Explainable Fairness in Recommendation

Yingqiang Ge
Rutgers University
yingqiang.ge@rutgers.edu

Juntao Tan
Rutgers University
juntao.tan@rutgers.edu

Yan Zhu
Meta Platforms, Inc.
yzhu@fb.com

Yinglong Xia
Meta Platforms, Inc.
yxia@fb.com

Jiebo Luo
University of Rochester
jluo@cs.rochester.edu

Shuchang Liu
Rutgers University
shuchang.liu@rutgers.edu

Zuohui Fu
Rutgers University
zuohui_fu@rutgers.edu

Shijie Geng
Rutgers University
shijie.geng@rutgers.edu

Zelong Li
Rutgers University
zelong.li@rutgers.edu

Yongfeng Zhang
Rutgers University
yongfeng.zhang@rutgers.edu

ABSTRACT
Existing research on fairness-aware recommendation has mainly focused on the quantification of fairness and the development of fair recommendation models, neither of which studies a more substantial problem—identifying the underlying reason of model disparity in recommendation. This information is critical for recommender system designers to understand the intrinsic recommendation mechanism and provides insights on how to improve model fairness to decision makers. Fortunately, with the rapid development of Explainable AI, we can use model explainability to gain insights into model (un)fairness. In this paper, we study the problem of explainable fairness, which helps to gain insights about why a system is fair or unfair, and guides the design of fair recommender systems with a more informed and unified methodology. Particularly, we focus on a common setting with feature-aware recommendation and exposure unfairness, but the proposed explainable fairness framework is general and can be applied to other recommendation settings and fairness definitions. We propose a Counterfactual Explainable Fairness framework, called CEF, which generates explanations about model fairness that can improve the fairness without significantly hurting the performance. The CEF framework formulates an optimization problem to learn the “minimal” change of the input features that changes the recommendation results to a certain level of fairness. Based on the counterfactual recommendation result of each feature, we calculate an explainability score in terms of the fairness-utility trade-off to rank all the feature-based explanations, and select the top ones as fairness explanations. Experimental results on several real-world datasets validate that our method is able to effectively provide explanations to the model disparities and these explanations can achieve better fairness-utility trade-off when using them for recommendation than all the baselines.

CCS CONCEPTS
• Computing methodologies → Artificial intelligence.

KEYWORDS
Explainable Fairness; Recommender Systems; Explainable Recommendation; Fairness in AI; Counterfactual Reasoning

ACM Reference Format:

1 INTRODUCTION
Nowadays, with the extensive deployment in various e-commerce platforms, recommender systems (RS) have been widely acknowledged for their strong capabilities of delivering high-quality services to users [19, 25, 27, 46, 66]. Despite these huge benefits, the issue of fairness in recommendation has also attracted considerable interests from both academia and industry [30, 42, 44, 55]. Fortunately, these concerns about algorithmic fairness have resulted in a resurgence of interest to develop fairness-aware recommendation models to ensure that such models would not become a source of unfair discrimination in recommendation [7, 17, 47, 77]. In the area of fairness-aware recommendation, existing research mainly focus on the quantification of fairness and the development of fair recommendation models. Fairness quantification aims to develop and investigate quantitative metrics that measure algorithmic disparities in ranking or recommendation [18, 21, 40]. For example, [18, 40] proposed and studied the recommendation quality fairness between active users and inactive users. Meanwhile, fair recommendation aims to find feasible algorithmic approaches that can adjust the recommendation results to reduce recommendation disparities. For example, [2, 24] proposed approaches to mitigating the popularity bias between different item groups.

Despite the great efforts on fairness-aware recommendation and possibly countless future emergence of discoveries, one fundamental question that has not been studied extensively yet is fairness diagnostics, i.e.,

• RQ What are the sources that result in model disparities in recommendation?

Considering the huge commercial and social values that recommender systems bring to various web platforms and the society, we believe that the answer to this RQ is critical for recommendation system designers to understand the intrinsic recommendation mechanism and to provide insights for decision makers on how to improve model fairness. Yet, the answer to this question turns out to be unsurprisingly challenging especially when the predictive
model is a large-scale deep black-box model with large numbers of input features. For example, it is hard to tell how input features (such as screen size, battery, camera) would influence the exposure unfairness. Note that some pioneer works in other areas have leveraged Explainable AI to seek for feature-based explanations for certain fairness outcome. For instance, Begley et al. used Shapley value to attribute the model disparity in classification [5, 50]. Though their methods successfully provide explanations to the model disparities in simple tasks, they are not suitable for recommender systems where the model inputs could be extremely large and sparse, which may bring huge computational cost when calculating Shapley value for each feature. Furthermore, existing methods only partially answers the above question, since they only explain either utility or fairness alone, ignoring the fact that there is an inherent trade-off between fairness and utility, which has been demonstrated by several recent work both empirically and theoretically [28, 33, 34, 44, 45, 71]. And this incomplete view may potentially downgrade the stringency of the method because explanations that have the same effect on model fairness may not have the same effect on model utility.

