Attentive Score Aggregation for Group Recommender Systems

HORIA TURCUMAN, Technical University of Munich, Germany

This paper proposes an attention learning perspective on Score Aggregation, method used in Group Recommender Systems for group opinion prediction. The group recommendation task is challenging due to intricate group dynamics, and so far has not been thoroughly studied. In contrast, the standard, individual recommendation is well understood and solved through powerful methods. Score Aggregation builds on top of the individual ratings. It proposes to model the group decision process through weights which are used to form the weighted average of the group members' ratings. The assumption and motivation of this work are that by modeling the weights in an attentive way, the method becomes expressive enough to capture the group dynamics well and produce very good results.

CCS Concepts: • **Information systems** → **Rank aggregation**; **Learning to rank**; **Recommender systems**; • **Computing methodologies** → **Neural networks**.

Additional Key Words and Phrases: attention network

ACM Reference Format:
Horia Turcuman. 2022. Attentive Score Aggregation for Group Recommender Systems. 1, 1 (December 2022), 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The task of single-user recommendation has been extensively studied and effective solutions have been proposed [3]. This is indeed a great achievement that motivates researchers to tackle an even more convoluted but still related task: group recommendation. Humans are often involved in social activities such as watching movies, city breaks, or shopping. The quality of recommendations they receive strongly impacts their experiences and therefore the success of a recommender system. The dynamics of the decision-making process adds an additional layer of complexity on top of the standard, individual recommendation. Studies [2] show that diverse factors such as social relationships, expertise, and members' similarity strongly influences the decision of the group and it is important to correctly model them. Rather than manually identifying and modeling these factors, deep learning approaches postulate that they can be learned from data and automatically modeled through optimization. A couple of networks based on this approach [?] have been already proposed with various degrees of success, each considering more and more complex factors such as user-user, user-group, user-item relationships, external social attributes, and so forth. Some of them have identified the benefits of using attention neural networks for capturing these relationships, which led to state-of-the-art results and, consequently, are the main motivation for this work.

The Score Aggregation method uses individual user ratings to compute the group rating. Many different aggregation formulas have been studied so far, some of them yielding good results in certain cases, but none being suited for all kinds of groups and all situations. In contrast proposed deep learning methods learn user, group, and item representations

Author's address: Horia Turcuman, horia.turcuman@tum.de, Technical University of Munich, Arcisstraße 21, Munich, Bayern, Germany, 80333.

and networks that use these representations to finally determine the group-level scores. Group representation is the aggregation of members' representations and they are viewed as users in the sense that their representations are used in the same way as users' representations to determine the item score. These representations are learned through attention layers and networks that take into account various relationships that represent influential factors. This paper proposes learning the weights used in score aggregation using attention models such as the ones introduced in a few successful networks [1, 4, 7, 8] that have otherwise been used for group representation learning.

2 RELATED WORK

Previous works have explored the decision-making process of various groups, score and preference aggregation strategies, and influential factors. Others build on this knowledge and apply different strategies based on external group properties or inferred from data. Concretely, [2] combines three aggregation strategies namely Minimum Misery, Average Satisfaction, and Maximum Satisfaction, based on social relationships, expertise, and members' similarity. It offers good intuition for how these properties play a role in the decision process and why each strategy is better suited for a specific type of group. It also introduces the idea that important group properties can be inferred from data. The method proposed in this paper is more flexible in the sense that it can theoretically, through optimization, arrive at any combination of the three strategies mentioned if they are optimal, but does not restrict itself only to these combinations.

A couple of different group representation learning models [1, 4, 7, 8] have been recently developed, each identifying possible interactions and modeling them through attention layers and networks. Two of them are closely related to the model presented in this paper since the attention architectures from these two [1, 7] are extracted and used for learning the weights for score aggregation. On the other hand, they are all different because of the way these attention mechanisms are used.

This work demonstrates two different ways in which attention can be implemented for score aggregation and compares the performances of these architectures on a synthetically generated dataset leaving for future work the task of performing further experiments on real datasets and evaluating the effectiveness of the models on these more challenging settings.

