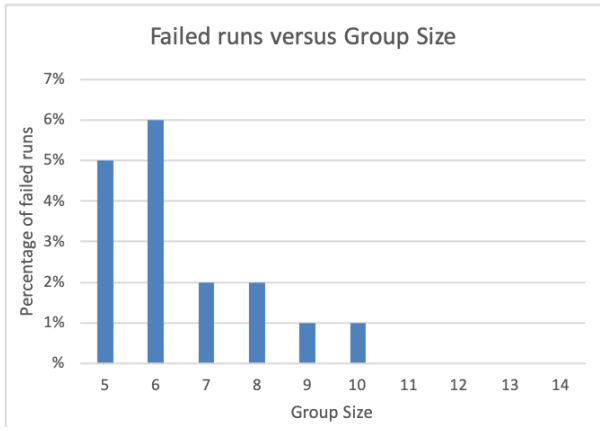

Figure 4: Failed runs vs. group size

4.2 Effect of different ratios of unrated movies

As discussed in Section 3, after an initial round of rating random movies, users will get a mix of unrated movies they may like, and movies that other people in the group liked. To investigate how this mix should be balanced, for our second experiment, we consider how this balancing should be done. We consider group sizes ranging from 5 to 14, and require for users to reach perfect agreement on a movie. That is, a movie should be found that all users rate positively. To limit our simulation time, we send 10 movies to each user per round and limit a session to a maximum of 50 swipes per user. As in the previous experiment, we focus on how many failed runs occur. Results are plotted in Figure 5.

From the plot, it can be seen that, as in the previous experiment, larger groups have less failed sessions than smaller groups. Furthermore, with regard to the ratio of unrated movies within the

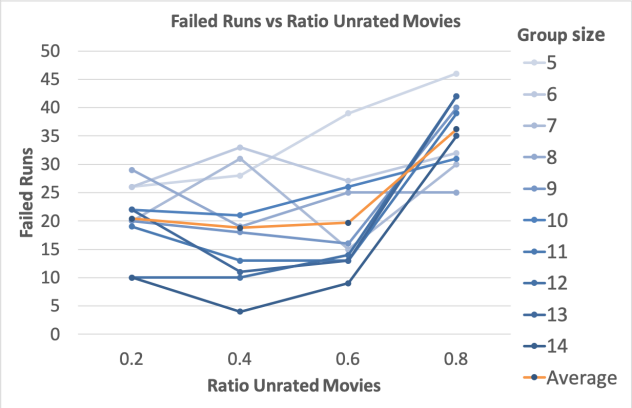

Figure 5: Failed runs vs. Ratio unrated movies

Figure 6: Failed runs vs Required agreement ratio

set of 10 offered movies, there seemingly is an optimum between 0.4-0.6. There is a clear trade-off between search space and finding agreement. If the amount of unseen movies is too small, not enough movies are rated, and a movie satisfying all users in a group might simply be missed. This probably also causes the larger rate of failure for small groups. However, if it is too large, not enough movies are 'exchanged' between users, and agreement is found too late.

4.3 Effect of different required agreement ratios

For the previous two experiments we required perfect agreement between the users in a session, in order for it to end successfully. However, it may be acceptable to consider consensus to be reached if a fraction of users within the group approves the movie. This especially is useful when a run will take too many swipe rounds, without finding a movie on which everyone in the group agrees; in that case, such a movie may not exist. In our simulation, we define a problematically long run to be a session taking longer than 50 swipes per user, as from a user interaction viewpoint, this will likely be above an acceptable threshold in real life. We investigated how many failed runs would occur when having different agreement ratios: for 0.5, 0.75 and perfect agreement. To limit the simulation time, We gave users 10 movies per swiping round before updating the model, and a 0.5 ratio for unrated movies, as this gave the best results on average in our second experiment. As visible in Figure 6, the amount of failed runs can easily be halved by lowering the required agreement. For larger groups, this agreement reduction is less important, due to the amount of failed runs already being relatively low.

