Counterfactual Explanations for Group Recommendations

Maria Stratigi[1,*], Nikos Bikakis[2] and Kostas Stefanidis[1]

[1]Tampere University, Tampere, Finland
[2]Hellenic Mediterranean University, Chania, Greece

Abstract

Recommendations provided by group recommendation systems are often complex and difficult for average users to comprehend. To address this issue, many existing approaches incorporate explanations alongside recommendations. However, as group recommendation models become increasingly sophisticated, offering clear and understandable explanations becomes more challenging. In response, we propose a system that delivers counterfactual, model-agnostic explanations. Counterfactual explanations focus exclusively on items with which the group has interacted, enhancing their interpretability. Additionally, model-agnostic explanations mitigate the intricacies of complex group recommendation models by treating them as black boxes, thereby simplifying the process and making the recommendations more accessible to users. We propose two heuristic approaches to produce fair group counterfactual explanations, i.e., explanations that consider all group members' input and are not focused on individual group members. We conduct an evaluation of our methods utilizing the MovieLens dataset, highlighting our proposed methods' effectiveness in generating counterfactual explanations for groups within a practical and efficient timeframe.

Keywords

Recommendation Systems, Explanations, Fairness, Counterfactual Explanations

1. Introduction

Recommendation systems are widely used across various applications, helping users navigate vast amounts of information by providing personalized suggestions. One common example is movie recommendations, where the system suggests films based on individual preferences. However, for many users, the underlying processes—whether for individual or group recommendations—remain opaque and difficult to understand. This is particularly true for group recommender systems, which aggregate the preferences of multiple users to suggest relevant items. As the complexity of these systems increases, the challenge of explaining how recommendations are generated becomes more difficult, leading to potential mistrust or frustration, especially when the system fails to meet user expectations [1].

To generate explanations for recommendations, systems typically rely on information from all users within the system. This requires access to system data to clarify why a particular item is recommended. However, service providers are often concerned with maintaining client privacy and security, meaning that third parties cannot access or be informed about this sensitive data. One solution to this challenge is the use of counterfactual explanations, which only leverage the interactions of the specific user requesting the explanation, addressing privacy concerns.

Counterfactual explanations have been previously explored [2]. Such explanations are still influenced by the underlying recommendation system and thus cannot be universally applied across all recommendation models. A model-agnostic approach to counterfactual explanations, presented in [3], works by altering the user's interactions with the system and repeatedly querying the recommendation engine. In this method, some items are systematically removed from the user's history, and the recommendation system is called again to check if the item in question is still suggested. If the item is removed, the set of items responsible for that change is provided as the explanation. This results in an explanation such as: "*If you had not liked item A, item B would not have been recommended*", where A is a set of removed items, and B is the item being explained.

Building on this approach, we focus on model-agnostic counterfactual explanations, which are becoming increasingly important. Many systems either do not provide access to their underlying recommendation models or involve models too complex for the average user to interpret. In this paper, we explore how counterfactual explanations can be generated for groups of users, rather than just for individual users. Group recommendations are a popular research field since it is now easier than ever to organize and participate in group activities (e.g., [4, 5]). However, group recommendations are inherently more complex since we must consider multiple and diverse interests. This naturally suggests that providing explanations for these recommendations is also more complex. An additional layer of complexity in group recommendations arises when we need to ensure that all group members are treated equally. This means that the explanation should not highlight individual members but should instead include items that have been interacted with by the majority of the group.

An additional problem for counterfactual explanations for group recommender systems is the large search space. Typically, users have multiple interactions with the system, and each such interaction is a probable part of a counterfactual explanation. Examining all possible combinations of items is not computationally feasible. Assuming that the user has given n ratings to the system, then possible counterfactual explanations are 2^n. For group recommenders, we multiply this by the number of group members.

Since exhaustively searching through all combinations of items is prohibited, we propose two heuristic approaches. We leverage utility scores (Section 2.2) to quickly parse through a large number of items. The first approach, *Sliding Window* (Section 3.1), aggregates all utility scores for the items and orders them in a list. We apply a sliding window on this list, aiming to quickly find a large subset of items that, if they were removed from the group members' interactions, then the item in need of explanation is no longer suggested to the group, i.e., a counterfactual explanation.

The second approach, *Four Quadrant* (Section 3.2), allocates the items based on their utility scores in a four-

DOLAP 2025: 27th International Workshop on Design, Optimization, Languages and Analytical Processing of Big Data, co-located with EDBT/ICDT 2025, March 25, 2025, Barcelona, Spain
*Corresponding author.
maria.stratigi@tuni.fi (M. Stratigi); bikakis@hmu.gr (N. Bikakis); konstantinos.stefanidis@tuni.fi (K. Stefanidis)
© 2025 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

quadrant graph, where the axis of the graph represents the four utility metrics: Item Intensity, Item Rating, Item Popularity, and Item Relevance (for more details please refer to Section 2.2). We only examine items that are present within a circular region. If no explanation is found, then we will increase the radius of the circle.

Overall, the main contributions of our work are the following:

- We introduce the notion of counterfactual explanations for group recommendation, which, to our knowledge, represents the first work in this field.
- We present group counterfactual utility metrics that measure the likelihood of an item being part of a counterfactual explanation.
- We propose two heuristic algorithms to find fair counterfactual explanations for groups, Sliding Window and Four Quadrant Algorithms.
- We evaluate our proposed methods using the real-world recommendation dataset, MovieLens, focusing on two different group types: those with high engagement levels with the system and those with moderate engagement.

The rest of the paper is structured as follows. Section 2 describes the group counterfactual explanation problem, and Section 3 details our two proposed heuristic methods for producing such explanations. Section 4 presents the experimental evaluation and analyzes the results. Section 5 presents the related work. Finally, Section 6 concludes this paper.