In this paper, we propose a novel framework to explain the recommendation (un)fairness based on a counterfactual reasoning paradigm. Particularly, we focus on a common setting with feature-aware recommendation and item exposure unfairness (popularity bias) [1, 2, 24] since feature-base explanations are more straightforward and easy to understand, which would be a great demonstration of the effectiveness of our method. However, the proposed approach is very general and can be applied to other recommendation settings with various fairness definitions. Specifically, we propose a Counterfactual Explainable Fairness (CEF) framework to generate feature-based explanations in terms of item exposure disparity for various black-box feature-aware recommendation models. We first follow prior works [13, 61, 74] to build a user-feature attention matrix as well as an item-feature quality matrix, and use both matrices to train a feature-aware recommendation model. Then, we aim to find the “minimal” changes to a given feature in the feature space that switch the recommendation results to a certain level of fairness. To avoid overwhelmed sacrificing of the recommendation quality, we also constrain the feature perturbation within a certain degree in the objective function. With the counterfactual learning objective and the perturbation constraint, our proposed framework is able to generate feature-level explanations that consider the fairness-utility trade-off. Finally, we calculate an explainability score in terms of the fairness-utility trade-off based on the counterfactual recommendation result of each feature. These scores help rank the feature-based explanations and we select the top ones as fairness explanations for the pre-trained recommendation model.

In general, the contributions of this work can be summarized as follows:

- We study the problem of explainable fairness in recommendation and propose a framework based on counterfactual reasoning. To the best of our knowledge, this is the first work that introduces explainable fairness in recommender systems.
- We design a learning-based intervention method to discover critical features that will significantly influence the fairness-utility trade-off and use them as fairness explanations for black-box recommendation systems.
- We conduct extensive experiments to evaluate our framework’s effectiveness and validate that explanations generated by CEF can achieve better fairness-utility trade-off when using them for recommendation than all the baselines.

2 RELATED WORK

There are several main research lines related to our work: explainable recommendation, fairness in recommendation and fairness explanation. We will briefly introduce each of them in this section.

2.1 Explainable Recommendation

Explainable recommendation has been an important topic in both academia and industry, which helps to improve the transparency, user satisfaction and trust over the recommender systems [73, 74]. Early approaches mainly attempt to make latent factor models explainable by aligning each latent factor with an explicit meaning such as item features [15, 74, 75]. Recently, with the ever prospering of deep learning technology, many neural algorithms are developed to explain recommendations based on neural models. For example, [51] proposed to attentively highlight particular words in user reviews as explanations, [16, 37] proposed to rank user review sentences as explanations, [14, 54] proposed visually explainable recommendation to highlight image regions or directly generate image as explanations, [9, 29, 36, 38, 39, 54] proposed to generate natural language explanations, [4] proposed set-based explanation for scrutability, [65] proposed contrastive explanations for comparison shopping, and [10, 11, 53, 64, 76] proposed neural-symbolic methods to improve both explainability and accuracy. In addition to text-based or image-based explainable recommendation, knowledge-aware explainable recommendation has also attracted research attention recently, such as [3, 18, 60, 63, 64].

Works using counterfactual reasoning to improve recommendation explainability [31, 56–58, 67] have been proposed very recently. Ghazimatin et al. [31] tried to generate provider-side counterfactual explanations by looking for a minimal set of user’s historical actions (e.g. reviewing, purchasing, rating) such that the recommendation can be changed by removing the selected actions. Xu et al. [67] proposed to improve this by using perturbation model to obtain counterfactuals. Tran et al. [58] adopted influence functions for identifying training points most relevant to a recommendation while deducing a counterfactual set for explanations. Tan et al. [57] proposed to generate and evaluate explanations that considers the causal relations to the outcome.

Yet, our work is different from prior works on two key points: 1) In terms of problem definition, prior works generate counterfactual explanations to explain user behaviors or recommendation results, while our method generates such explanations to explain the fairness-utility trade-off in recommendation. 2) In terms of technique, our method adopts a counterfactual reasoning framework from a global perspective, which explains the entire model behavior, while prior works focus on generating individual explanations for an individual recommendation result.
2.2 Fairness in Recommendation

The issue of fairness in recommendation has received growing concerns as recommender systems touch and influence people’s daily lives more deeply and profoundly [28, 41, 62]. Several recent works focusing on fairness quantification have found various types of bias and unfairness in recommendations, such as gender and race [12, 41, 69], item popularity [1, 2, 24, 28], and user activeness [18, 40]. Meanwhile, the relevant methods for fair recommendation focusing on providing fair recommendation results based on pre-defined fairness, can be roughly divided into three categories: pre-processing, in-processing and post-processing algorithms [42]. First of all, pre-processing methods usually aim to minimize the bias in the data sources. It includes fairness-aware sampling methodologies in the data collection process to cover items of all groups, balancing methodologies to increase coverage of minority groups, and repairing methodologies to ensure label correctness [23]. Secondly, in-processing methods aim at encoding fairness as part of the objective function, typically as a regularizer [1, 6, 24, 41]. Finally, post-processing methods modify the presentation of the results, e.g., by re-ranking through linear programming [40, 55, 68] or multi-armed bandit [8]. Based on the characteristics of the recommender system itself, there also have been a few works related to multi-sided fairness in multi-stakeholder systems [7, 22].

