Leveraging Monte Carlo Tree Search for Group Recommendation

Antonela Tommasel
ISISTAN, CONICET-UNCPBA
Tandil, Argentina
antonela.tommasel@isistan.unicen.edu.ar

J. Andres Diaz-Pace
ISISTAN, CONICET-UNCPBA
Tandil, Argentina
andres.diazpace@isistan.unicen.edu.ar

Abstract

Group recommenders aim to provide recommendations that satisfy the collective preferences of multiple users, a challenging task due to the diverse individual tastes and conflicting interests to be balanced. This is often accomplished by using aggregation techniques that select items on which the group can agree. Traditional aggregators struggle with these complexities, as items are chosen independently, leading to sub-optimal recommendations lacking diversity, novelty, or fairness. In this paper, we propose an aggregation technique that leverages Monte Carlo Tree Search (MCTS) to enhance group recommendations. MCTS is used to explore and evaluate candidate recommendation sequences to optimize overall group satisfaction. We also investigate the integration of MCTS with LLMs aiming at better understanding interactions between user preferences and recommendation sequences to inform the search. Experimental evaluations, although preliminary, showed that our proposal outperforms existing aggregation techniques in terms of relevance and beyond-accuracy aspects of recommendations. The LLM integration achieved positive results for recommendations' relevance. Overall, this work highlights the potential of heuristic search techniques to tackle the complexities of group recommendations.

CCS Concepts

• **Information systems** → **Recommender systems**.

Keywords

recommender systems, group recommendation, Large Language Models, heuristic search, novelty, diversity, fairness

ACM Reference Format:
Antonela Tommasel and J. Andres Diaz-Pace. 2024. Leveraging Monte Carlo Tree Search for Group Recommendation. In *18th ACM Conference on Recommender Systems (RecSys '24), October 14–18, 2024, Bari, Italy*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3640457.3691713

1 Introduction

Traditional recommender systems have primarily focused on individual preferences. However, as the consumption of digital content increasingly occurs in social contexts, the demand for group recommender systems has increased. These systems provide recommendations to groups of users, acknowledging their diverse preferences and balancing the dynamic nature of their interactions [26], which poses challenges and opportunities.

A common approach to group recommendations is to aggregate the predicted rankings or candidate item ratings individually obtained for each group member using strategies inspired by Social Choice Theory [5, 13]. For example, an average strategy computes the group item rating as the average of the group member ratings [13]. However, aggregation strategies often select items independently [10]. Thus, while each recommended item may be individually relevant, the overall sequence of items might lack diversity, novelty, or fairness, which are key aspects of recommendations. For instance, all items might be highly similar, or certain user preferences might be over-represented [10]. Enhancing recommendation diversity and novelty has been linked to openness to new perspectives, empathy [14, 22], and positive user experiences [6], while it also contributes to increased fairness and reduced biases [20].

Aggregating the recommendations of each group member can be seen as a search problem, in which the search space includes all items recommended to each member, and the objective is to find a meaningful sequence of items that maximizes the overall group satisfaction. Items in a sequence can be consumed either in order (e.g., a playlist) or individually as part of a set. An item sequence can also help balancing recommendation properties (e.g., fairness, diversity) across group members [10]. *Monte Carlo Tree Search* (MCTS) [4] has emerged as a versatile search technique for complex settings, due to its ability to explore decision paths (e.g., a sequence of recommendations) and evaluate outcomes (e.g., the quality of recommendations). These characteristics make MCTS well-suited for aggregating user recommendations into group recommendations.

In this context, we propose a group recommendation approach relying on MCTS to seek candidate sequences of recommendations that best satisfy group preferences with respect to diversity, novelty and fairness. Furthermore, Large Language Models (LLMs) represent a paradigm shift for NLP techniques, which has shown interesting applications to recommendation tasks [8]. In particular, the knowledge posed by LLMs can provide rich, context-aware insights into user and group preferences and their impact on recommendation sequences, which can inform the MCTS, aiming at generating more diverse and novel group recommendations.

We performed an experimental evaluation, instantiated in a movie recommendation task, which showed that the proposed technique can outperform traditional and state-of-the-art (SOTA) aggregation techniques, both in terms of relevance and diversity and novelty. In some cases, the integration of LLMs contributed to further improving the relevance of recommendations. This study, although preliminary, aims to enhance the quality of group recommendations, laying the groundwork for future research.