2. Group Counterfactual Explanations Problem

2.1. Preliminaries

Interacted Items. Let U be all the users involved in the system. Let I_u be the set of *items that the user* $u \in U$ *has interacted with*, and $I_G = \bigcup \forall_{u \in G} I_u$ be all the *items that the group members* $G \subseteq U$ *have interacted with.* Furthermore, $\rho(u, i) \in \mathbb{R}^+$ denote the *rating assigned to item* i by the *user* u. Note that, in our work the presence of ratings is not mandatory.

Recommendations. Additionally, $GRS(I_G)$ denote the *group recommendation system* which, given the set I_G, returns the *recommended items list* L^{I_G}, i.e., $GRS(I_G) = L^{I_G}$. For this work, we consider the group recommender system a black box. This allows us to freely use any group recommender system without considering their limitations.

Group Counterfactual Explanation. In a recommendation system for individual users, a counterfactual explanation about a recommended item i is defined as a set of items that if the user had not interacted with (e.g., rated, clicked, liked), then item i would not have been recommended. A similar concept can be defined in group recommendation systems.

Given a group $G \subseteq U$, and an item $t \in L^{I_G}$, which will be referred to as the *item of interest*, an explanation E for the appearance of the item t in the group recommendation list L^{I_G}, is a set of items, that if the group members had not interacted with, then the item of interest t would not have appeared in the group recommendation list L^{I_G}.

Formally, let L^{I_G} be a recommendation list, with $t \in L^{I_G}$; an item set $E \subseteq I_G$ is an *explanation*, if $t \notin L^{I_G \setminus E}$.

Counterfactual Cost. Producing the recommendations list L^{I_G}, is associated with a *recommender cost* ϕ, e.g., the response time and/or service cost. As a result, a counterfactual E is also associated with a *cost*, denoted as E_{cost}, which corresponds to the overall recommender cost required to find E. Hence, given that we need to invoke the recommender *n times* in order to find E, the *counterfactual cost* of E is:

$$E_{cost} = \sum_{1}^{n} \varphi \tag{1}$$

where φ represents the cost of a single recommender system call. For simplicity, in our evaluation we define $\varphi = 1$ to encapsulate how many times we called on the recommender system.

2.2. Group Counterfactual Utility

In order to access the "quality" of the counterfactual, we exploit several metrics related to: clarity, engagement, preferences, popularity, and relevance.

Counterfactual Minimality. We use the notion of *minimality* (a.k.a. *sparsity*) to access the clarity of a counterfactual, i.e., the shorter the explanations, the easier it is to understand [6, 7]. In our problem, the minimality $\mathsf{mini}(E) \in [0, 1]$ of a counterfactual E is quantified by the number of items included in E normalized by the overall number of interacted items $|I_G|$:

$$\mathsf{mini}(E) = \frac{|E|}{|I_G|} \tag{2}$$

Note that, in our problem the adopted minimality metric also captures the *actionability* (a.k.a. *feasibility*) of a counterfactual [8]. Intuitively, a counterfactual is more feasible if fewer number of changes occur.

Item Intensity. *Item Intensity* describes how many of the group members had interacted with an item. The intensity of an item reflects the degree to which it has been interacted with by individual members of the group. This characteristic provides insights into the extent of engagement that group members have had with a particular item, which in turn sheds light on the potential significance of that item within the context of the group. Higher intensity values suggest that the item has garnered substantial attention from group members. Hence, *we want the explanation items to have high intensity.*

Formally, given a group $G \subseteq U$ and an item $i \in I_G$, the *intensity of an item* $\mathsf{int}(G, i) \in (0, 1]$ is defined as:

$$\mathsf{int}(G, i) = \frac{\sum_{\forall u \in G} r(u, i)}{|G|} \tag{3}$$

where $r(u, i)$ returns 1 if user u has interacted with item i (i.e., $i \in I_u$), and 0 otherwise.

Item Rating. This metric builds upon Item Intensity metric by incorporating the collective evaluation of an item by all group members. Specifically, the *Item Rating* for a *group* G and an *item* i, denoted as $\mathsf{rate}(G, i)$, is computed as the *mean of the ratings assigned to an item* i *by members of the group* G.

$$\mathsf{rate}(G, i) = \frac{\sum_{\forall u \in G} \rho(u, i)}{|G|} \tag{4}$$

This metric offers a clear representation of the group's overall sentiment toward the item, providing an aggregated view of the group's preferences. Note that, in case that items ratings are not available the metric is omitted.

Item Popularity. Item Popularity refers to an item's overall appeal or recognition within all systems users U. *Item Popularity* for an *item* i, denoted as $\text{pop}(U, i)$ is quantified by *aggregating the ratings the item has received across all users.*

$$\text{pop}(U, i) = \sum_{\forall u \in U} \rho(u, i) \tag{5}$$

Popularity serves as a proxy for how widely appreciated or commonly engaged with an item is, regardless of individual user preferences. It is a model-agnostic metric, allowing its straightforward adoption across different recommendation systems and environments. Note that, in case that items ratings are not available the metric is omitted.

Item Relevance. Item Relevance measures how pertinent an item in I_G is. This is based on the predicted score for the *item of interest* t assigned to group members from the recommender system. Intuitively, the interacted items of a user u with a high predicted score for t are more influential in shaping the group recommendation list. Consequently, these items are more likely to be included in the counterfactual explanation. To compute *item relevance* $\text{rel}(G, i, t)$ in a model-agnostic manner, we treat each individual user as a group and query the group recommender for suggestions based on their preferences and interactions.

$$\text{rel}(G, i, t) = \sum_{\forall u \in G} h(u, i, t) \tag{6}$$

where

$$h(u, i, t) = \begin{cases} \text{predScore}(L^{I_u}, t) & \text{if } i \in I_u \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

The function $\text{predScore}(L^{I_u}, t)$ returns the predicted score of the item of interest t for user u, where L^{I_u} is the recommendation list resulted when the group recommender is invoked using as input only u's interacted items, i.e., I_u.