Moreover, there are two primary paradigms adopted in recent studies on algorithmic discrimination: individual fairness and group fairness [42]: individual fairness requires that each similar individual should be treated similarly; and group fairness requires that the protected groups should be treated similarly to the advantaged group or the populations as a whole. In this paper, we mainly focus on the item popularity fairness, which is a kind of group fairness and aims to achieve fair chances of exposure for different item groups [1, 2, 24].

2.3 Fairness Explanation

Explainability and fairness are two important perspectives for responsible recommender systems, however, the relationship between the two is still less explored. There have been several pioneering studies trying to derive explanations for model fairness [5, 50] in other tasks. For example, Begley et al. [5] leveraged Shapley value paradigm [52] to attribute the feature contributions to model disparity to generate explanations. It estimates the sum of individual contributions from input features, so as to understand which feature contributes more to the model disparity [5]. Though this type of methods successfully provide explanations to the model disparities, they are not suitable for recommender systems. First of all, the definition of Shapley value is the average marginal contribution of a feature value across all possible coalitions, meaning that the computation time increases super-exponentially with the number of features. In recommendation systems, this becomes impractical since it is very common to have a large number of user/item features in the feature space. Secondly, the Shapley value can only explain either utility or fairness alone [5, 50], but not the fairness-utility trade-off. However, our proposed Counterfactual Explainable Fairness (CEF) framework is able to mitigate the above problems.

3 EXPLAINABLE FAIRNESS

In this section, we first introduce how to use review information to generate user-feature matrix and item-feature matrix, then introduce the details of feature-aware recommendation systems. We introduce how to generate counterfactual explanations for fairness in section 3.4 and 3.5.

3.1 Feature Generation

Suppose we have a user set with $m$ users denoted as $\mathcal{U}$, an item set $\mathcal{V}$ with $n$ items and their interaction set $\mathcal{I} = \{(u, v)|u \in \mathcal{U}, v \in \mathcal{V}, u \text{ has interacted with } v\}$. Based on an open source toolkit for phrase-level sentiment analysis, called “Sentires”\(^1\), we can easily convert the raw review information into a set of quadruples $\mathcal{W} = \{(u_l, v_l, f_l, s_l)\}_{l=1}^N$. Specially, each element $(u_l, v_l, f_l, s_l)$ in $\mathcal{W}$ means user $u_l \in \mathcal{U}$ mentioned feature $f_l \in \mathcal{F}$ of item $v_l \in \mathcal{V}$ with sentiment $s_l \in S$, where $\mathcal{F}$ denotes the set of all features with size $r$ and the sentiment set $S = \{positive(+1), negative(-1)\}$. For example, in the review of “I like the color of this sweater, but the sleeve is not satisfied, since it is too tight for me,” the features are “collar” and “sleeve”, and the user expresses positive and negative sentiments on them. The final extracted tuples are “(user, item, color)” and “(user, item, sleeve, negative)”, respectively. Following the same method described in [13, 57, 74], we construct a user-feature attention matrix $\mathbf{A} \in \mathbb{R}^{m \times r}$ and an item-feature quality matrix $\mathbf{B} \in \mathbb{R}^{n \times r}$ using all the quadruples in $\mathcal{W}$, where $A_{u,f}$ indicates to what extent the user $u$ cares about the feature $f$, and $B_{v,f}$ indicates how well the item $v$ performs on the feature $f$. Specifically, $A$ and $B$ are calculated as:

$$
\begin{align*}
A_{u,f} &= \begin{cases} 
0, & \text{if user } u \text{ did not mention feature } f \\
1 + (M - 1) \left( \frac{2}{1 + \exp(-t_{u,f})} - 1 \right), & \text{else}
\end{cases} \\
B_{v,f} &= \begin{cases} 
0, & \text{if item } v \text{ has no review on feature } f \\
1 + \frac{M - 1}{1 + \exp(-t_{v,f})}, & \text{else}
\end{cases}
\end{align*}
$$

(1)

where $M$ is the rating scale in the system, which equals to 5 (stars) in most cases, $t_{u,f}$ is the frequency that user $u$ mentioned aspect $f$, $t_{v,f}$ is the frequency that item $v$ is mentioned on feature $f$, and $t_{u,f}$ is the average sentiment of these mentions. For both $A$ and $B$ matrices, their elements are re-scaled into the range of $(1, M)$ using the sigmoid function (see Eq.(1)) to match with the original system’s rating scale. Readers may refer to [74, 75] for more details and the same user-feature and item-feature matrix construction technique can also be found in [20, 35, 57, 61].

3.2 Feature-aware Recommender Systems

Once given the user-feature attention matrix $\mathbf{A}$ and item-feature quality matrix $\mathbf{B}$, we define a ranking model $g$ that predicts the user-item ranking score $\hat{y}_{u,v}$ for user $u$ and item $v$ by:

$$
\hat{y}_{u,v} = g(A_{u}, B_{v} | Z, \Theta)
$$

(2)

where $A_{u}$ and $B_{v}$ are the vector of user $u$ and the vector of item $v$, $\Theta$ is the model parameter, and $Z$ represents all other auxiliary information. Depending on the application, $Z$ could be rating scores,
clicks, text, images, etc., and is optional in the recommendation model \( g \).