3 DEFINITIONS

3.1 Problem Statement

Let n, m be the number of users and items respectively, $G \subseteq [n]$ denote a group, $s_u(u, i) : [n] \times [m] \to [0, 1]$ denote the score user u gives to item i and $s_g(i, G) : [m] \times \mathcal{P}([n]) \to [0, 1]$ denote the score group G gives to item i. Given n, m, s_u, and a set of tuples $\mathcal{D} = \{(i, G, s)\}$ of items, groups, and scores meaning that the item i has been rated with by the group G with score s, we are interested in producing s_g that rates for each group each item. Following best machine learning practices, we split the dataset $\mathcal{D} = \mathcal{D}_{train} \cup \mathcal{D}_{val} \cup \mathcal{D}_{test}$, and opt to minimize the mean squared error (MSE) loss:

$$\frac{1}{|\mathcal{D}_{test}|} \sum_{(i,G,s) \in \mathcal{D}_{test}} (s_g(i, G) - s)^2$$

3.2 Score Aggregation

Since we are interested in score aggregation, the function s_g can be written in terms of the input function s_u:

$$s_g(i, G = \{u_1, ..., u_{|G|}\}) = b(G) + \sum_{u \in G} \alpha(u, i, G) \cdot s_u(u, i)$$

where $\alpha(u, G, i)$ are the weights of user u depending on the group G and item i and $b(G)$ is some bias of the group. The bias is added since as [2] notes, while in a group, members might follow different goals than if they were individually considered. We will compare two different ways of learning the weights, both using attention networks.

3.3 User and Item Embeddings

For learning the weights using attention networks, either embeddings of users and items are used and learned together with the network's weights or an encoder can be used to map from user preferences, movie ratings, and other user characteristics to a low dimensional embedding space. Let us denote $U = \{\mathbf{u}_1, ..., \mathbf{u}_n\}$, $I = \{\mathbf{i}_1, ..., \mathbf{i}_m\}$ the embeddings of users respectively of items. Note that the embeddings are written in bold. The functions α that give the weights will use elements of these sets since they work with these embeddings instead of just with the user and item indexes.

4 CHALLENGES

A couple of challenges show up when doing score aggregation in contrast to representation learning. One evident problem is that individual user scores need to be available for all pairs of users and items, but this is never available in real datasets. To test on real data, an individual score prediction method [3] can be used as an initial step to complete the dataset. Of course, the result of the group recommender is then dependent on how good these individual predictions are. It is therefore challenging to apply score aggregation on real dataset since the lack of data, large the number of users or items and small number of groups could make the training fail. This is what was observed when training the individual user rating predictor on the CAMRa2011 data [6]. Due to lack of data, the network fails to generalize well, hence the group predictors can not be trained. To overcome this problem, a large synthetic movie dataset is generated on which the group recommendation models can be readily trained, skipping in this way the initial error-prone step of filling up the dataset with individual ratings. The experiments performed on this datasets demonstrate the expresiveness and power of the models and leave the task of filling in real dataset with sensible ratings to be further studied.

Another problem one needs to handle is the fact that the groups of users have variable sizes. This makes training especially difficult since parallelism can only be partially used with the proposed non-sequential models. Careful implementation has a large influence on the final results of the experiments.

5 MODELS

Two attention network architectures have been identified. In this section, short descriptions of each architecture are given with the goal of applying them to task of score aggregation, comparing their results and assessing how much different factors contribute to the decision-making process.

5.1 Attentive Group Recommendation (AGREE) [1]

AGREE models expertise (user-item interaction) of users on items. Intuitively, if users are familiar with an item or with items similar to it, they should have (and do have in real life) a more decisive say on the item. To model this, the following 2-layer network is proposed:

$$o(u,i) = \mathbf{h}^T \mathrm{ReLU}(\mathbf{P}_i \mathbf{i} + \mathbf{P}_u \mathbf{u} + \mathbf{b})$$

$$\alpha(u,i,G) = \mathrm{softmax}(o(u,i)) = \frac{\exp o(u,i)}{\sum_{u' \in G} \exp o(u',i)}$$

The weight matrices $\mathbf{W}_u, \mathbf{W}_c$ and bias b are used to project user embeddings and item embeddings into a hidden space, and together with the activation function, they form the first layer of the network. The $\mathbf{h}$ vector projects the output of the first layer into $\mathbb{R}$ giving a logit for each member of the group. The softmax activation is applied since the weights need to sum up to 1. This simple architecture does not consider factors as the influence among group members. This can be seen if one things about the logits of a user-item pair in different groups. Since the members of the group are not taken into account when computing the logits, they are going to be the same no matter what group is considered. Intuitively this is not how the decision process works and is therefore not very expressive.