RecSys '24, October 14–18, 2024, Bari, Italy

ACM ISBN 979-8-4007-0505-2/24/10
https://doi.org/10.1145/3640457.3691713

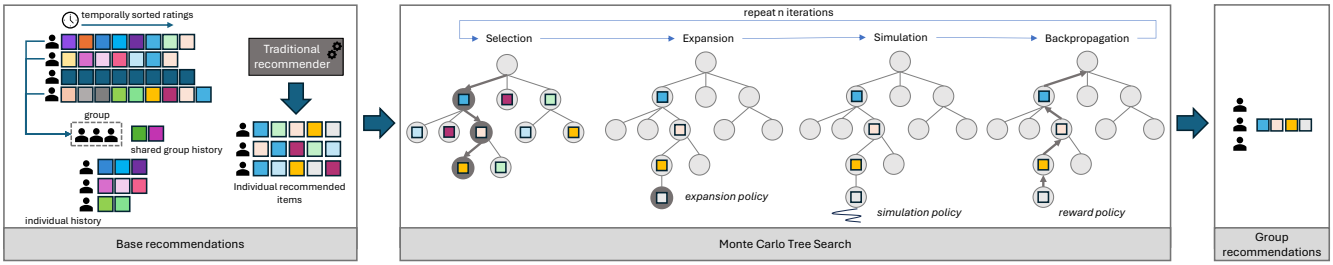

Figure 1: Schematic steps of the proposed pipeline

2 Related works

Group recommendation systems often rely on aggregating the recommendations obtained for each group member [13]. Items are generally selected independently. Recently, some studies [10, 12, 26] began examining the properties of the set of recommended items, particularly focusing on fairness, in terms of balancing user utilities within a group, and following a Greedy approach for item selection.

Xiao et al. [26] defined the overall user satisfaction as the linear combination of social welfare (user utilities in a group) and fairness (balance of user utilities in a group). Utility was measured as the mean predicted item relevance, while fairness followed standard utility-based definitions such as least misery, variance and min-max ratio aiming at minimizing utility differences between group members. Kaya et al. [10] defined fairness inspired on intent-aware information retrieval, aiming at diversifying recommendations to cover users' different interests. Item relevance was based on Borda, and fairness was defined as the sum of each group member's probability of finding at least one relevant item within recommendations. Unlike [26], the proposal was rank-sensitive, as item order was essential in balancing item utility across group members. Malecek and Peska [12], inspired by the D'Hondt seat allocation method, focused on maximizing the per-user proportional fraction of relevance.

While the works above are based on greedy approaches that focus on the immediate next step by selecting the locally optimal choice, our approach allows for a more thorough exploration of the search space, which potentially leads to (better) medium-term decisions that consider item dependencies. Moreover, unlike prior works, which only define fairness in terms of utilities, we also focus on the diversity and novelty of recommended items. Similarly to [10], in our proposal, the recommendation order is important.

Regarding MCTS and recommender systems, Rajapakse and Leith [17] used MCTS to learn user preferences in a cold-start scenario. Liebman et al. [11] applied MCTS in a reinforcement learning scenario to recommend playlists. Finally, recently, Zhao et al. [28] proposed using LLMs as an MTCS policy to guide the search in a planning scenario, showing promising results in improving an MTCS schema in some scenarios. To the best of our knowledge, MCTS has not yet been explored for group recommendation.

3 Task and method

We defined a pipeline (Figure 1) that tackles the generation of base recommendations for each group member, and the MCTS-based aggregation technique. The output is the sequence of items comprising the group recommendations.

As a first step, we assume an underlying recommender system generating recommendations for each group member based on their individual preference profile (e.g., item interaction history). Aggregating such recommendations so that a sequence of items best satisfies the group is cast to a search problem. In this setting, the possible decisions are the items recommended to each group member, and the objective is to obtain a sequence of decisions that maximizes group satisfaction. Assessing group satisfaction is challenging [10, 26], specially when introducing objectives related to the overall diversity, novelty and fairness of recommendations. In our approach, we propose MCTS to select promising item sequences without making a (full) combinatorial search. Furthermore, we integrate an LLM as an alternative strategy to evaluate group satisfaction. This means that rather than computing diversity and novelty metrics based on the items, we ask the LLM to assess the items and provide a numeric score. We conjecture that an LLM can be prompted to perform a more holistic evaluation of the situation (e.g., group context, users' history, available items) than traditional metrics, as recently suggested in [15, 28]. That is, as the MCTS searches for candidate items, we leverage the LLM knowledge to assess the suitability of the item sequences for a group.