In this work, we explore different implementations of \( g \) to demonstrate the effectiveness of our proposed framework. The general architecture of \( g \) is a multi-layer neural network, that is:

\[
\hat{y}_{u,o} = W_T \sigma_T (\ldots (W_1 \sigma_1 (\text{merge}(A_u, B_u)) + b_1) + \ldots) + b_T
\]

(3)

where, for the \( t \)-th layer \((t = 1, 2, \ldots, T)\), \( \sigma_t \) is a non-linear activation function, \( W_t \) and \( b_t \) are weights and bias terms, respectively; \( \text{merge}() \) is an operator merging the user-feature and item-feature vectors, and we explore it within the following functions:

- **Element-wise Product Merge**:
  
  \[
  \text{merge}(A_u, B_u) = W_L A_u^T \odot W_R B_u^T.
  \]
  
  (4)

  where \( W_L \) and \( W_R \) are trainable parameters, and \( \odot \) represents the element-wise product (a.k.a. Hadamard product).

- **Concatenation Merge**:
  
  \[
  \text{merge}(A_u, B_u) = [W_L A_u^T, W_R B_u^T].
  \]
  
  (5)

  where \( W_L \) and \( W_R \) are trainable parameters.

Then, we train the model with a cross-entropy loss:

\[
\text{Loss} = - \sum_{u,v,y_{u,v}=1} \log \hat{y}_{u,v} - \sum_{u,v,y_{u,v}=0} \log (1 - \hat{y}_{u,v})
= - \sum_{u,v} y_{u,v} \log \hat{y}_{u,v} + (1 - y_{u,v}) \log (1 - \hat{y}_{u,v})
\]

(6)

where \( y_{u,v} = 1 \) if user \( u \) previously interacted with item \( v \), otherwise \( y_{u,v} = 0 \).

Generally, the recommendation model \( g \) can be any ranking model as long as it takes the user-feature and the item-feature vectors as the input. The implementation and training of \( g \) will be detailed in the experiment section.

Finally, given \( \{U, V, T, A, B, g\} \), our task is to generate feature-based explanations in terms of recommendation disparity for the black-box recommendation model \( g \). Besides, most of the important symbols used in the paper can be referred in Tab. 1.

### 3.3 Fairness and Disparity

In this work, we consider explaining the exposure unfairness due to popularity bias in recommendation. Given a recommendation model \( g \), we will have a certain recommendation result \( K_K = \{R(u_1, K), R(u_2, K), \ldots, R(u_m, K)\} \) containing all users’ top-\( K \) recommendation lists. These recommendations determine the exposures of items, which is used to measure the fairness and disparity of the model. We then split items into two groups based on their number of exposures in the recommendation list and denote \( G_0 \) as popular item group and \( G_1 \) as long-tailed item group. Based on the above notations, we list some popular algorithmic fairness definitions related to popularity bias as follows:

#### 3.3.1 Demographic Parity (DP)

Demographic parity in recommendation scenarios requires that the average exposure of the items from each group is equal [24, 55]. First, given \( K_K \), we denote the number of exposures in group \( G_l \) as

\[
\text{Exposure}(G_l|K_K) = \sum_{u \in \mathcal{U}} \sum_{v \in R(u, K)} I(v \in G_l), l \in \{0, 1\}.
\]

(7)

where \( I \) is the indicator function.

\[\text{Exposure}(G_0|K_K) = \frac{|G_0|}{|\mathcal{G}_0|} \cdot \text{Exposure}(G_0|K_K) - |G_0| \cdot \text{Exposure}(G_1|K_K)
= \alpha \cdot |\mathcal{G}_1| \cdot \text{Exposure}(G_1|K_K))
\]

(9)

where \( \alpha \in (0, 1) \). Note that when \( \alpha = |G_0|/|\mathcal{G}_0| \) and the equation holds strictly, the above expression would be exactly the same as demographic parity.

#### 3.3.3 Disparity

In practice, we can take the difference between the two sides of the equalities in the above definitions as a quantification measure for disparity. For example,

\[
\Psi_{DP} = |G_1| \cdot \text{Exposure}(G_0|K_K) - |G_0| \cdot \text{Exposure}(G_1|K_K)
\]

(10)

\[
\Psi_{EK} = \text{Exposure}(G_0|K_K) - \alpha \cdot |\mathcal{G}_1| \cdot \text{Exposure}(G_1|K_K)
\]

(11)

are two popular algorithm disparity measures used in fairness learning algorithms [24].