5.2 Interact and Decide: Medley of Sub-Attention Networks for Effective Group Recommendation [7]

This model goes one step further by modeling apart from the expertise, the influence (user-user interaction) of users on each other. This is an important aspect that can be observed in real life: imagine how one parent may have different options if they are watching a movie with their child in comparison to when he is with his friends.

The first step of the MoSAN architecture is to construct for each user in the group G, $|G| - 1$ weights corresponding to each other user in the group. This is at a first glance not compatible with score aggregation but we will see how these weights can be combined to obtain only $|G|$ weights in the end. Figure 1 (taken from [7]) shows the main idea of this model.

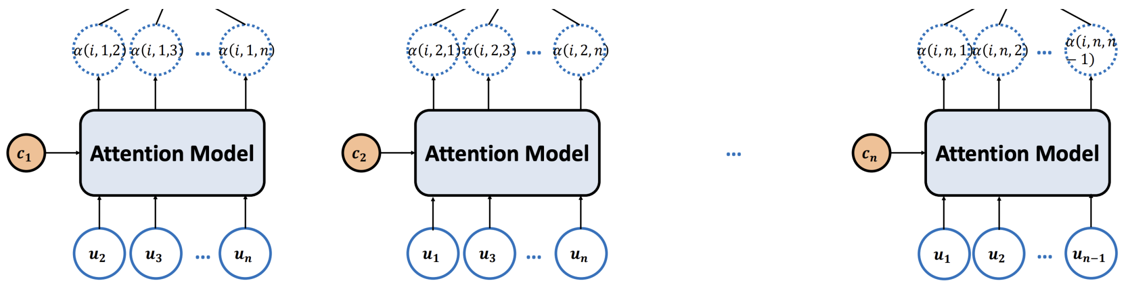


Fig. 1. High-level overview of the Medley of Sub-Attention Networks (MoSAN) model. Each sub-attention network is representative of a single group member, interacting with all other group members in order to learn its preference score

Note that since the picture is taken from the original paper, it uses a different notation for the weights. In what follows the α in the picture is replaced by o to keep the notation consistent with the rest of the text.

In addition to the embeddings defined in the previous section, this uses a set of embeddings $C = \{\mathbf{c_1}, ..., \mathbf{c_n}\}$ called the user-context vector, mainly used to differentiate the ownership of the attention sub-network. Each attention sub-network models the interaction between one user and all other group members. For $c \neq u$ the weights used in MoSAN are:

$$a(c,u) = \mathbf{w}^T \Phi(\mathbf{W}_c \mathbf{c} + \mathbf{W}_u \mathbf{u} + \mathbf{b}) + d$$
$$o(c,u,G) = \text{softmax}(a(c,u)) = \frac{\exp a(c,u)}{\sum_{\substack{u' \in G \\ u' \neq c}} \exp a(c,u')}$$

The reader can already notice that this is not using the item embeddings, therefore not modeling the expertiese. This is because the item embeddings are used later on in the model. Since there is no later step in our setting, we add the item knowledge here:

$$a(c,u,i) = \mathbf{w}^T \Phi(\mathbf{W}_c \mathbf{c} + \mathbf{W}_u \mathbf{u} + \mathbf{W}_i \mathbf{i} + \mathbf{b}) + d$$
$$o(c,u,i,G) = \text{softmax}(a(c,u,i)) = \frac{\exp a(c,u,i)}{\sum_{\substack{u' \in G \\ u' \neq c}} \exp a(c,u',i)}$$

Here both c and u are user indices, and the bold versions are their user context and user embedding. The weight matrices $\mathbf{W}_u, \mathbf{W}_c$ and bias b are used to project the user-context embedding and user embedding into a hidden space, and together with the activation function Φ they form the first layer of a network. The activation function is neglected in [7] by setting it to identity. The weight $\mathbf{w}$ and the bias d are the parameters of the second layer. The result is then normalized to form the weights o. All the matrices and biases used are shared among the attention sub-networks.