Counterfactual Utility. Given a counterfactual E the *utility* $U(E)$ is defined as a combination of *counterfactual minimality*, and items *intensity*, *rating*, *popularity*, and *relevance* metrics:

$$U(E) = f(\text{mini}(\,),\ \theta_{\forall i \in E}(\text{int}(\,), \text{rate}(\,), \text{pop}(\,), \text{rel}(\,))) \tag{8}$$

where function f *aggregates counterfactual minimality with item-based scores*, and function θ *aggregates the items scores for all items in* E.

2.3. Problem Definition

Next, we formally define the the *Group Counterfactual Explanation* problem (GCF).

Group Counterfactual Explanation Problem (GCF). Given a *group* G; a *group interacted items* I_G; a *group recommended items list* L^{I_G}; a *item of interest* i; a *recommender cost* φ; and a *budget* B in terms of recommender cost; our goal is to find a *group counterfactual explanation* E^*, such that the *explanation utility* $U(E^*)$ is maximized and the *counterfactual cost* E^*_{cost} is lower than the budget B.

$$E^* = \arg\max U(E) \quad \text{s.t.} \quad E^*_{cost} \leq B$$

Computational Complexity. In order to solve the GCF problem we have to examine the power set of the group interacted items I_G, i.e., all the possible group interacted items subsets. That is, we have to examine $2^{|I_G|-1}$ sets (empty set is omitted). Considering that the recommender cost for each set is ϕ, the computational complexity is $O(\phi \cdot 2^{|I_G|})$.

2.4. Group Counterfactual Fairness

In this work, in addition to counterfactual utility, we also examine the notion of fairness in group counterfactual explanations. Specifically, we define fairness in terms of how equally group members have interacted with the items presented in the explanation. The goal is for each member of the group to have engaged with the same set of items, ensuring that no individual is singled out, and the explanation reflects the experiences of the entire group.

Group Counterfactual Fairness. Given a group $G \subseteq U$ and an explanation E, the *fairness of the explanation* E, $\text{fair}(G, E) \in (0, 1]$ is defined as follows. The higher the value, the fairer the explanation.

$$\text{fair}(G, E) = \frac{\sum_{\forall u \in G} z(u, E)}{|G|} \tag{9}$$

$$z(u, E) = \begin{cases} 1 & \text{if } \exists i \in E \text{ and } i \in I_u \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

Equation 9 calculates the extent of group members that have interacted with at least one item in the explanation. Ideally, we want all group members to have interacted with at least one item, i.e., $\text{fair}(G, E) = 1$. On the other hand, if $\text{fair} = 1/|G|$, it suggests that the explanation is focused on one member of the group, leaving the others unaccounted for.

3. Algorithms

Due to the large search space (i.e., $2^{|I_G|}$), finding an optimal solution is computationally prohibitive even for a small number of users. To cope with the complexity of the GCF problem, we design two efficient heuristic for finding group counterfactual explanations.

3.1. Sliding Window Algorithm (SW)

The first approach we introduce employs a sliding window technique applied to an ordered list of items. The *Sliding Window* (SW) algorithm operates as follows: for each item that has been interacted with by at least one group member, we compute the item-based metrics included in the utility formula (Eq. 8). These scores are aggregated into a final score for each item, enabling the ordering of items from highest to lowest score.

Once the items have been ordered, we apply a sliding window of fixed size to the items list. For instance, with a window size of 5, the window initially covers items 1 through 5, then shifts one position forward to cover items 2 through 6, and so on, until the entire list has been processed. At each step, the items within the current window are temporarily excluded from the group interacted items, and the

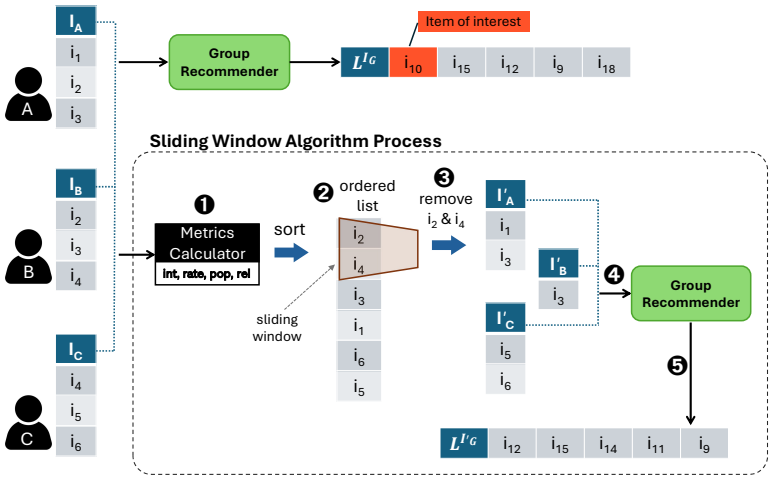

Figure 1: Sliding Window Algorithm Example, with the size of the window set to 2. On the left, we have a group of three users, A, B, and C, along with their corresponding lists of items they have interacted with: I_A, I_B, and I_C. Initially, a group recommender system receives these lists and produces the group recommendation list L^{I_G}, where $I_G = I_A \cup I_B \cup I_C$, with the first item relegated to be the target item. The Sliding Window Algorithm starts by calculating and aggregating the four Item Utilities: Item Intensity, Item Rating, Item Popularity and Item Relevance (step 1). Since the size of the window is set to 2 we select the top 2 items in the produced ordered list (step 2) and remove them from the group members' interactions (step 3). Lastly we call on the group recommender system again with the altered interaction lists (step 4) and check if the target item has been removed from the group recommendation list (step 5).

group recommender system is invoked to compute the new recommendations. If the item of interest is not included in the recommendation list, the items included in the window correspond to a valid counterfactual explanation.

When the window items correspond to a counterfactual and in order to improve minimality, we search for a potentially smaller set of items that also qualifies as a counterfactual. To this end, we conduct an exhaustive search across all possible combinations of items within the window, evaluating which subset of items best generates the counterfactual explanation. This search is performed in an ascending order based on the number of items being considered, starting with individual items and progressively expanding to larger combinations. In this manner, the first valid counterfactual explanation that is discovered will be the smallest possible subset of items, thus ensuring that the explanation is both compact and precise. We have selected to keep the window size small in order to minimize the number of possible combinations that we have to examine.

