SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

August 2024, Vol. 67, Iss. 8, 182102:1–182102:15
https://doi.org/10.1007/s11432-023-3974-2

Learning-based counterfactual explanations for recommendation

Jingxuan WEN[1,2], Huafeng LIU[1,2*], Liping JING[1,2*] & Jian YU[1,2]

[1]*School of Computer and Information Technology, Beijing Jiaotong University, Beijing* 100044*, China;*
[2]*Beijing Key Lab of Traffic Data Analysis and Mining, Beijing Jiaotong University, Beijing* 100044*, China*

Received 19 January 2023/Revised 8 June 2023/Accepted 21 August 2023/Published online 25 July 2024

Abstract Counterfactual explanations provide explanations by exploring the changes in effect caused by changes in cause. They have attracted significant attention in recommender system research to explore the impact of changes in certain properties on the recommendation mechanism. Among several counterfactual recommendation methods, item-based counterfactual explanation methods have attracted considerable attention because of their flexibility. The core idea of item-based counterfactual explanation methods is to find a minimal subset of interacted items (i.e., short length) such that the recommended item would topple out of the top-K recommendation list once these items have been removed from user interactions (i.e., good quality). Usually, explanations are generated by ranking the precomputed importance of items, which fails to characterize the true importance of interacted items due to separation from the explanation generation. Additionally, the final explanations are generated according to a certain search strategy given the precomputed importance. This indicates that the quality and length of counterfactual explanations are deterministic; therefore, they cannot be balanced once the search strategy is fixed. To overcome these obstacles, this study proposes learning-based counterfactual explanations for recommendation (LCER) to provide counterfactual explanations based on personalized recommendations by jointly modeling the factual and counterfactual preference. To achieve consistency between the computation of importance and generation of counterfactual explanations, the proposed LCER endows an optimizable importance for each interacted item, which is supervised by the goal of counterfactual explanations to guarantee its credibility. Because of the model's flexibility, the trade-off between quality and length can be customized by setting different proportions. The experimental results on four real-world datasets demonstrate the effectiveness of the proposed LCER over several state-of-the-art baselines, both quantitatively and qualitatively.

Keywords recommender system, explainable recommendation, item-based explanation, counterfactual inference, counterfactual explanation

1 Introduction

Filter failure significantly affects the effectiveness of user decision-making [1]. Recommender systems refine decision-making efficiency and improve user satisfaction by recommending items with potential interest for users. With the improvement of recommendation performance, recommendation models become too complex to be described in a human-interpretable manner. Thus, explaining why a recommendation is produced is imperative to improve the transparency, trustworthiness, and user satisfaction of recommender systems [2]. One straightforward way for explanation is to use content information to provide semantic meaning, such as movie tags [3], knowledge graph [4], and text review [5,6]. For instance, some researchers extracted key information from user reviews to capture user preferences, thereby providing explanations. Additionally, tag-based explanation methods provide explanations according to the matching extent between item tags and user preference. We choose a tag-based explanation as an example shown in Figure 1. However, content information is not always available [7]. In addition to content information, it is intuitive to use interacted items as explanations because users are more familiar with them [2]. It selects a subset of items with which the current user interacts for the explanation, which is called an item-based explanation.

* Corresponding author (email: hfliu1@bjtu.edu.cn, lpjing@bjtu.edu.cn)

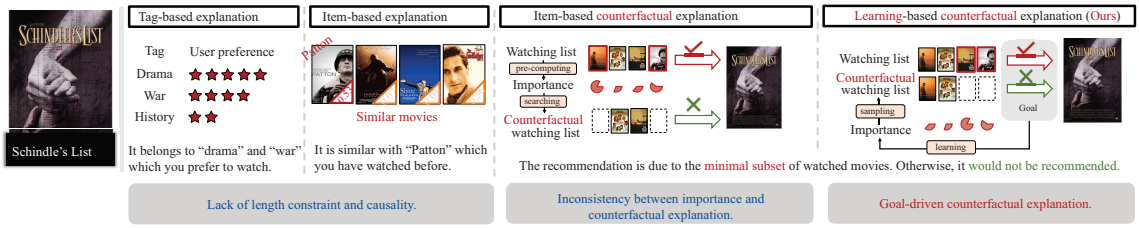

Figure 1 (Color online) Illustration of three types of explanations for recommendation. For CEs, we list existing methods and our learning-based methods for comparison.