In the MoSAN model, these weights are multiplied with the corresponding user embedding and added together. This corresponds to the following score aggregation:

$$\begin{aligned} s_g(i,G) &= \frac{1}{|G|} \sum_{\substack{c,u \in G \\ c \neq u}} o(c,u,i,G) \cdot s_u(u,i) \\ &= \frac{1}{|G|} \sum_{u \in G} \left(\sum_{\substack{c \in G \\ c \neq u}} o(c,u,i,G) \right) \cdot s_u(u,i) \\ &= \sum_{u \in G} \alpha(u,i,G) \cdot s_u(u,i) \end{aligned}$$

Where α is:

$$\alpha(u,i,G) = \frac{1}{|G|} \sum_{\substack{c \in G \\ c \neq u}} o(c,u,i,G)$$

Note that the division by $|G|$ is necessary since outputs o of each attention sub-network add up to 1 and there are exactly $|G|$ such networks.

6 IMPLEMENTATION

The bottleneck of the two models is the fact that batches can not be processes as whole but a for-loop needs to be used. In the MoSAN network, even two for-loops are required since the context embeddings of all group member paired with all other group member embeddings are needed. This results in a very slow batch processing time making the training practically impossible. To alleviate this problem, the implementation moves part of the burden outside the batch processing function into a preprocessing step. With the information about group members, one can arrange the necessary embedding ids for each group in ready-to-be-used tensors which are stored in a list and can be indexed at training time. This introduces memory consumption but it drastically reduces the processing time and the model can be trained. The implementation can be hound here.

7 TRAINING

To find the best parameters, the datasets are split into training, validation and test parts of 80%, 10% and 10% respectively. The ADAM optimizer is used and the learning rate starts at 0.01 and is reduced when the model stops progressing. To find the suitable embedding dimension the values 16, 32 and 64 are tried and the hidden dimension is always set to the same value as the embedding dimension. The chosen batch size 1024 and the number of epochs is 5. Because of the standard for loops in the models, the training goes particularly slow, one batch needing more than 0.1 seconds to be processed for the first model and a couple of seconds for the second. Fortunately the networks do not need more than 4-8 epochs to converge which makes training possible. The objective function is the MSE loss as defined in a previous chapter.

8 EXPERIMENTS

There are a couple of research question one can attempt in our setting:

(1) How expressive are the models? Given much training data, how well do the models unravel the hidden decision processes of the groups and how do they perform?
(2) How good are the predictions? MSE loss and it's L1 correspondent are useful metrics to assess this.
(3)

To answer the questions a synthetically generated dataset will be used since this can be basically as large as needed and users can be involved in many different groups which is not the case with real datasets.

8.1 Synthetic Movie Dataset

The purpose of this dataset is to be used in understanding if the proposed models are able to pick up hidden structure in the data in order to have an idea of how promising they are. It doesn't claim to be similar to real datasets or that the considered features are correctly mimicked. Nevertheless it can be used to assess the predictors to some extend.

The generation process considers movie categories, different preferences of users for these categories, expertise, two different user types namely adults and children, three kinds of relationships between users, and three group types with different decision-making processes. User ratings are generated based on user type and preferences using the skewed normal and rayleigh distributions. Based on relationships, users are assigned to multiple groups which use one of the three decision processes described in [2]. The group types are as follows:

(1) Close Friends (Friends, Families, Couples): this group type uses the maximum satisfaction decision approach. Following this method, movies are rated with the highest of their ratings among group members. The idea is that being close friends, they watch movies together often and will be satisfied in turns. They are also likely to be satisfied if another member is especially happy and they are easily influenced by one another.
(2) Acquaintances: they essentially use the average satisfaction approach which averages the ratings of the group members. The average is actually implemented as a weighted average with weights adjusted based on the expertise of users on the movie category and on the user type (children have more priority).

(3) Strangers: by trying to be polite they make sure no one is too disappointed. They, therefore, use the least misery rating approach which assigns a movie the lowest of its ratings among the group members.

For generation, we use 700 adults, 300 children, 5000 movies, 11 movie categories and 450 groups (150 of each type). The high number of groups, low number of users and the fact that group ratings for all movies can be produced should allow a suitable model to very well train on this data.

The plots in figure 2 show the different properties of the dataset. They look similar to what one would expect from a movie dataset, hence, the experiments on it are of interest.