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathcal{U} )</td>
<td>The set of users in a recommender system</td>
</tr>
<tr>
<td>( \mathcal{V} )</td>
<td>The set of items in a recommender system</td>
</tr>
<tr>
<td>( \mathcal{T} )</td>
<td>The set of user-item interactions in a recommender system</td>
</tr>
<tr>
<td>( \mathcal{F} )</td>
<td>The set of features in a recommender system</td>
</tr>
<tr>
<td>( \mathcal{S} )</td>
<td>The set of sentiments in a recommender system</td>
</tr>
<tr>
<td>( m )</td>
<td>The number of users</td>
</tr>
<tr>
<td>( n )</td>
<td>The number of items</td>
</tr>
<tr>
<td>( r )</td>
<td>The number of features</td>
</tr>
<tr>
<td>( u )</td>
<td>A user ID in a recommender system</td>
</tr>
<tr>
<td>( v )</td>
<td>An item ID in a recommender system</td>
</tr>
<tr>
<td>( f )</td>
<td>A feature index in a recommender system</td>
</tr>
<tr>
<td>( s )</td>
<td>A sentiment index in a recommender system</td>
</tr>
<tr>
<td>( A )</td>
<td>A user-feature attention matrix</td>
</tr>
<tr>
<td>( B )</td>
<td>A item-feature quality matrix</td>
</tr>
<tr>
<td>( A'f )</td>
<td>The item-feature quality matrix after intervention with ( \Delta_u )</td>
</tr>
<tr>
<td>( B'f )</td>
<td>The item-feature quality matrix after intervention with ( \Delta_v )</td>
</tr>
<tr>
<td>( \mathcal{G}_0 )</td>
<td>The set of popular items</td>
</tr>
<tr>
<td>( \mathcal{G}_1 )</td>
<td>The set of long-tailed items</td>
</tr>
<tr>
<td>( y_{u,v} )</td>
<td>Ground-truth value of the pair ( (u, v) )</td>
</tr>
<tr>
<td>( \hat{y}_{u,v} )</td>
<td>Predicted value of the pair ( (u, v) )</td>
</tr>
<tr>
<td>( K )</td>
<td>The length of the recommendation list</td>
</tr>
<tr>
<td>( R_K )</td>
<td>The set of recommendation lists with length ( K ) for all users</td>
</tr>
<tr>
<td>( \Theta )</td>
<td>Parameters of black-box recommendation model</td>
</tr>
</tbody>
</table>

Then, we can express demographic parity fairness as follows,

\[
\frac{\text{Exposure}(G_0|K_K)}{|G_0|} = \frac{\text{Exposure}(G_1|K_K)}{|G_1|}.
\]

(8)
3.4 Counterfactual Reasoning

With the above notations and definitions of item exposure fairness, we can measure the disparity of the top-K recommendation result $R_K$. Then, the objective of our counterfactual reasoning problem is to generate feature-based explanations for the given black-box recommendation model $g$. The essential idea of the proposed explanation model is to discover a slight change $Δ_0$ on each feature via solving a counterfactual optimization problem, which minimizes the disparity and a perturbation constraint that represents the effort to change the disparity, so that we can know which feature(s) are the underlying reasons for model disparity.

Specially, for each user-feature vector $A_f$, we slightly intervene with a vector $Δ_u \in \mathbb{R}^m$ (and for each item-feature vector $B_f$, we intervene with $Δ_o \in \mathbb{R}^r$), more specifically, the value of certain user feature $f$ for all users $A_f$ will be added to $Δ_u$ and get $A_u^f$, (or the value of certain item feature $f$ for all items $B_f$ will be added to $Δ_o$ and get $B_o^f$). With the new user-feature matrix $A_u^f$ and item-feature matrix $B_o^f$, $g$ will change the recommendation result from $R_K$ to a counterfactual result $\hat{R}_K$. More importantly, this will also change the fairness measure of that result to $Ψ^f$, where $Ψ^f$ can either be $Ψ^f_{EK}$ or $Ψ^f_{DP}$ depending on the choice of disparity. And our goal is to look for the minimum intervention on user/item feature that is able to result in the greatest reduction in terms of disparity or unfairness. Thus, objective function would be:

$$
\min \|Ψ^f\|_2^2 + λ\|Δ\|_2
$$

where $Δ$ can be either $Δ_u$ or $Δ_o$ or the concatenation of them ($Δ = [Δ_u, Δ_o]$), $λ \in (0, 1)$ is a hyper-parameter that is used to control the weight between the two terms, and $Ψ^f$ can be:

$$
Ψ^f_{EK} = \text{Exposure}(G_0|R^f_K) - α \cdot \text{Exposure}(G_1|R^f_K) = \sum_{u \in U} \sum_{v \in \text{CF}(u,K)} I(v \in G_0) - α \cdot \sum_{u \in U} \sum_{v \in \text{CF}(u,K)} I(v \in G_1)
$$

(13)

$$
Ψ^f_{DP} = \text{Exposure}(G_0|R^f_K) - |G_0| - |G_1| \cdot \text{Exposure}(G_1|R^f_K) = \sum_{u \in U} \sum_{v \in \text{CF}(u,K)} I(v \in G_0) - |G_0| \cdot \sum_{u \in U} \sum_{v \in \text{CF}(u,K)} I(v \in G_1)
$$

(14)

A major challenge to optimize Eq. (12) is the non-differentiable nature of $Ψ^f$. As a relaxation, we replace the indicator function $I(\cdot)$ in the original definition (Eq. (14) or Eq. (13)) with $g(\cdot, \cdot)$, which is the predicted ranking score, and normalize the final results to stabilize the gradients of the objective function. And the resulting disparity metric $\hat{Ψ}^f$ becomes:

$$
\hat{Ψ}^f_{EK} = \frac{\sum_{u \in U} (\sum_{v \in \text{CF}(u,K)} g(A_u^f, B_o^f) - α \cdot \sum_{v \in \text{CF}(u,K)} g(A_u^f, B_o^f))}{\sum_{u \in U} \sum_{v \in \text{CF}(u,K)} g(A_u^f, B_o^f)}
$$

or becomes:

$$
\hat{Ψ}^f_{DP} = \frac{\sum_{u \in U} (\sum_{v \in \text{CF}(u,K)} g(A_u^f, B_o^f) - \frac{|G_0|}{|G_1|} \cdot \sum_{v \in \text{CF}(u,K)} g(A_u^f, B_o^f))}{\sum_{u \in U} \sum_{v \in \text{CF}(u,K)} g(A_u^f, B_o^f)}
$$