Most existing methods, such as attention weights [8] (illustrated in Figure 1), provide item-based explanations based on the correlation between interacted items and recommended items. However, these methods can only provide similarity scores, and the formation of the final explanation requires manual determination (e.g., the threshold of similarity for determining explanations needs manual adjustment). It is time-consuming and labor-intensive, cannot guarantee the shortest explanation, and does not consider individual differences among users and recommendation results. Moreover, correlation is the product of the simplest cognitive ability, i.e., observing [9]. The failure to disentangle correlation from causality can lead to suboptimal or even erroneous explanations [10]. In contrast, counterfactual describes the causality by exploring the changes in effect caused by changes in cause [10,11]. Therefore, counterfactual explanations (CEs) have recently received significant attention because they provide users with more convincing explanations [12–15]. Operationally, its goal is to perturb the input slightly to change its original prediction result [12,16]. Accordingly, we conclude that a good CE should possess two properties: good quality and short length. Good quality refers to the expectation that the model's prediction result for the perturbed input will change successfully, whereas short length means the smaller the perturbation to the input, the better.

In recommendation scenarios, item-based CEs for top-K recommendations can be generated by answering a counterfactual question: "What would user preference be if a subset of interactions performed by the current user were removed?" Figure 1 also shown an example of the item-based CE. For the current user, the reason for recommending the movie "Schindle's List" is that "If you had not watched 'The Killing Fields' and 'Patton', 'Schindle's List' would not be recommended." The two properties mentioned above can also be extended to recommendation scenarios. One is of good quality, i.e., CE can reflect the "cause" of the recommended item. Thus, we expected that it could make the recommended item drop out of the top-K recommendation list by removing a subset of interactions. The other is of short length, i.e., CE should be concise to be easy for users to understand. Thus, the subset should be as small as possible. To find the items that truly cause the recommendations, some recent studies have endowed each interacted item with importance to guide the generation of CEs [7, 17, 18]. Specifically, they first calculate the importance of each interacted item and then use different search strategies, either heuristic or greedy, to generate CEs based on the precomputed importance. Although they provide some directions for item-based CEs in recommendation scenarios, they still have some limitations. On the one hand, the computation of importance is separated from the generation of CEs, which may fail to characterize the true importance of interacted items to CEs. On the other hand, for a specific algorithm, it is impossible to adjust the proportion of quality and length on demand. In particular, in real-world scenarios, users prefer meaningful but slightly long explanations to short, worthless explanations. However, once the search algorithm is fixed, the quality and length of CEs are deterministic and negatively correlated. Thus, for a specific algorithm, it is difficult for researchers to generate CEs with more convincing interpretability at the cost of length if the generated CEs are less convincing.

To overcome these obstacles, this study proposes a learning-based counterfactual explanations for recommendation (LCER), as shown in the fourth example of Figure 1. The goal of LCER, as well as other item-based CE methods, is to generate CEs with good quality and short length. By jointly modeling the factual and counterfactual preferences, the proposed LCER unifies the recommendation and counterfactual inference processes in an overall framework to provide CEs based on personalized recommendations. To differentiate the effect of the interacted items for generating CEs, we endow an optimizable importance for each interacted item. The learning of importance is guided by the goal of CEs to guarantee its faithfulness. Meanwhile, CEs are learned with the learning of importance instead of the search strategy. In this way, consistency between the computation of importance and the generation of

CEs can be ensured. Additionally, the proposed LCER enables the determination of different proportions of quality and length on demand within the goal of CEs. Thus, the trade-off between quality and length can be customized, and the learned importance can be more consistent with the goal of CEs. We also conducted a series of experiments on four public datasets[1)] to evaluate the effectiveness of the proposed LCER. The experimental results demonstrated significant improvements in the proposed LCER, both quantitatively and qualitatively.