5 CONCLUSION AND OUTLOOK

In order to acquire more realistic group recommender data for movie-watching scenarios, we proposed the design of a crowd-sourcing application that can assist the decision-making process towards a movie to watch with a current group of users. Current results from our simulation study indicate that such a system could indeed allow for reaching common consensus within few interaction steps.

Our simulation setup allows for more extensive studies to be run. For example, it should be investigated how dynamically updating the unrated movie and agreement ratio during a run may minimize failures. Starting with a higher ratio of unrated movies to maximize search space and adding more already rated movies in later rounds might optimize the trade-off between search space and finding agreement. Lowering agreement ratio in later rounds might solve sessions that are taking too long. It might also avoid getting stuck on unsolvable groups.

Also, it will be essential to investigate whether giving a 1 to 5 feedback rating at the end of a movie recommendation session can indeed be a relevant depiction of group effects in movie enjoyment.

In real life, there will not always be a full 'cold start' within a group. This might speed up our process towards convergence, and should both be investigated as part of simulations, and in real life scenarios.

Furthermore, users have probably watched some of the movies suggested. This might significantly lower the effectiveness of our application, and this should be tested more extensively. Generally, considering the framing of a preference elicitation app, if the user has seen a movie already, there should be an interaction option to skip it right now. For example, if the user swipes up the movie is incorporated in the rating group session, but the model will not be trained on it. This way, people won't negatively rate a movie they liked but only watched recently. The lowered effectiveness of the recommender could potentially be mitigated by such a design.

Our intention is to further expand our simulation studies, and start testing a prototype of our proposed app in real life with real groups of users, e.g. at student dormitories.

REFERENCES

[1] Sihem Amer-Yahia, Senjuti Basu Roy, Ashish Chawlat, Gautam Das, and Cong Yu. 2009. Group recommendation: Semantics and efficiency. *Proceedings of the VLDB Endowment* 2, 1 (2009), 754–765.
[2] Shlomo Berkovsky and Jill Freyne. 2010. Group-based recipe recommendations: analysis of data aggregation strategies. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 111–118.
[3] Shlomo Berkovsky, Jill Freyne, and Mac Coombe. 2009. Aggregation trade offs in family based recommendations. In *Australasian Joint Conference on Artificial Intelligence*. Springer, 646–655.
[4] Yen-Liang Chen, Li-Chen Cheng, and Ching-Nan Chuang. 2008. A group recommendation system with consideration of interactions among group members.

Expert systems with applications 34, 3 (2008), 2082–2090.

[5] Alexander Felfernig, Ludovico Boratto, Martin Stettinger, and Marko Tkalčič. 2018. *Evaluating Group Recommender Systems*. Springer International Publishing, Cham, 59–71. https://doi.org/10.1007/978-3-319-75067-5_3

[6] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2016), 19.

[7] Nicolas Hug. 2017. Surprise, a Python library for recommender systems. http://surpriselib.com

[8] Anthony Jameson. 2004. More than the sum of its members: challenges for group recommender systems. In *Proceedings of the working conference on Advanced visual interfaces*. ACM, 48–54.

[9] Judith Masthoff. 2011. Group recommender systems: Combining individual models. In *Recommender systems handbook*. Springer, 677–702.

[10] Mark O'Connor, Dan Cosley, Joseph A Konstan, and John Riedl. 2001. PolyLens: a recommender system for groups of users. In *ECSCW 2001*. Springer, 199–218.

[11] Thomas Ulz, Michael Schwarz, Alexander Felfernig, Sarah Haas, Amal Shehadeh, Stefan Reiterer, and Martin Stettinger. 2017. Human computation for constraint-based recommenders. *Journal of Intelligent Information Systems* 49, 1 (2017), 37–57.

[12] Zhiwen Yu, Xingshe Zhou, Yanbin Hao, and Jianhua Gu. 2006. TV program recommendation for multiple viewers based on user profile merging. *User modeling and user-adapted interaction* 16, 1 (2006), 63–82.