8.2 Results

The following table gives the test losses of both networks for each of the embedding sizes. To better understand the numbers it is useful to convert the to L1 loss values by taking the square root, hence they are introduced in the table as well.

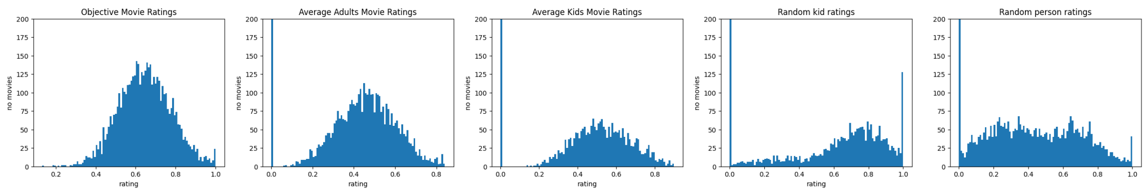
	MSE AGREE	MSE MoSAN	L1 AGREE	L1 MoSAN
16	0.00381	0.00650	0.061	0.080
32	0.00287	0.00587	0.053	0.076
64	0.00259	0.00656	0.050	0.080

Figure 3 depicts the training metrics for the AGREE-like network. All the models tend to converge t similar low values and there is no overfitting even without regularization on the largest model. This could mean that the model could reach even better results through further fine-tuning. It seems that the 16 dimension embeddings are able to reach low values but for better results, one needs to increase the dimension. Training did not become much slower by increasing the dimension so one could again look for better results by going to higher dimensions. These results are encouraging and suggest that one could replace the user and item embeddings by an encoder network to reduce memory burden and to be able to introduce new users and items. Details of this are presented in the last experiment.

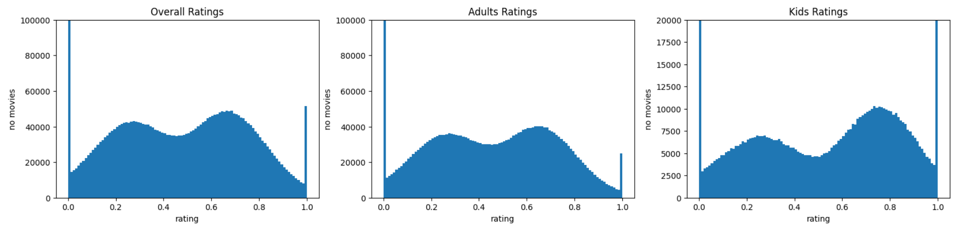
The MoSAN-like model, whose training metrics are shown in figure 4, does not perform as well as the AGREE model. Some reasons for the poor performance might be the failure to integrate the influence of members on each other in the dataset, the difficulty of training the MoSAN-like model, and the regularization as weight decay. With a lower weight decay, the model overfits. Using larger weight decay, does not reach the shown results. Since this model is more powerful than AGREE, one could argue that there is an issue with the regularization method used and further experiments may be able to yield better results.

Overall, it is clear that the AGREE-like model achieves good results with little training. For this reason, the first is chosen for the last experiment consisting of replacing the user and item embeddings with encoders. These encoders take as inputs all the ratings associated with a user or an item and output an embedding in a low dimensional space. The encoder weights are trained together with the rest of the network. What one gains by using such an encoder is the possibility of adding new users and items (for which all individual ratings are known) to the existing data, forming groups that contain the added users, and asking for group predictions on the added items. Intuitively, a suitable encoder architecture can lead to just as good results as found with the basic AGREE-like model.

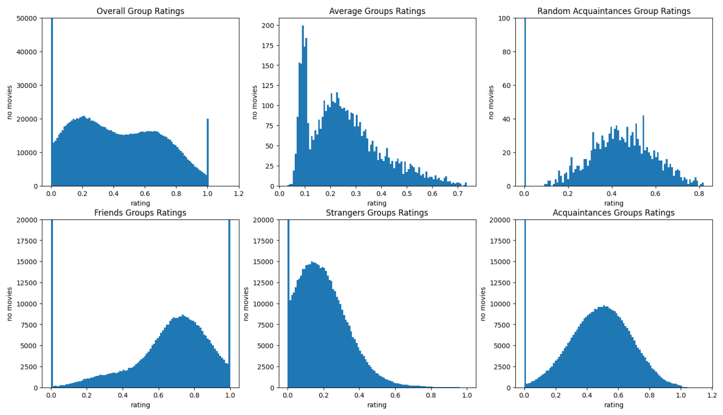
For training, the embedding size of 32 is chosen since it yielded very good results so far and it requires fewer parameters. The architecture contains 3 linear layers with ReLU activations of sizes 512, 128, and 32. The training shown in figure 5 goes a lot slower on a CPU this time and the test loss of 0.00973 is not encouraging. This architecture is still desirable in our setting, hence exploring it further would be important.