2 Related work

CE raises the understanding of algorithmic decisions by means of a "what-if" explanation, whose core lies in a counterfactual question: "What would the prediction be if input features had been different?" To answer this question, CEs are expected to flip the algorithmic decision with a slight perturbation on the original input. However, turning over the decision is not enough; a good CE[2)] should also be sparse to raise understanding, be feasible to give guidance for future decisions, and be able to contest decisions [16].

Research on CE has received widespread attention since [12]. Given an input x and its corresponding output y through a model f, Wachter et al. [12] formulated the optimization problem of CEs as $\arg\min_{\boldsymbol{x}'}(f(\boldsymbol{x}')-y')^2+\lambda d(\boldsymbol{x},\boldsymbol{x}')$, where λ is a trade-off coefficient, and y' denotes the desired output. $d(\cdot,\cdot)$ is the L_1 norm weighted by inverse median absolute deviation (MAD), where L_1 norm induces sparsity, and MAD makes it more robust to outliers. On this basis, subsequent studies have been devoted to generating better CEs. To ensure the feasibility of CEs, Mahajan et al. [13] proposed that not only the proximity in sample space between CEs and original inputs should be constrained, and the causal relationships among input features should be preserved. DiCE [14] generated a diverse set of CEs for each individual input to allow users to choose a more suitable CE to guide their future behavior. To avoid defining distance function for input features, which may have multiple types, C-CHVAE [15] performs perturbation in latent space, leveraging the advantages of generative models to generate CEs in the data manifold. Additionally, there is also some discussion about CEs. For example, Laugel et al. [19] developed three desired properties of CEs and discussed how to quantify them. By comparing attribution-based explanations [20,21] and CEs, Mothilal et al. [22] presented that attribution-based explanations focus on sufficiency, whereas CEs concern necessity and their complementarity should not be neglected.

Applications in recommender system. To the best of our knowledge, generating explanations based on counterfactual reasoning for recommendation was first proposed in FIA [23]. For a test user u and one of the recommended items i for u, FIA measures the influence [24] of removing an interacted item on the prediction score of (u,i) pair and takes the most influential interacted item as the explanation. By portraying the dependence between interacted and recommended items, FIA answers the counterfactual question: "What would the prediction change if this interaction was not provided?" However, explanations generated by FIA do not ensure that the recommendation result would change if one interaction was removed. Thus, they are not CEs, although they are generated based on counterfactual reasoning. On this basis, Accent [7] sorts the interacted items by their influence on the prediction score of the recommended item and adds items into the counterfactual set in order of influence from high to low until the recommended items are replaced. Except for guiding the generation of CEs by influence score, Kaffes et al. [18] employed normalized length and impotence to design a heuristic search strategy. Prince [17] conducted the user-item interactions as a bipartite graph and generated CEs using Personalized PageRank [25]. In other words, all the above methods first calculate the importance of interacted items, and then generate CEs under the guidance of the precomputed importance. Except for utilizing a subset of interacted items as CEs, some researches [26,27] have generated CEs on feature level, which inevitably require side information to make the meaning of features understandable. Otherwise, the feature may not correspond to actual semantics, making it unable to provide effective explanations. Moreover, obtaining auxiliary information is more difficult than interaction information due to user privacy [28]. This paper, on the other hand, sets out from the perspective of data provenance [29]. We explore the impact of user interaction data on the final recommendation results and provide explanations instead of using additional auxiliary information to improve the responsibility of recommendation models based merely on interaction data. Moreover, since the explanations are derived from user interaction data, this approach can additionally remind us of some biases that exist in the recommender system, such as the popularity bias

1) Our implementation is available at https://github.com/xuan92ta/LCER.

2) It is not necessary to satisfy all of these properties to be considered good.

phenomenon confirmed indirectly in Subsection 4.3, by analyzing the relationship between the interacted items finally selected for explanation and the popularity of items.