In our study, we focus on two research questions:

• **RQ1**: *Does MCTS allow to improve the quality of group recommendations?* We compare the performance of group recommendation aggregation strategies with MCTS-based strategies.

• **RQ2**: *Does the combination of MCTS and LLM contribute to better assess group satisfaction?* We extend the MCTS reward definition with an LLM and assess whether recommendations are improved.

3.1 Monte Carlo Tree Search

MTCS [4] is a heuristic algorithm for sequential decision processes, which implements an iterative tree search that balances exploration (i.e., explore a new decision) and exploitation (i.e., pick the best known decision). In this tree, nodes correspond to states while edges denote transitions (e.g., actions or decisions) from one state to another. Instead of performing an exhaustive search in the space of alternatives, MCTS only searches a subset of nodes and identifies promising parts of the tree to be explored. MCTS then simulates the outcome (or reward) of a given state and propagates those values up in the tree. Thus, a key advantage of MCTS is its effectiveness for exploring alternatives in large spaces. As the algorithm iterates more, the likelihood of reaching optimal decisions increases.

In our recommendation problem, each tree node represents a decision of an item being recommended to a group. Thus, a path from the root node to any other node captures a possible sequence of items to be recommended. These paths are stored in the respective nodes as states. We see the reward of a node as a score that indicates the degree to which its state (i.e., the path ending at that node, or item sequence) satisfies the whole group.

The MCTS algorithm involves four phases (Figure 1):

1) Selection. It starts from the root node and selects the next node to explore (i.e., the next item to recommend) at each level of the existing tree. A common policy is to rely on an Upper Confidence Bound (UCB) to balance exploitation and exploration. This phase ends when a leaf node is visited and the next node is either not yet represented or a terminal state. In our problem, a leaf node (and a terminal state) means that there are no more items to recommend to the group and we have obtained a full item sequence.

2) Expansion. If a terminal state is not reached, this phase adds a new child node to the tree, representing a state that results from applying a feasible action on the parent state. Since feasible actions map to candidate items in our problem, applying an action is equal to deciding on a given item to be recommended. If a terminal state is reached, a *reward policy* computes the node's outcome, which is typically a domain-specific numeric score (e.g., diversity, novelty, or an LLM-based assessment). The process then moves to phase 4.

3) Simulation. It performs a simulation, such as choosing actions randomly, until a terminal state is reached. The purpose of the simulation is to quickly (and approximately) determine the reward of a node, when performing a deep expansion of the nodes is unfeasible or computationally expensive. The outcome of the (simulated) terminal state is again assessed via the *reward strategy*. In particular, we can use simulation in our problem to speed up the search, although the quality of the resulting sequences might suffer.

4) Backpropagation. Scores are propagated back from the last visited node to the root, updating node statistics. This helps determining the current best action at any given time. In our problem, the current best action is the next item to be recommended to the group to optimize its satisfaction. By following the best action and traversing the tree until reaching a terminal state we can also obtain the best path, which is the desired sequence of items to recommend.

The MCTS output is the best sequence of actions found given the current tree search. The algorithm has several configurable parameters (e.g., the maximum number of expansions, or iterations) that influence its behavior. Furthermore, one can set different policies for expansion, simulation and reward. A simple option for expansion and simulation is to randomly choose candidate actions. The reward policy is often a function of the action/state representation.