(15)

(16)

Thus, our final objective for a given feature is

$$
\min \|\hat{Ψ}^f\|_2^2 + λ\|Δ\|_2
$$

The first term aims to realize the greatest reduction in terms of pre-defined disparity or unfairness. The second perturbation constraint represents the edit distance between original inputs and the corresponding counterfactuals. Finally, for each feature, we solve an optimization problem defined as Eq. (17) and use the corresponding counterfactual recommendation result to calculate the explainability score, which will be detailed in the next section.

3.5 Generate Feature-based Explanations

For each feature in the feature space, we will solve the optimization problem defined as Eq. (17) and consider $Δ$ as the only trainable parameter. Once finished optimizing, we will get the "minimal" change $Δ$ and the corresponding recommendation results under such change to that feature. Then, we use $\text{Proximity}$—the average edit distance between original input and the corresponding counterfactual—to measure the degree of perturbation. And we use $\text{Validity}$—the change of fairness caused by the feature’s perturbation—to measure the degree of influence on fairness [48, 49, 57, 59].

$$
\text{Proximity} = \frac{\|Δ\|_2^2}{m \cdot K}
$$

(18)

$$
\text{Validity} = \frac{\text{Exposure}(G_0|R^f_K) - \text{Exposure}(G_1|R^f_K)}{m \cdot K} - \frac{\text{Exposure}(G_0|R^f_K)}{m \cdot K}
$$

(19)

where $m$ is the number of users and $K$ is the length of recommendation lists.

Finally, the explainability score ($ES$) is the linear combination of $\text{Proximity}$ and $\text{Validity}$, which is shown as follows:

$$
ES = \text{Validity} - β \cdot \text{Proximity},
$$

(20)

where $β \in (0, 1)$ and larger score represents better explainability.

This score determines the ranking of a feature in terms of its ability to reduce the disparity of model $g$ while keeping the perturbation small. Note that the original value of the feature corresponds to the optimal recommendation utility of $R_K$ that the model $g$ learned, so larger proximity score may imply a greater sacrifice of utility. Thus, the inclusion of this term in the objective function and the scoring function will result in an explanation finding process that is aware of the influence on both fairness and recommendation utility.

4 EXPERIMENTS

4.1 Datasets

To evaluate the models under different data scales, data sparsity and application scenarios, we perform experiments on three widely-used real-world datasets [26, 29, 32, 43, 57]. Some basic statistics of the experimental datasets are shown in Table 2.

- **Yelp** dataset contains users’ reviews on various kinds of businesses such as restaurants, dentists, salons, etc. This dataset contains 6,685,900 reviews, 192,609 businesses, 200,000 pictures in 10 metropolitan areas.

- **Amazon** dataset contains user reviews on products in Amazon e-commerce system\(^2\). The Amazon dataset contains 29 sub-datasets corresponding to 29 product categories. We adopt two datasets

\(^2\)https://www.yelp.com/dataset

\(^3\)https://nijiaanno.github.io/amazon/index.html
of different scales to evaluate our method, which are CDs & Vinyl and Electronics.

Since the Yelp and Amazon review datasets are very sparse, similar as previous work [57, 61, 74], we remove the users and items with fewer than 20 reviews. For each dataset, we first sort the records of each user based on the timestamp, and then hold-out the last 5 interacted items together with 100 randomly sampled negative items for each user to serve as the test data to evaluate black-box recommenders and do fairness explanation. The last item in the training set of each user is put into the validation set. Since we focus on item exposure fairness, we need to split items into two groups $G_0$ and $G_1$ based on item popularity. It would be desirable if we have the item impression/listing information and use it to group items, however, since Yelp and Amazon datasets are public dataset and only have interaction data, we use the number of interaction to group items in them. Specifically, for Yelp and Amazon review datasets, the top 20% items in terms of number of interactions belong to the popular group $G_0$, and the remaining 80% belong to the long-tail group $G_1$.

![Table 2: Basic statistics of the experimental datasets.](image)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#User</th>
<th>#Item</th>
<th>#Review</th>
<th>#Aspect</th>
<th>Density</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yelp</td>
<td>12,028</td>
<td>20,181</td>
<td>502,158</td>
<td>106</td>
<td>0.208%</td>
</tr>
<tr>
<td>CDs &amp; Vinyl</td>
<td>3,225</td>
<td>46,709</td>
<td>179,992</td>
<td>118</td>
<td>0.119%</td>
</tr>
<tr>
<td>Electronics</td>
<td>2,762</td>
<td>19,449</td>
<td>51,777</td>
<td>77</td>
<td>0.096%</td>
</tr>
</tbody>
</table>

4.2 Black-box Recommender System

As mentioned before, we first follow prior works [13, 61, 74] to build a user-feature attention matrix and an item-feature quality matrix, and use both matrices together with the user-item interaction history to train a feature-aware recommendation model.