(a) From left to right the number of ratings (or movies) corresponding to each two-decimal possible rating is shown. The first figure shows the objective ratings of movies, that is the ratings that movie critics would agree upon. Starting with these ratings, users that do not like a category will rate less and those that like a movie's category rate more. Next plots show averages and sampled users rating distributions. The large number of 0 ratings comes from the fact that some categories are only suitable to one type of member (adult or kid) and users rate with 0 an unsuitable category. Large number of clamping generated values.



(b) These plots show the number of ratings for each possible two-decimal rating taking all user-item pairs into account not averaging per movie. The two modes correspond to two different preferences: like and unlike categories.

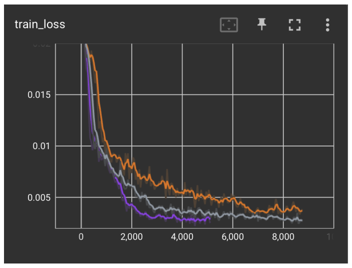


(c) These plots show the number of movies for all possible two-decimal ratings. The modes in the lower-level plots correspond to the decision strategies of different groups. The are many movies that averaged over groups get a low rating. This happens since unsuitable categories are rated with 0, there is a higher bias towards lower ratings since when people don't like a category they are likely to rate movies in that category much lower than the objective rating, and the minimum misery approach is used by strangers.

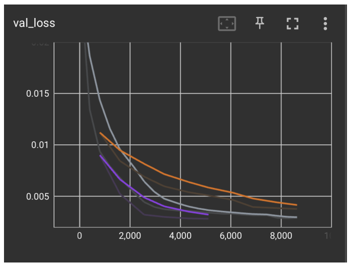
Fig. 2. Sanity check plots of the synthetic dataset.

9 CONCLUSION

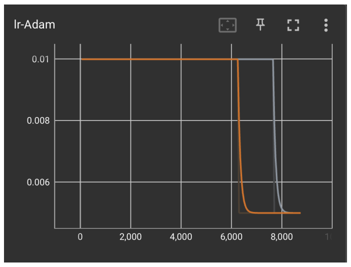
The two considered models have shown the ability to learn meaningful weights for score aggregation and seem to be a good alternative to other methods of designing these weights. More importantly, attention seems to be an innovative way of learning and designing group recommendation systems and it should be more extensively explored for this task. Unexpectedly, the simpler network yields better results than the more complex one but trials on real datasets are needed to confirm this result.



(a) learning rates

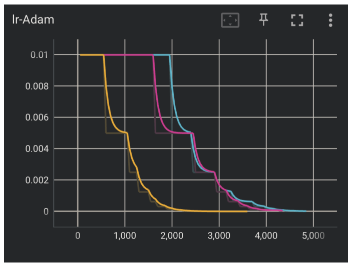


(b) training losses

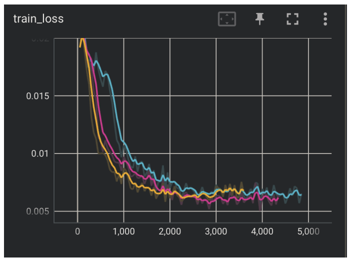


(c) validation losses

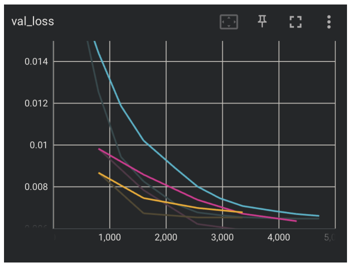
Fig. 3. Training of the AGREE-like model with various embedding dimensions.