Reward policy configuration. In our problem, a basic reward policy is to use traditional metrics for diversity and novelty. Along this line, we experimented with different reward alternatives:

i) *Diversity (MCTS-diversity).* Diversity refers to the variety of the candidate recommendations, ensuring that they cover a broad range of topics, genres, categories, or styles. It aims at preventing recommendations from being too similar to each other, avoiding redundancy and yielding a balanced selection that captures different user/group preferences. Diversity was measured as the average intra-list dissimilarities [18] of the recommended items. Dissimilarity was, in turn, measured as the Euclidean distance of the item representations (e.g., a content-based representation).

ii) *Novelty (MCTS-novelty).* Novelty refers to how new or unexpected the recommended items are when compared to what users have interacted with previously. It aims at recommending content to users/groups beyond their typical consumption patterns. Novelty was measured as the average intra-list dissimilarities between the candidate recommendations and users' item consumption history. For dissimilarities, the same metric as in *MCTS-diversity* was used.

iii) *Maximal Marginal Relevance (MCTS-MRR).* It computes a linear combination of diversity and novelty [3].

iv) *Fairness (MCTS-fairness).* We treat it as the average proportion of users to whom the candidate items were originally recommended.

v) *Weighted Fairness-Novelty (MCTS-weighted).* In this case, *MCTS-fairness* acts as a multiplier of *MCTS-novelty*.

To explore the role of LLMs in assessing rewards [15, 28], we defined strategies aiming at analyzing the interactions between user and group preferences and candidate items. We replaced the previous strategies by an evaluation prompt, instructing the LLM to return a numeric score (between 0 and 100)[1]:

i) *Base (MCTS-LLM)*: The LLM is asked to evaluate candidate items based on group preferences (the set of items liked by all members) and individual preferences (for each member, the set of items exclusively liked by them), considering how well items satisfy the whole group focusing on diversity and novelty. As a variant, the score should also consider the order of candidate items.

ii) *Ranking (MCTS-LLM-rank)*: The LLM is asked to evaluate candidate items based on group preferences, individual preferences, and the recommendations made for each group member. The score should reflect diversity, novelty, fairness, or their combination. Unlike *MCTS-LLM*, the LLM is informed about the items recommended to each user, even if those are not part of the candidate set.

iii) *Position (MCTS-LLM-pos)*: Similar to *MCTS-LLM-rank*, but instead of the recommendations for each user, the LLM receives information of whether the items were recommended to each user and in which ranking positions.

4 Experimental setup

When evaluating recommendations [2], it is critical to decide whether to follow a *full evaluation*, in which any item can be recommended to any group member; or a *restricted/condensed evaluation* in which only a selected set of groups' unseen items can be recommended.

The restricted evaluation helps addressing data size and sparseness. First, only a small fraction of potentially relevant items are rated [2, 16], and the probability of being rated is not uniform, but biased by the data collection process [12, 19]. This can cause the base recommender to suffer from popularity bias, favoring top items over items that might be acceptable to more group members, even if they are rated lower [12]. Second, in the full evaluation, the base recommender and the aggregation technique are evaluated together (as a coupled pair [16]), potentially introducing bias, as the aggregation technique's quality is influenced by the recommender's quality. By restricting the candidate items, we limit the recommender's influence, focusing the evaluation on the aggregation strategies.

We defined the group candidate items to include all items in the test set (and their corresponding real ratings) liked by all members, at least 5 exclusively liked by each member (liked by a user and either disliked or not rated by any other), and 5 disliked by each member and not liked by others[2].

[1]The full prompts and code can be found in the companion repository: https://github.com/tommantonela/MCTS_group_recommendation

[2]The rationale for the number of items was to build both a comprehensive set of candidate items and allow for better preference control towards specific users. Also, for the LLM-based strategies, it helps to avoid exceeding LLMs' token limits.

Data collection. We used the *MovieLens* dataset[3], including user ids, movies, ratings, genres, and timestamps. We considered ratings later than January 1, 2016, and kept users rating more than 30 movies. The ratings of each user were split based on timestamps, with a 60%/20%/20% train/val/test partition. The test set of each group member only included movies that were not part of any member's train set to avoid recommending seen elements as part of the group recommendation [10]. Movies were considered liked if rated 4 or higher. For the purpose of analyzing diversity/novelty, a content-based representation of movies was defined based on their genre vector. For the LLM-based MCTS strategies, the "group preferences" and "individual preferences" were built considering the movies in users' train sets.

Group formation. As data did not include group information, we created synthetic groups[4]. Following [10, 26], we simulated 150 groups of 2 to 8 members with similar watching histories (a proxy for similar interests). We computed user similarity over the full data collection [1, 27] based on the Pearson Correlation of their ratings [1], ensuring each pair's similarity exceeded a set threshold.