Sliding Window Algorithm Example. Figure 1 demonstrates the sliding window process. Assume a group of three members, A, B, and C, each has their own interacted items sets: I_A, I_B, and I_C, respectively. These interacted items list are given to a group recommender system which outputs a group recommendation list L^{I_G}. Let's assume that the item of interest (i.e., the item in need of explanation) is i_{10}. ❶ We consider all the interacted items of the group members and calculate the item-based scores for each item. ❷ Then, we aggregate these scores into a final score and order the list in descending order based on that score. ❸ Next, apply a sliding window of size 2, which will initially remove from group members interacted items lists the items included in the window, i.e., the first two items, i_2 and i_4. In our example, initially we have $I_A = i_1, i_2, i_3$ and the updated interacted items list I'_A is $I'_A = i_1, i_3$. ❹ Finally, we invoke the recommender given as input the updated interacted items lists I'_A, I'_B and I'_c. ❺ The resulted group recommendation list $L^{I'_G}$ do not include the item of interest i_{10}, so the items with the window $\{i_2, i_4\}$ corresponds to a group counterfactual.

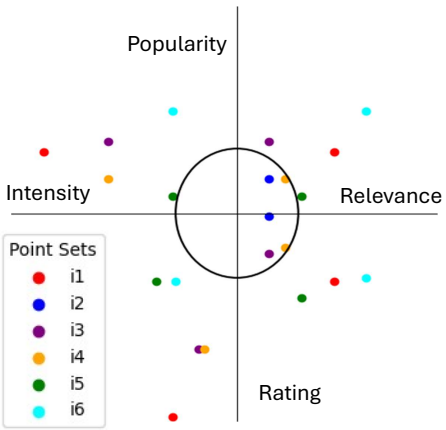

Figure 2: Four Quadrants Algorithm Example. The algorithm first calculates the four Item metrics: Item Intensity, Item Rating, Item Popularity, and Item Relevance, and allocates the items in a four-quadrant graph based on these values as coordinates. A circle with the axis origin as its center and a radius of r determines which items will be examined.

3.2. Four Quadrants Algorithm (Quad)

In *Four Quadrants Algorithm* (Quad) we conceptualize the item-based metrics used in the utility score (Eq. 8), as the axes of a quadrant graph, where each axis corresponds to one of the metric, as illustrated in Figure 2. This representation enables the pairing of the metrics into four pairs:

1. Popularity - Intensity
2. Popularity - Relevance
3. Relevance - Rating
4. Rating - Intensity

To position the items on the graph, we use their respective scores for each metric. Each item is thus represented

four times, corresponding to its metrics scores along the respective axis of the graph. Given that the metrics have significantly different numerical ranges, such as popularity being represented by a large number and Item Intensity being a score constrained within the range (0,1], it is essential to normalize all the values to a common scale. Consequently, we apply a normalization process to ensure that each score is transformed into the range [0, 1].

We examine items located within a circular region centered at the origin of the quadrant graph (i.e., the point representing the normalized value 0 for each characteristic). The radius of this circle, denoted as r, determines the area of interest. Items that are considered most relevant, such as those with higher popularity, should ideally be positioned closer to the center of the graph. To achieve this, we adjust the item scores by subtracting each score from 1, thereby making higher scores closer to the center. This transformation ensures that more significant values, such as higher popularity scores, will correspond to positions closer to the origin of the quadrant graph.

In a manner similar to the previous approach, we remove from consideration those items that fall within the circular region defined by the radius r. After adjusting the dataset accordingly, we call the group recommender with the modified information to determine if the item of interest remains in the group recommendation list. If the item of interest is still included, we incrementally increase the radius r and repeat the process. This procedure continues until the item of interest is no longer recommended, at which point we have identified a subset of items that provide a counterfactual explanation.

When a counterfactual explanation is identified, we follow an approach akin to the previous one, wherein we systematically explore all possible combinations of items within the subset. This search is conducted in ascending order of the combination size to find the smallest possible subset that serves as the counterfactual explanation.

However, as previously noted, performing an exhaustive search becomes computationally prohibitive when the number of items is large. To mitigate this, we leverage the fact that items can appear multiple times within the circular region. Specifically, instead of evaluating all items, without the loose of generality, we restrict our search to those items that appear at least twice within the circle. This restriction significantly reduces the search space, enabling us to perform the exhaustive search in a feasible manner. For example, in Figure 2, we have a circle with $r = 0.2$, for which only two items, i_2 and i_4 are present twice. So, we will only examine all possible combinations of those two items, namely: $[\{i_2\}, \{i_4\}, \{i_2, i_4\}]$.

4. Experimental Analysis

In this section, we present the experimental results, evaluating the proposed algorithms: *Sliding Window* (SW) and *Four Quadrants* (Quad), varying several parameters and comparing them against a baseline method. In the first experiment (Sect. 4.2) we examine the counterfactual cost produced by the algorithms. In the next experiment (Sect. 4.3) we evaluate the group counterfactual fairness, and finally examine the sizes of the explanations (Sect. 4.4).

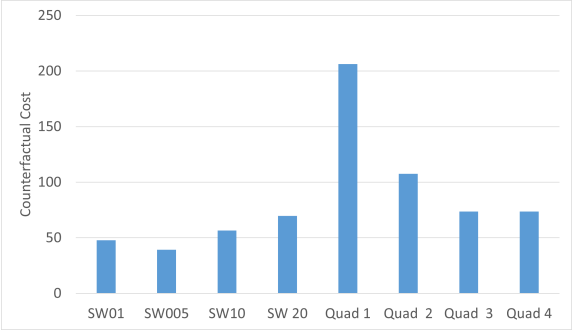

Figure 3: Counterfactual cost [High-engagement groups]. The average times each algorithm needed to call the group recommender system until it produced a group counterfactual explanation.