(a) learning rates

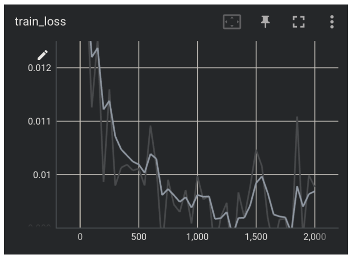


(b) training losses

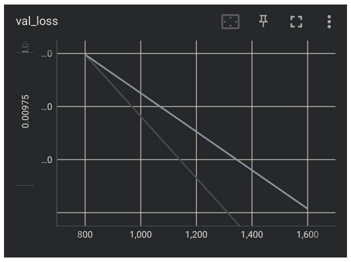


(c) validation losses

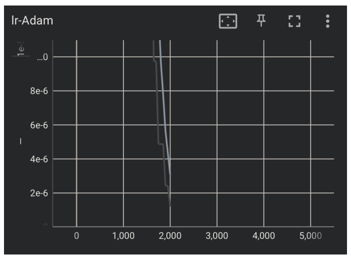
Fig. 4. Training of the MoSAN-like model with various embedding dimensions.



(a) learning rates



(b) training losses



(c) validation losses

Fig. 5. Training of the AGREE-like model with encoder and the 32-dimensional embedding space.

10 FURTHER WORK

The given methods could yield, through more training trials and different implementations better results than what they have so far shown. Testing them on real datasets would also be an important step in determining their usefulness. The attentive score aggregation setting can be further explored using other architectures and designing the weights in different ways. Possible such architectures are the ones presented in [4, 8]. They add the notion of group embeddings and other mechanisms that intend to capture other types of interactions (group-user, group-item) which have empirically been shown to play a role and to improve the models. Such architectures might become even more difficult to train and so smarter implementations are required. Lastly, attention models with variable input sizes could prove to be more suited to the discussed task of score aggregation weights modeling. They would remove the need for for-loops in the batch processing functions making in this way the training much faster and leading to possibly hidden very good results.

REFERENCES

[1] Da Cao, Xiangnan He, Lianhai Miao, Yahui An, Chao Yang, and Richang Hong. 2018. Attentive Group Recommendation. In *The 41st International ACM SIGIR Conference on Research amp; Development in Information Retrieval* (Ann Arbor, MI, USA) *(SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 645–654. https://doi.org/10.1145/3209978.3209998

[2] Mike Gartrell, Xinyu Xing, Qin Lv, Aaron Beach, Richard Han, Shivakant Mishra, and Karim Seada. 2010. Enhancing Group Recommendation by Incorporating Social Relationship Interactions. In *Proceedings of the 16th ACM International Conference on Supporting Group Work* (Sanibel Island, Florida, USA) *(GROUP '10)*. Association for Computing Machinery, New York, NY, USA, 97–106. https://doi.org/10.1145/1880071.1880087

[3] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web* (Perth, Australia) *(WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 173–182. https://doi.org/10.1145/3038912.3052569

[4] Ruxia Liang, Qian Zhang, and Jianqiang Wang. 2021. Hierarchical Fuzzy Graph Attention Network for Group Recommendation. In *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. 1–6. https://doi.org/10.1109/FUZZ45933.2021.9494581

[5]]DLGR Shane Miao. [n. d.]. Deep Learning Models for Group Recommendations. https://github.com/lucasvinhtran/group-recommender-systems#deep-learning. Accessed: 2022-11-29.

[6] Alan Said, Shlomo Berkovsky, Ernest W. De Luca, and Jannis Hermanns. 2011. Challenge on Context-Aware Movie Recommendation: CAMRa2011. In *Proceedings of the Fifth Conference on Recommender Systems (RecSys2011)* (Chicago, IL, USA). ACM, New York, NY, USA.

[7] Lucas Vinh Tran, Tuan-Anh Nguyen Pham, Yi Tay, Yiding Liu, Gao Cong, and Xiaoli Li. 2019. Interact and Decide: Medley of Sub-Attention Networks for Effective Group Recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Paris, France) *(SIGIR'19)*. Association for Computing Machinery, New York, NY, USA, 255–264. https://doi.org/10.1145/3331184.3331251

[8] Bojie Wang and Yuheng Lu. 2021. Graph Neural Netwrok with Interaction Pattern for Group Recommendation. *ArXiv* abs/2109.11345 (2021).