Base recommendations. Base recommendations were performed using *ImplicitMF* [9], a commonly-used top-performing matrix factorization technique tailored for implicit feedback settings. *ImplicitMF* has a single hyper-parameter, the number of factors, which we tuned over the validation set optimizing nDCG.

LLM choice. The LLM-based strategies used `Gemma:2b-instruct`, which showed a good balance between execution time and answer quality (in terms of prompt adherence)[5].

Baselines. We considered: i) Common social choice aggregation techniques [13]: *additive*, *least-misery*, and *Borda* rank aggregation; ii) *GreedyLM* [26]; iii) *GFAR* [10] ; iv) *EP-FuzzDA* [12]; and v) LLM-based aggregation (*LLM-agg*). The LLM was provided with groups' common and users' individual watching history, the list of movies recommended to each user, and asked to aggregate the user recommendations into one list satisfying all group members.

Evaluation metrics. We considered both relevance and beyond-accuracy perspectives [21]: coverage, ordering and diversity/novelty. For *relevance*, precision and nDCG[6]. For *coverage*, zero-recall (as in [10]), measuring the fraction of group members not receiving any relevant recommendation. This is also considered a fairness metric, as values close to zero indicate that every group member received at least one relevant item. As users watch movies in a certain order, and *order* is important in generating the MCTS candidate sequences, we considered Kendall's Tau rank correlation between the real watched order and the recommendation sequence[7]. For *diversity* and *novelty*, we followed the same definitions as in the reward strategies.

We selected the top-5 recommendations. Metrics were computed at user level, and then summarized (e.g., minimum, mean, median, variance) to obtain the group score [10, 23]. For each user, recommendations were deemed correct if they appeared in their test set with a rating of 4 or higher. To assess the statistical significance of differences, the Wilcoxon paired samples test ($\alpha = 0.05$) was used, including multiple test corrections.

[3]https://grouplens.org/datasets/movielens/25m/

[4]Code and more details can be found in the companion repository.

[5]We also evaluated `Mistral:7b` and `Llama3`, but discarded them due to either excessive execution times on available hardware or inappropriate prompt adherence.

[6]We followed the binary definition: items are either relevant or non-relevant.

[7]As correlation metrics require sequences to have the same number of elements, the analysis only included the sub-set of correctly recommended ones.

	prec	nDCG	zrecall	corr	div	nov
LLM-agg	0.127	0.273	0.584	0.199	0.721	0.746
additive	0.099	0.238	0.623	0.075	0.64	0.72
least-misery	0.127	0.259	0.641	**0.48**	0.475	0.717
Borda	0.12	0.291	0.536	0.147	0.709	0.732
GreedyLM [26]	0.100	0.244	0.610	0.192	0.744	0.748
GFAR [10]	0.114	0.288	0.536	-0.083	0.761	0.751
EP-FuzzDA [12]	0.117	0.285	0.546	0.222	0.747	0.746
MCTS-diversity	0.105	0.263	0.576	0.239	**0.872**	0.779
MCTS-novelty	0.125	0.295	0.535	0.330	0.737	**0.806**
MCTS-MRR	0.108	0.269	0.568	0.253	*0.867*	*0.790*
MCTS-fairness	0.105	0.262	0.578	0.359	0.676	0.722
MCTS-weighted	0.120	0.287	0.546	0.292	0.742	**0.809**
MCTS-LLM	*0.140*	*0.317*	*0.508*	0.352	0.666	0.71
MCTS-LLM-rank	0.128	0.285	0.560	*0.440*	0.681	0.724
MCTS-LLM-pos	**0.148**	*0.344*	**0.460**	0.340	0.700	0.735

Table 1: Summarized mean group scores ($k = 5$) (best results in bold, second best in *italic*)

5 Analysis

Table 1 presents the mean results for groups of 5 members[8]. Group metrics were computed as the mean member score. For correlations, means were computed based on a meta-analysis [24, 25]. The MTCS was run for 200 iterations, and no simulation strategy was used.