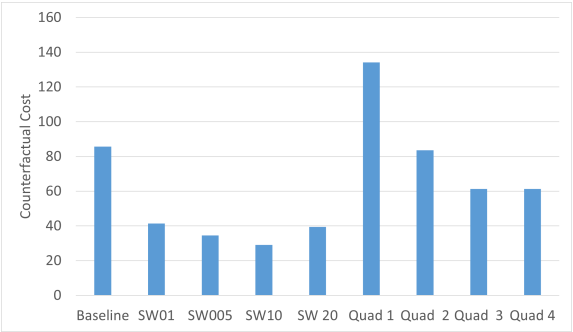

Figure 4: Counterfactual cost [Moderate-engagement groups]. The average times each algorithm needed to call the group recommender system until it produced a group counterfactual explanation.

4.1. Experimental Setup

Methods Parameters. For *SW algorithm*, we examine two window sizes (i.e., percentages based on the size of the list of items with which at least one group member has interacted). Particularly, we consider two percentages: 10% (SW01) and 5% (SW005). Furthermore, in SW evaluation, we examine two fixed window sizes: ten (SW10) and twenty (SW20). Regarding *Quad algorithm*, we conduct experiments varying the number of times items appear inside the circle, setting this parameter to 1, 2, 3, and 4, denoted as Quad1, Quad2, Quad3, and Quad4, respectively. The starting value of circle radius r is 0.1. For comparison, we also consider the *Baseline* approach, where we exhaustively examine all possible combinations of items that at least one group member has interacted with.

Dataset. We utilized the MovieLens 100K dataset [9], which contains 100K ratings from 600 users on 9K movies. We compose groups based on two types of users: (1) user with a high number of ratings (more than 400), and (2) users with a moderate number of ratings (between 100 and 300). We refer to these groups as *high-engagement* and *moderate-engagement*, respectively. For each type of group, we analyze 20 different groups.

Group Recommendation System. The experiments were conducted in Python, using the Surprise library to create collaborative filtering models and the SciPy library for optimization and batch searches. We employed the k-nearest

Figure 5: Fairness [High-engagement groups]. The average number of group members who interacted with at least one item in the group counterfactual explanation.

neighbors (KNN) technique with user-based collaborative filtering to generate recommendations for individual users. To create group recommendations, we utilized one of the most popular group recommendation models, aggregating the single recommender system's predicted scores of each item for all group members. The aggregation function we utilized was the Average. The group recommendation list L^{I_G} consists of the ten items with the highest aggregated scores. For simplicity, the item of interest, i.e., the item we want to explain, is always the first one in L^{I_G}.

The group recommendation model was retrained several times as new batches of movies were removed. This iterative process of batch elimination and model retraining was essential in simulating realistic scenarios where recommendations adapt based on user feedback or potential counterfactual situations.

4.2. Cost Evaluation

We first analyzed the computational cost required to find a counterfactual explanations. This evaluation was conducted across 40 groups, evenly split between moderate- and high-engagement groups (20 each). The cost was quantified as the average number of calls to the group recommender system required to find a counterfactual explanation. Figures 3 and 4 illustrate these costs for high- and moderate-engagement groups, respectively.

For high-engagement groups, the Baseline approach was excluded due to its prohibitively high cost. This high cost arises from the fact that the Baseline approach will first examine individual items and then item combinations, making the approach impractical since the list for all possible items we want to examine, I_G, is quite large. The size of I_G is approximately 2.000 since we have five members, and each one has given more than 400 ratings.

Sliding Window Algorithm (SW). The Sliding Window algorithm emerged as the most efficient in minimizing the cost of counterfactual explanation generation. Its performance varied between moderate- and high-engagement groups. For moderate-engagement groups (Fig. 4), the best configuration was SW10, which uses a fixed window size of 10. This configuration is effective because the smaller size of the item list in these groups ensures that ten items are sufficient to identify the counterfactual explanation efficiently.

In contrast, for high-engagement groups (Fig. 3), the best approach was SW005, where the window size is dynamically set to 5% of the total number of items in I_G. This dynamic adjustment accommodates the significantly larger sized I_G lists in high-engagement groups, often more than three times the size of those in moderate groups. In such cases, a fixed window size of 10 proved insufficient, requiring additional calls to the recommender system to locate the counterfactual explanation.

It is worth noting that larger window sizes, such as those used in SW01 and SW20, reduced the time to locate a counterfactual window, i.e., a subset of items that can produce a counterfactual explanation, but resulted in higher overall costs. This was due to the inclusion of numerous items in the window, necessitating additional calls to the recommender system to find the minimal counterfactual explanation.

Four Quadrants Algorithm (Quad). The Four Quadrants algorithm exhibited a different cost profile. Among its configurations, Quad 1 incurred the highest cost for both moderate- and high-engagement groups. This can be attributed to the algorithm's consideration of all items within the circular region without any filtering, resulting in an extensive initial set of items to examine.

Costs decreased as stricter criteria were applied, such as requiring items to appear multiple times within the circular region before being considered. This observation supports our hypothesis that filtering items based on repeated inclusion is essential for cost reduction. Both Quad 3 and Quad 4 demonstrated comparable performance, effectively filtering out outlier items, such as those with low popularity or those rated by a single group member, thereby reducing computational overhead.

Discussion. The Sliding Window algorithm demonstrated superior efficiency in identifying counterfactual explanations with lower number of calls to the group recommender system. To optimize performance, the window size must balance being large enough to locate the counterfactual window quickly while avoiding excessive inclusion of items, which would increase costs. The dynamic configuration of SW005 proved particularly effective, adapting to varying group engagement levels.

The Four Quadrants Algorithm, while less adaptable, exhibited consistent performance across group types: high and moderate engagement. Quad 3 and Quad 4 were especially effective in filtering out irrelevant items, ensuring steady computational costs across different engagement levels. These results highlight the trade-offs between adaptability and reliability, underscoring the need for tailored algorithmic designs to accommodate varying group dynamics in counterfactual explanation generation.