In this work, to demonstrate the idea of counterfactual explainable fairness, we use a simple deep neural network as the implementation of the recommendation model $g$, which includes one fusion layer followed by three fully connected layers with size {feature size, 256}, while for Concatenation merge, it is {feature size, 256}. The final output layer is a sigmoid activation function so as to map $y_{u,v}$ into the range of $(0, 1)$.

The model parameters are optimized by stochastic gradient descent (SGD) optimizer with a learning rate of 0.01. After the recommendation model is trained, all the parameters will be fixed in the counterfactual reasoning phase and explanation evaluation phase. The recommendation performance on Element-wise product merge (Eq. (4)) and Concatenation merge (Eq. (5)) are presented in Tab. 3. For convenience and simplicity, the evaluations of fairness explanation methods presented in the experiment section are based on Element-wise product merge (Eq. (4)).

Table 3: Summary of recommendation performance on three datasets for black-box recommendation models using Element-wise Product merge (Eq. (4)) and Concatenation merge (Eq. (5)) in term of F1 and NDCG.

<table>
<thead>
<tr>
<th>Recommender</th>
<th>F1 (%) @5</th>
<th>F1 (%) @20</th>
<th>NDCG (%) @5</th>
<th>NDCG (%) @20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yelp</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Element-wise</td>
<td>17.161</td>
<td>16.563</td>
<td>16.069</td>
<td>29.192</td>
</tr>
<tr>
<td>Concatenation</td>
<td>16.266</td>
<td>16.929</td>
<td>17.338</td>
<td>29.780</td>
</tr>
<tr>
<td>Electronics</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Element-wise</td>
<td>15.112</td>
<td>13.975</td>
<td>16.384</td>
<td>25.886</td>
</tr>
<tr>
<td>Concatenation</td>
<td>15.083</td>
<td>14.044</td>
<td>16.350</td>
<td>25.946</td>
</tr>
<tr>
<td>CDs &amp; Vinyl</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Element-wise</td>
<td>21.463</td>
<td>18.517</td>
<td>23.150</td>
<td>35.162</td>
</tr>
<tr>
<td>Concatenation</td>
<td>20.737</td>
<td>18.443</td>
<td>22.393</td>
<td>34.672</td>
</tr>
</tbody>
</table>

4.3 Baselines

Since there is no existing method specifically designed to explain fairness in recommendation. We adopt the following explanation methods as baselines:

- **Random**: We randomly choose multiple features from the feature space without replacement and use them as explanation results.
- **Popularity**: We rank all the features in the user-feature matrix and item-feature matrix based on their number of existences, and select the top ones as explanations, and denote them as Pop-User and Pop-Item, respectively.
- **EFM** [74]: The Explicit Factor Model (EFM) for explainable recommendation. This work integrates matrix factorization with explicit features to align latent factors with explicit aspects for explanation. In this way, it predicts the user-feature preference scores and item-feature quality scores. The original EFM uses the element-wise product of user-feature vector and item-feature vector and select the top ones as explanations to a given user-item pair. To generate global explanations, we calculate the average value of each feature from both user side and item side, and use them as explanations. Therefore, we have EFM-User and EFM-Item. Note that these features only explains the recommendation utility but do not explain the fairness.
- **Feature-based Explanation by Shapley Values (SV)**: Begley et al. [5] leveraged Shapley value-based methods to attribute the model disparity as the sum of individual contributions from input features to understand which feature contributes more or less to the model disparity. Considering the large number of features in the feature space, instead of using all possible coalitions, which is $r!$, we randomly sample 100 feature coalitions to calculate the Shapley value for each feature.

For CEF, we choose to minimize Eq. (17), where $\Delta = [\Delta_u, \Delta_v]$, $\Psi^{cf} = \Psi^{cf}_{DP}$ (Eq. (14)), and $\frac{|G_0|}{|G_1|} = \frac{1}{4}$. We set the hyper-parameter $\lambda = 1$ and $K = 5$. The model parameters are optimized by Adam optimizer with a learning rate of 0.01.

4.4 Evaluation Methods and Metrics

Once we obtain the feature-based explanations from each baseline as well as our proposed CEF, we need to compare the effectiveness
of these results, in other words, their contributions to the fairness-utility trade-off. In order to evaluate the feature-based explanations, we follow the widely deployed erasure-based evaluation criterion in Explainable AI. The intuition behind the erasure-based criterion is to measure how much the model performance would drop after the set of the “most important” features in an explanation is removed [70, 72]. Similarly, in the setting of explainable fairness, we use it to measure the fairness-utility trade-off in recommendation, namely, how much the recommendation performance would drop and how much the recommendation fairness would improve after the set of the “most important” features in an explanation is removed. Specifically, for each feature-based explanation result, we erase the set of the “most important” features in both the user-feature and item-feature matrices for all users and items, then input the erased user-feature and item-feature matrices into pre-trained recommendation model \( g \) to generated a new recommendation results. Based on the recommendation performance and fairness of the new results, we compare the effectiveness of each explainable fairness methods.

We select several most commonly used top-\( K \) ranking metrics to evaluate the model’s recommendation performance after erasure, including \( \text{F1 Score} \), and \( \text{NDCG} \). For fairness evaluation, we define

Figure 1: The accuracy-fairness trade-off curves for NDCG and Long-tail Rate on various datasets. The upper-right corner of each figure (high accuracy, low disparity) is preferred. Each data point is generated by cumulatively removing top 5 features in the explanation lists provided by explanation methods.
where $D$ is the number of erased features. Bold scores are used to indicate the greatest values.