5.1 RQ1. Baselines versus MCTS

LLM-agg achieved high values, showing statistically significant differences for precision, nDCG and novelty compared to traditional baselines and *GreedyLM*. It was only outperformed by *least-misery* (precision), although the differences were not statistically significant. *Least-misery* achieved the highest precision, lowest diversity and strongest correlations. Conversely, *additive* achieved the lowest precision and nDCG. SOTA baselines showed statistically significant improvements in novelty and diversity over all traditional baselines, with *GFAR* and *EP-FuzzDA* also statistically significantly improving nDCG compared to *least-misery* and *additive*. In general, over 50% of users in a group did not receive at least one relevant recommendation (zrecall). *GreedyLM* performed statistically significantly worse than other baselines in precision, nDCG, and diversity.

Compared to baselines, *MCTS-novelty* had the best performance. On average, it enhanced the precision and nDCG of traditional and SOTA baselines by 11.9% and 11.07%. MCTS correlations were higher than for the baselines (except for *least-misery*), with *MCTS-novelty* showing statistically significant differences and better recommendation ordering. *MCTS-fairness* achieved statistically significantly lower results than baselines for at least one metric.

Although MCTS strategies did not improve the relevance of *LLM-agg*, they showed statistically significant differences in diversity and novelty. Also, MCTS strategies improved the zrecall of several baselines. *MCTS-diversity* and *MCTS-MRR* achieved the highest diversity (and low precision), while *MCTS-novelty* and *MCTS-weighted* achieved the highest novelty, with average improvements of 9%. Statistically significant differences were observed for all baselines in both cases.

[8]Additional results can be found in the companion repository.

The statistically significantly differences observed favouring the MCTS-based technique when compared to the greedy-based baselines ([10, 12, 26]) can be attributed to MCTS's exploration of a broader range of item sequences. While greedy algorithms make locally optimal choices at each step, focusing on the immediate gain, MCTS explores multiple sequences of items, allowing it to discover less obvious, but potentially more rewarding, options. This exploration helps MCTS identify novel recommendations that a greedy algorithm might miss, as it prioritizes the overall group satisfaction of whole recommendation sequences rather than the short-term gains introduced by (incrementally) adding one item to the recommendations. By balancing exploration and exploitation MCTS is more likely to introduce new items, thus, enhancing novelty in the recommendations.

Among the MCTS strategies, *MCTS-novelty* generally showed statistically significant improvements, with average improvements of 14.73%, 9.46%, and 4.09% in precision, nDCG, and novelty, respectively. *MCTS-diversity* only obtained good results for diversity. *MCTS-fairness* had the worst overall results, with the exception of correlation; nonetheless, differences were not statistically significant. *MCTS-MRR* did not enhance the results of *MCTS-novelty* and *MCTS-diversity*. These findings suggest that prioritizing novelty in the reward function yields better results than prioritizing diversity.

> The MCTS strategies statistically significantly improved beyond-accuracy metrics while maintaining similar precision and nDCG levels. The best results were observed when novelty was included in the reward function. Correlation differences underscored the rank-sensitive nature of recommendations.

5.2 RQ2. Integrating LLM in MCTS

The last block in Table 1 presents the best results for the MCTS LLM-reward strategies[9]. Most differences were not statistically significant, indicating that prompting the LLM to consider diversity, novelty, fairness (or their combination) had little effect on outcomes. The only exceptions favoured diversity for *MCTS-LLM-Pos* (nDCG) and *MCTS-LLM-Rank* (diversity). Although the LLM may have a general knowledge of concepts like diversity, novelty, or fairness, it might not be enough to adequately assess the candidate movies and improve recommendations. For *MCTS-LLM*, asking the LLM to consider the order of candidate movies did not benefit recommendations. This suggests that prompt changes were too subtle or did not adequately stress aspects' importance. Thus, further exploration of prompt design, with clear definitions and justifications, is needed.

Results showed that including user recommendation details (*MCTS-LLM-Pos* and *MCTS-LLM-Rank*) statistically significantly improved relevance and novelty. Hence, providing insights of the base recommender's behavior and how recommendations align with users can enhance the LLM's reasoning, leading to better recommendations. *MCTS-LLM-Pos* achieved statistically significantly higher precision and nDCG than *MCTS-LLM-Rank*. This could also be related to prompt quality and length, as including all user recommendations could introduce noise, whereas simply indicating items' positions can be less confusing for the LLM.

[9]The full set of results and statistical significance analysis can be found in the companion repository.