4.3. Fairness Evaluation

In this experiment, we examine the impact of metrics and methods on achieving counterfactual fairness. Figures 5 and 6 present the fairness metric defined in Equation 9, which measures the proportion of group members who have interacted with at least one item in the counterfactual explanation. Higher fairness metric values indicate better performance in ensuring inclusivity across group members. Ideally, all group members would have interacted with every item in the explanation. However, achieving that is challenging due to the diversity of interactions within the group.

The Sliding Window algorithm demonstrated superior performance compared to the other approaches, primarily

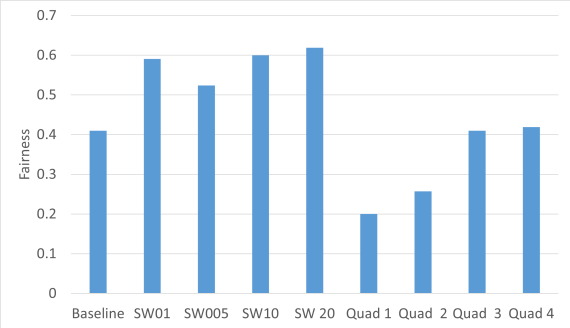

Figure 6: Fairness [Moderate-engagement groups]. The average number of group members who interacted with at least one item in the group counterfactual explanation.

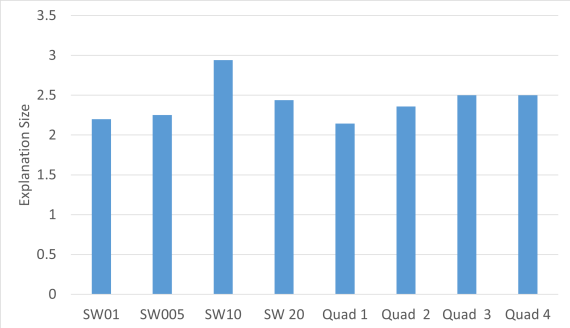

Figure 7: Explanation size [High-engagement groups]. The average number of items included in the counterfactual explanations generated by each algorithm.

due to its direct consideration of the number of group members interacting with an item. This is captured through the item intensity metric (Equation 3), where the score for an item increases with the number of group members who have interacted with it. By ranking items based on their combined scores, the Sliding Window algorithm prioritizes items with high intensity—those that multiple group members have interacted with—early in the counterfactual explanation generation process. This prioritization inherently improves fairness, as it ensures greater representation across the group.

In contrast, the Four Quadrants algorithm struggles to achieve comparable fairness levels. This arises from the way item scores are scaled. High-intensity items, such as those interacted with by four out of five group members, receive an item intensity score of 0.8. To align with the algorithm's scoring framework, this value is reversed by subtracting it from 1, resulting in a score of 0.2. Consequently, these items are not considered, as their scores fall above the initial radius threshold r value, which is set to 0.1.

Moreover, for an item to be included in the counterfactual explanation, it must meet additional criteria, such as achieving sufficient counterfactual utility scores. For example, the item must exhibit both high popularity within the system and relevance within the group. These stringent requirements further limit the Four Quadrants algorithm's ability to prioritize high-intensity items, ultimately leading to lower fairness scores compared to the Sliding Window algorithm.

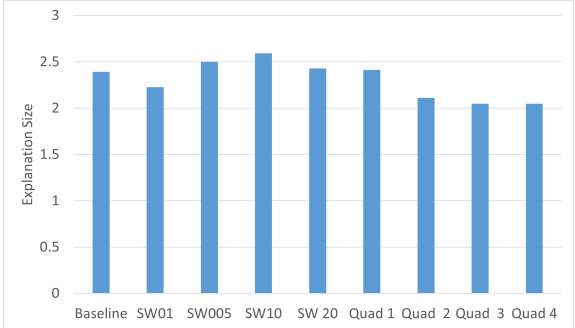

Figure 8: Explanation size [Moderate-engagement groups]. The average number of items included in the counterfactual explanations generated by each algorithm.

4.4. Explanation Size Evaluation

Figures 7 and 8 showcase the size (number of items) of the explanations generated by the algorithms. Under our defined group counterfactual utility, we want the explanations provided to be minimal, i.e., to consist of as few items as possible. The Four Quadrant algorithm manages to find explanations with the least amount of items. This demonstrates the algorithm's ability to identify the most "effective" explanations. We quantify these explanations as effective since they manage to alter the initial group recommendation list by removing the least amount of interactions from the group members.

We can observe that the explanation size does not increase with the number of interactions among the group. The sizes remain in the same order for both high and moderate-engagement groups as presented in Figures 7 and 8. However, the cost of finding these items is higher the more interactions the group has. As shown in Figures 3 and 4, to find counterfactual explanations for high-engagement groups, the algorithms require more calls to the group recommender system, indicating the difference in search space size.

5. Related Work

5.1. Group Recommendations

Group recommendations have a significant research background [5]. The most popular approach for producing group recommendations is to employ a standard single-user recommendation system and apply it to each individual group member (e.g., [10, 11, 4, 12]). Then, we aggregate the group members lists into one single group recommendation list.

In the aggregation stage, a group recommendation system can take into account different criteria. For example, [13] suggests a group recommendation model which considers each individual group member's influence during the aggregation phase. Differently, [14] exploits ideas from attention network and neural collaborative filtering to deduce the aggregation strategy from the available data. Similarly, [15] in addition to an attention mechanism, it also employs a bipartite graph embedding model to infer each member's influence to the group's final choice, while [16] uses a social network enhanced with user preferences and the social interactions among the group members, to locate the group's choices.