<table>
<thead>
<tr>
<th>Methods</th>
<th>F1@5 (%)</th>
<th>NDCG@5 (%)</th>
<th>Long-tail Rate@5 (%)</th>
<th>KL@5 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>E=5</td>
<td>E=10</td>
<td>E=20</td>
<td>E=5</td>
</tr>
</tbody>
</table>

$KL(D_1||D_2) = \sum_{j \in \{0,1\}} D_1(j) \log \frac{D_1(j)}{D_2(j)}$  \hspace{1cm} (21)

where $D_1$ represents the true group distribution between $G_0$ and $G_1$ in top-K recommendation list, and $D_2 = [|G_0|/|V|, |G_1|/|V|]$ represents their ideal distribution of the overall population.

### 4.5 Experimental Results

The major experimental results are shown in Fig. 1, where we plot the fairness-utility trade-off, i.e., the relationship between NDCG and Long-tail Rate (namely, 1-Popularity Rate) with different length of recommendation lists ($@5$, $@20$, $@50$). Since the relationship between F1 and Long-tail Rate has very similar conclusions, we choose not to present them here. Each data point Fig. 1 is generated by cumulatively removing top 5 features in the remaining explanation list provided by each explanation method. We also present the values of F1@5, NDCG@5, Long-tail Rate@5 and KL@5 in Tab. 4 after removing top-5, top-10, top-20 features in each explanation result to quantitatively analyse the results.

First, in Fig. 1 and Tab. 4, we can easily find that all the methods, even randomly selecting features and erasing them, can improve recommendation fairness. Besides, the higher the number of features we erase, the lower the disparity rate we can achieve. This is easy to understand as erasing features will mitigate the representation gap between popular items and long-tailed items, causing more under-represented items to be recommended. However, it also brings huge decline to the recommendation performance. For example, compared with the original recommendation performance on NDCG@5, the method with the worst trade-off behavior drops relatively 2.119 % on Yelp, 21.786 % on Electronics, and 7.072 % on CDs & Vinyl when deleting top 5 features. Second, we can see that even though the idea of using popular features as explanations is very intuitive, their performance may even be worse than random selection, which indicates that compared with fairness, popular features either from user side or item side are more sensitive to recommendation performance, while random selection guarantees low probabilities of choosing those scarce features, which in turn results in better trade-off. Third, the performances of SV are much worse than CEF as it only explains disparity alone, ignoring the inherent trade-off between fairness and utility. Finally, in Fig.
1, where the blue dotted line represents the performance of our proposed CEF framework, it is obvious that the feature-based explanations provided by CEF are capable of achieving much better fairness-utility trade-off on datasets with various scales and densities. Specifically, compared with the original recommendation performance on NDCG@5, CEF method drops only relatively 0.851 % on Yelp, 2.682 % on Electronics, and 0.594 % on CDs & Vinly, while it increases relatively 13.431 % on Yelp, 25.73 % on Electronics, and 3.085 % CDs & Vinly at Long-tail Rate@5.

As is shown in Fig. 2, the evaluations on CEF-User and CEF-Item are worse than the baselines, which indicates the effectiveness of our proposed framework. Moreover, even though CEF-User and CEF-Item are worse than CEF, they are still far more better than most of the baselines, especially, CEF-User is better than all the baselines when using them to do fair learning. In the future, we hope to design algorithmic methods that can generate multiple explanations at the same time without greedy choosing them through explainability scores. (One possible solution would be using penalizing vectors to control the number of perturbed features.)

**ACKNOWLEDGMENTS**

We appreciate the valuable feedback of the reviewers. This work was supported in part by NSF IIS 1910154, 2007907, 2046457 and Facebook Faculty Research Award. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsors.

**A APPENDIX: CASE STUDY**

In this section, we provide the top-5 feature-based explanations that are generated by each method on Yelp dataset. The explanation results are shown in Tab. 5, which exactly verifies our motivation that it is difficult to manually identify feature explanations for exposure unfairness and popularity bias in recommender system. For example, it is hard to tell how input features (like chicken, cheese, pizza) would influence the exposure opportunity in restaurant recommendation. Thus, we do need explainable fairness methods to identify such features in recommendation.

<table>
<thead>
<tr>
<th>Method</th>
<th>Feature-based Explanations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pop-User</td>
<td>food, service, chicken, prices, hour</td>
</tr>
<tr>
<td>Pop-Item</td>
<td>food, service, prices, visit, hour</td>
</tr>
<tr>
<td>EFM-User</td>
<td>store, patio, dishes, dish, rice</td>
</tr>
<tr>
<td>EFM-Item</td>
<td>flavor, decor, dishes, inside, cheese</td>
</tr>
<tr>
<td>SV</td>
<td>server, size, pizza, food, restaurant</td>
</tr>
<tr>
<td>CEF</td>
<td>meal, cheese, dish, chicken, taste</td>
</tr>
</tbody>
</table>

Table 5: Top-5 feature-based explanations on Yelp dataset.

**REFERENCES**


Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In CIKM.


Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In CIKM.