Comparing the LLM-aware rewards with those from **RQ1** revealed that incorporating LLMs increased precision and nDCG by up to 10.86% and 6.82%. Correlations were also boosted. However, diversity and novelty decreased, with a statistically significant drop in diversity. This decline may be due to prompt definition and LLM's assumptions about diversity and novelty, as discussed previously.

> The LLM-guided reward strategies improved the relevance of baselines and non-LLM strategies. Results highlighted the potential of LLMs and hinted points for enhancements.

6 Conclusions & Outlook

We proposed a novel approach based on MCTS for group recommendation. We empirically evaluated several configurations, which showed that MCTS (even without LLMs) can outperform existing baselines in relevance and beyond-accuracy aspects.

Despite the promising results, our study has some limitations that can be addressed in future research. First, we only focused on the commonly used *MovieLens* [10, 26] collection. While this choice demonstrated the technique's applicability, evaluating other domains (e.g., music, books) and group characteristics is necessary for assessing the generalizability of our findings. Second, we considered only one base recommender. Evaluating other recommenders would help assess the sensibility of the proposed technique to base recommendations and whether improvements follow similar trends. Third, although we considered a restricted evaluation scenario to reduce the dependency on the quality of the base recommendations, we still considered the real ratings as ground truth. In this sense, other evaluation scenarios could be explored. For example, we can aim for an unbiased [7], or a decoupled evaluation in which the base recommendations are used as ground truth simulating that all user preferences are known and aggregators are evaluated with respect to how effectively they combine those preferences [16].

Fourth, evaluation was limited to a specific LLM and parameters. As discussed in Section 5, more analyses are needed to determine if alternative LLMs and prompting techniques can yield better results. Fifth, the reported MCTS results involved a predefined number of iterations, random selection of items for the expansion strategy and no simulation. We think that, in addition to the reward strategy, LLMs could be applicable to MCTS expansion and simulation strategies, albeit with higher computational costs. As part of an exploratory evaluation[10], we ran the MCTS with a random candidate simulation strategy, which helped reducing the algorithm's computational cost with almost no result degradation. We also noticed that the reward strategies reached a limit beyond which more iterations did not improve results. This encourages us to pursue further work on integrating MCTS and LLMs as heuristics for group recommendation. Finally, from a broader perspective, MCTS can be part of a reinforcement learning approach targeting long-term novelty, diversity and fairness in a multi-objective context.

Ethical statement. While LLMs can enhance recommendation accuracy and user experience, they also raise concerns about fairness and bias. Ensuring transparency, accountability, and fairness in LLM-based recommendation systems is crucial for addressing ethical aspects and maintaining user trust.

[10]Results can be found in the companion repository.

References

[1] Linas Baltrunas, Tadas Makcinskas, and Francesco Ricci. 2010. Group recommendations with rank aggregation and collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*. 119–126.

[2] Rocío Cañamares, Pablo Castells, and Alistair Moffat. 2020. Offline evaluation options for recommender systems. *Information Retrieval Journal* 23, 4 (2020), 387–410.

[3] Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. 335–336.

[4] Rémi Coulom. 2006. Efficient selectivity and backup operators in Monte-Carlo tree search. In *International conference on computers and games*. Springer, 72–83.

[5] Alexander Felfernig, Müslüm Atas, Denis Helic, Thi Ngoc Trang Tran, Martin Stettinger, and Ralph Samer. 2023. Algorithms for group recommendation. In *Group recommender systems: An introduction*. Springer, 29–61.

[6] Bruce Ferwerda, Mark P. Graus, Andreu Vall, Marko Tkalcic, and Markus Schedl. 2017. How Item Discovery Enabled by Diversity Leads to Increased Recommendation List Attractiveness. In *Proceedings of the Symposium on Applied Computing* (Marrakech, Morocco) *(SAC '17)*. Association for Computing Machinery, New York, NY, USA, 1693–1696. https://doi.org/10.1145/3019612.3019899

[7] Alois Gruson, Praveen Chandar, Christophe Charbuillet, James McInerney, Samantha Hansen, Damien Tardieu, and Ben Carterette. 2019. Offline evaluation to make decisions about playlistrecommendation algorithms. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 420–428.

[8] Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2024. Large language models are zero-shot rankers for recommender systems. In *European Conference on Information Retrieval*. Springer, 364–381.

[9] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE international conference on data mining*. Ieee, 263–272.