[17] considers the interactions between group members to determine the best aggregation strategy. These interac-

tions were modeled as multiple voting processes in order to simulate how a consensus is reached, and a stacked social self-attention network was proposed to learn the voting scheme of the group members. In dividing a large group of people into subgroups based on their own interests, [18] offers a novel method of producing recommendations for a large group. Specifically, it identifies a set of potential candidate media-user pairs for each subgroup and aggregate the CF recommendations lists for each such pair. [19] proposes a two-phase group recommender that targets to satisfy all the group members. In the first phase, it tries to satisfy the whole group, and in the second phase, it tries to satisfy the members individually by filtering out irrelevant items to each member.

In [20], each group member is assigned a utility score based on how relevant the recommended items are to them. Then it balances the utility of the group members and generates a group recommendation list. In [21], the utility of a user is defined by the similarity between the individual and group recommendations of the user. Their approach involves considering sets of N-level Pareto optimal items when creating the group recommendation list. As part of the aggregation phase, [22] proposes a notion of rank-sensitive balance. As far as possible, the first recommendation should balance the interests of all group members. Similarly, the first two items together must also do the same, and so on.

5.2. Counterfactual Explanations

Counterfactual explanations have been a potent tool to improve recommendation systems' transparency and reliability in recent years. These explanations clarify why certain recommendations were made by showing how little adjustments to the input data could have a significant impact on the results. This method is especially relevant to group recommender systems, in which a recommendation is generated by combining the preferences of several individuals. For the group's recommendations to be understood by all members and how their preferences influenced their final decision, transparency is extremely crucial.

By altering image embeddings to ascertain whether visual aspects impact recommendations, CAVIAR [23] proposes counterfactual explanations for visual recommender systems. Similarly, Wang et al. [24] addressed the biases in recommendation systems related to demographic factors like age and gender by introducing CFairER, a model for fairness-aware counterfactual explanations. This approach may be crucial when recommendations are made, and fairness among members of different backgrounds is an issue. Graph-based methods have also been explored in search of counterfactual explanations. GNNUERS, a counterfactual reasoning approach to explain fairness problems in GNN-based recommendation systems, is presented in [25].

Additionally, Stratigi et al. [26] covered "why-not" questions in collaborative filtering, offering explanations for items that were not recommended. In recommender systems, when some group members could wonder why their preferred items weren't included in the recommendation, this kind of explanation could be crucial. This idea was expanded to graph-based recommender systems by Attolou et al. [27], who offered "why-not" explanations for items that were left out of recommendations. CETD technique [28] generates counterfactual explanations for sequential recommender systems by taking into account temporal relationships in user behavior. This method can be used in settings where members' preferences may vary over time and require explanations that take those changes into account.

A recent study [29] presents a framework for producing explainable counterfactual recommendations, which explain why a certain item was recommended as well as how small adjustments could have produced alternative results. Similar to this, Yao et al. [30] suggested a learning approach that focuses on user interaction data to customize explanations for counterfactual explanations in recommendations.

MACER [31] is a reinforcement learning-based, model-agnostic counterfactual explanation paradigm. It is especially well suited for group recommendation systems, where several models may be used to aggregate user preferences, as it produces item-based explanations that are relevant to any recommendation system. LXR [32] proposes a post-hoc, model-agnostic approach to counterfactual explanations that utilizes self-supervised learning to identify the most critical user interactions with respect to a recommended item. [33] explore attribute-level counterfactual explanations for Heterogeneous Information Networks (HINs)-enhanced recommendations. The focus is on providing clear explanations regarding the disparities in item exposure. Their approach aims to foster fair allocation of items that are preferred by users but currently receive less visibility.

These investigations laid the groundwork for expanding counterfactual explanations into the field of group recommender systems. However, the majority of research to date has been conducted on individual recommendations. In contrast, we examine how to produce counterfactual explanations through the lens of group recommendations, something that, to our knowledge, has not been researched up to now. Group recommendations introduce a new layer of complexity to the problem, and it is essential to reconcile the varying preferences of group members with transparent, implementable counterfactual explanations that guarantee equity and satisfaction for each individual concerned.

6. Conclusion

This paper proposes two heuristic approaches to finding counterfactual explanations for group recommendations. Since many probable combinations of items can constitute a group counterfactual explanation, we define group counterfactual utilities to help us locate an explanation in a reasonable time. Our two proposed algorithms utilize these utility scores in a different manner. The Sliding Window algorithm aggregates the scores and orders the items in a descending order. Then, it applies a sliding window on the list and checks if the items in the window are a counterfactual explanation. If they are, the algorithm exhaustively searches all combinations of items in the window to find the most concise explanations, i.e., the explanation consisting of the least number of items. The Four Quadrant algorithm arranges the items based on their utility scores in a four-quadrant graph. Then, check the items that are located inside a circular area. Similarly to Sliding Window, when it finds a subset of items that produce a counterfactual explanation, it exhaustively searches for the minimal explanation. We evaluate our proposed methods utilizing a real work dataset, MovieLens, for various methods' parameters. Our results show that the Sliding Window can produce the explanations faster. However, the Four Quadrant algorithm is not overly affected by the increase in the size of the items it needs to search.

References

[1] G. Giannopoulos, G. Papastefanatos, D. Sacharidis, K. Stefanidis, Interactivity, fairness and explanations in recommendations, in: Adjunct Publication of the 29th ACM Conference on User Modeling, Adaptation and Personalization, UMAP 2021, Utrecht, The Netherlands, June 21-25, 2021, ACM, 2021, pp. 157–161.

[2] A. Ghazimatin, O. D. Balalau, R. S. Roy, G. Weikum, PRINCE: provider-side interpretability with counterfactual explanations in recommender systems, in: WSDM, 2020.

[3] V. Kaffes, D. Sacharidis, G. Giannopoulos, Model-agnostic counterfactual explanations of recommendations, in: Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization, UMAP '21, Association for Computing Machinery, New York, NY, USA, 2021, p. 280–285.

[4] E. Ntoutsi, K. Stefanidis, K. Nørvåg, H. Kriegel, Fast group recommendations by applying user clustering, in: Conceptual Modeling - 31st International Conference ER, volume 7532 of *Lecture Notes in Computer Science*, 2012, pp. 126–140.