[10] Mesut Kaya, Derek Bridge, and Nava Tintarev. 2020. Ensuring fairness in group recommendations by rank-sensitive balancing of relevance. In *Proceedings of the 14th ACM Conference on recommender systems*. 101–110.

[11] Elad Liebman, Piyush Khandelwal, Maytal Saar-Tsechansky, and Peter Stone. 2017. Designing better playlists with monte carlo tree search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 31. 4715–4720.

[12] Ladislav Malecek and Ladislav Peska. 2021. Fairness-preserving group recommendations with user weighting. In *Adjunct Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization*. 4–9.

[13] Judith Masthoff. 2010. Group recommender systems: Combining individual models. In *Recommender systems handbook*. Springer, 677–702.

[14] Madison Grace McClung and Angèle Christin. 2018. Filter Bubbles And Music Streaming : The Influence of Personalization And Recommendation Algorithms on Music Discovery Via Streaming Platforms.

[15] Allen Nie, Ching-An Cheng, Andrey Kolobov, and Adith Swaminathan. 2023. Importance of Directional Feedback for LLM-based Optimizers. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*. https://www.microsoft.com/en-us/research/publication/importance-of-directional-feedback-for-llm-based-optimizers/

[16] Ladislav Peska and Ladislav Malecek. 2021. Coupled or Decoupled Evaluation for Group Recommendation Methods?. In *Perspectives@ RecSys*.

[17] Dilina Chandika Rajapakse and Douglas Leith. 2022. Fast and accurate user cold-start learning using monte carlo tree search. In *Proceedings of the 16th ACM Conference on Recommender Systems*. 350–359.

[18] Dimitris Sacharidis. 2019. Diversity and Novelty in Social-Based Collaborative Filtering. In *Proceedings of the 27th ACM Conference on User Modeling, Adaptation and Personalization* (Larnaca, Cyprus) *(UMAP '19)*. ACM, New York, NY, USA, 139–143. https://doi.org/10.1145/3320435.3320479

[19] Yuta Saito, Suguru Yaginuma, Yuta Nishino, Hayato Sakata, and Kazuhide Nakata. 2020. Unbiased recommender learning from missing-not-at-random implicit feedback. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 501–509.

[20] Javier Sanz-Cruzado and Pablo Castells. 2018. Enhancing structural diversity in social networks by recommending weak ties. In *Proceedings of the 12th ACM conference on recommender systems*. 233–241.

[21] Javier Sanz-Cruzado and Pablo Castells. 2018. Enhancing Structural Diversity in Social Networks by Recommending Weak Ties. In *Proceedings of the 12th ACM Conference on Recommender Systems* (Vancouver, British Columbia, Canada) *(RecSys '18)*. ACM, New York, NY, USA, 233–241. https://doi.org/10.1145/3240323.3240371

[22] Maria Taramigkou, Efthimios Bothos, Konstantinos Christidis, Dimitris Apostolou, and Gregoris Mentzas. 2013. Escape the bubble: Guided exploration of music preferences for serendipity and novelty. In *Proceedings of the 7th ACM conference on Recommender systems*. 335–338.

[23] Christoph Trattner, Alan Said, Ludovico Boratto, and Alexander Felfernig. 2023. Evaluating group recommender systems. In *Group recommender systems: an introduction*. Springer, 63–75.

[24] Robbie CM van Aert. 2023. Meta-analyzing partial correlation coefficients using Fisher's z transformation. *Research Synthesis Methods* 14, 5 (2023), 768–773.

[25] David A Walker. 2003. JMASM9: converting Kendall's tau for correlational or meta-analytic analyses. *Journal of Modern Applied Statistical Methods* 2 (2003), 525–530.

[26] Lin Xiao, Zhang Min, Zhang Yongfeng, Gu Zhaoquan, Liu Yiqun, and Ma Shaoping. 2017. Fairness-aware group recommendation with pareto-efficiency. In *Proceedings of the eleventh ACM conference on recommender systems*. 107–115.

[27] Quan Yuan, Gao Cong, and Chin-Yew Lin. 2014. COM: a generative model for group recommendation. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 163–172.

[28] Zirui Zhao, Wee Sun Lee, and David Hsu. 2024. Large language models as commonsense knowledge for large-scale task planning. *Advances in Neural Information Processing Systems* 36 (2024).