[5] J. Masthoff, Group recommender systems: Aggregation, satisfaction and group attributes, in: F. Ricci, L. Rokach, B. Shapira (Eds.), Recommender Systems Handbook, Springer, 2015, pp. 743–776.

[6] T. Miller, Explanation in artificial intelligence: Insights from the social sciences, Artif. Intell. 267 (2019).

[7] S. Wachter, B. D. Mittelstadt, C. Russell, Counterfactual explanations without opening the black box: Automated decisions and the GDPR, Harvard Journal of Law & Technology 31 (2018).

[8] R. Guidotti, Counterfactual explanations and how to find them: literature review and benchmarking, Data Min. Knowl. Discov. 38 (2024).

[9] F. M. Harper, J. A. Konstan, The movielens datasets: History and context, ACM Trans. Interact. Intell. Syst. 5 (2015) 19:1–19:19.

[10] S. Amer-Yahia, S. B. Roy, A. Chawlat, G. Das, C. Yu, Group recommendation: Semantics and efficiency, PVLDB 2 (2009) 754–765.

[11] L. Baltrunas, T. Makcinskas, F. Ricci, Group recommendations with rank aggregation and collaborative filtering, in: RecSys, 2010.

[12] M. Stratigi, E. Pitoura, J. Nummenmaa, K. Stefanidis, Sequential group recommendations based on satisfaction and disagreement scores, J. Intell. Inf. Syst. 58 (2022) 227–254.

[13] Q. Yuan, G. Cong, C.-Y. Lin, Com: A generative model for group recommendation, in: KDD, 2014.

[14] D. Cao, X. He, L. Miao, Y. An, C. Yang, R. Hong, Attentive group recommendation, in: SIGIR, 2018.

[15] H. Yin, Q. Wang, K. Zheng, Z. Li, J. Yang, X. Zhou, Social influence-based group representation learning for group recommendation, in: ICDE, 2019.

[16] A. Salehi-Abari, C. Boutilier, Preference-oriented social networks: Group recommendation and inference, in: RecSys, 2015.

[17] L. Vinh Tran, T.-A. Nguyen Pham, Y. Tay, Y. Liu, G. Cong, X. Li, Interact and decide: Medley of sub-attention networks for effective group recommendation, in: SIGIR, 2019.

[18] D. Qin, X. Zhou, L. Chen, G. Huang, Y. Zhang, Dynamic connection-based social group recommendation, IEEE TKDE (2018).

[19] J. K. Kim, H. K. Kim, H. Y. Oh, Y. U. Ryu, A group recommendation system for online communities, International Journal of Information Management (2010).

[20] L. Xiao, Z. Min, Z. Yongfeng, G. Zhaoquan, L. Yiqun, M. Shaoping, Fairness-aware group recommendation with pareto-efficiency, in: RecSys, 2017.

[21] D. Sacharidis, Top-n group recommendations with fairness, in: SAC, 2019.

[22] M. Kaya, D. Bridge, N. Tintarev, Ensuring fairness in group recommendations by rank-sensitive balancing of relevance, in: RecSys, 2020.

[23] N. Jain, V. Sharma, G. Sinha, Counterfactual explanations for visual recommender systems, in: Companion Proceedings of the ACM Web Conference 2024, WWW '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 674–677.

[24] X. Wang, Q. Li, D. Yu, Q. Li, G. Xu, Counterfactual explanation for fairness in recommendation, ACM Transactions on Information Systems 42 (2024). URL: https://doi.org/10.1145/3643670. doi:10.1145/3643670.

[25] G. Medda, F. Fabbri, M. Marras, L. Boratto, G. Fenu, Gnnuers: Fairness explanation in gnns for recommendation via counterfactual reasoning, ACM Transactions on Intelligent Systems and Technology (2024).

[26] M. Stratigi, K. Tzompanaki, K. Stefanidis, Why-not questions & explanations for collaborative filtering, in: Z. Huang, W. Beek, H. Wang, R. Zhou, Y. Zhang (Eds.), Web Information Systems Engineering – WISE 2020, volume 12343 of *Lecture Notes in Computer Science*, Springer, Cham, 2020, pp. 321–336. URL: https://doi.org/10.1007/978-3-030-62008-0_21. doi:10.1007/978-3-030-62008-0_21.

[27] H. Attolou, K. Tzompanaki, K. Stefanidis, D. Kotzinos, Why-not explainable graph recommender, in: 2024 IEEE 40th International Conference on Data Engineering (ICDE), IEEE Computer Society, Los Alamitos, CA, USA, 2024, pp. 2245–2257.

[28] M. He, B. An, J. Wang, H. Wen, Cetd: Counterfactual explanations by considering temporal dependencies in sequential recommendation, Applied Sciences 13 (2023) 11176. URL: https://doi.org/10.3390/app132011176. doi:10.3390/app132011176.

[29] J. Tan, S. Xu, Y. Ge, Y. Li, X. Chen, Y. Zhang, Counterfactual explainable recommendation, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, CIKM '21, Association for Computing Machinery, New York, NY, USA, 2021, pp. 1784–1793.

[30] Y. Yao, C. Wang, H. Li, Learning to counterfactually explain recommendations, ArXiv abs/2211.09752 (2022). URL: https://api.semanticscholar.org/CorpusID:253581408.

[31] Q. Wang, Reinforced model-agnostic counterfactual explanations for recommender systems, in: Proceedings of SPIE, volume 12700, 2023, pp. 127002V–127002V. doi:10.1117/12.2682249.

[32] O. Barkan, V. Bogina, L. Gurevitch, Y. Asher, N. Koenigstein, A counterfactual framework for learning and evaluating explanations for recommender systems, in: ACM Web Conference (WWW), 2024.

[33] X. Wang, Q. Li, D. Yu, Q. Li, G. Xu, Counterfactual explanation for fairness in recommendation, ACM Trans. Inf. Syst. 42 (2024).