Although existing item-based CE methods provide some guidance for generating CEs, there are several limitations. First, the importance of interacted items is precomputed regardless of the generation of CEs. Therefore, the separation between the computation of importance and generation of CEs may fail to reflect the true importance of interacted items to CEs. In other words, the faithfulness of importance cannot be ensured. Moreover, a good CE should simultaneously have good quality and short length. However, once search algorithm is fixed, the quality and length of CEs are deterministic and negatively correlated. Thus, it is not possible to adjust the proportion of quality and length on demand for a specific algorithm. The proposed LCER learns the importance of interacted items for CEs through a learning process and then generates CEs. Its learning process is guided by the goal of CE, ensuring consistency between the importance and final explanation goal. Moreover, thanks to this learning process, the ratio of length and quality of CEs can be adjusted as needed when setting the goal. For example, within an acceptable range, we can sacrifice a bit of the length of the CE to achieve a better explanation effect.

3 The proposed model

Suppose that there are N users and M items, then the implicit feedback is usually represented by a binary matrix $\boldsymbol{X}^f = \left[\boldsymbol{x}_1^f, \boldsymbol{x}_2^f, \ldots, \boldsymbol{x}_N^f\right]^{\mathrm{T}} \in \mathbb{N}^{N\times M}$, where $\boldsymbol{x}_u^f \in \mathbb{N}^{M\times 1}$ is the factual feedback of user u and corresponding interacted item set is denoted as $\mathcal{N}_u$. The mainstream idea of collaborative filtering is to learn collaborative embeddings $\boldsymbol{Z}$ from user-item interactions and then use it to predict user preferences [30–32]. From the generative model perspective, the collaborative embeddings can also be regarded as latent representations, from which the factual feedback can be generated [32]. The learning of collaborative embeddings can be formalized as maximizing the posterior

$$\arg\max_{\boldsymbol{Z}} p\big(\boldsymbol{Z}|\boldsymbol{X}^f\big). \tag{1}$$

With the development of the recommender system, its explainability received widespread attention. It is expected to provide explanations, such as "which interaction items play a decisive role in the final recommendation results." In this section, we additionally introduce "counterfactual feedback $\boldsymbol{X}^{\mathrm{cf}}$" to provide CEs by jointly modeling the factual and counterfactual feedback.

3.1 Overview

For each user, the item-based CE consists of a subset of interacted items, and it is expected to change user preference if removing this subset from the user's interaction records. To better understand the generation process of item-based CEs, we introduce the term "counterfactual set" $\mathcal{C}$ to denote the subset of interacted items selected to be removed. The resulting feedback is called counterfactual feedback $\boldsymbol{x}_u^{\mathrm{cf}}$. Due to the large number of subsets, especially when interactions are numerous, the counterfactual set (counterfactual feedback)[3)] is required to be continuously optimized to meet the goal of CEs. Ultimately, items in the final counterfactual set constitute item-based CEs.

By definition, the counterfactual feedback is derived from factual feedback. Thanks to the flexibility of generative models in defining the generation process [32,33], which provides a powerful way to capture the underlying structure and generative mechanism of data, we can get a complete view of how all variables relate to each other, allowing us to delve deeper into the causal relationships between variables. Thus, we aim to model this process from the generation process perspective and provide their joint distribution for each user u:

$$p\big(\boldsymbol{x}_u^{\mathrm{cf}}, \boldsymbol{x}_u^f\big) = \int p(\boldsymbol{z}_u) p\big(\boldsymbol{x}_u^{\mathrm{cf}}, \boldsymbol{x}_u^f|\boldsymbol{z}_u\big)\mathrm{d}\boldsymbol{z}_u. \tag{2}$$

Here, $\boldsymbol{z}_u$ is a fixed-dimensional vector, which can be regarded as the latent representation of user u. Then, to further portray the generation of counterfactual feedback, there are two questions to be answered: (1) which items are chosen to constitute the counterfactual set, and (2) why those items are selected. For this reason, we introduce two additional variables: counterfactual indicator $\boldsymbol{e}_u$ and importance $\boldsymbol{t}_u$. Counterfactual indicator $\boldsymbol{e}_u$ points out the index of items chosen to be removed from factual feedback.

3) Actually, there is a one-to-one correspondence between counterfactual feedback and set.

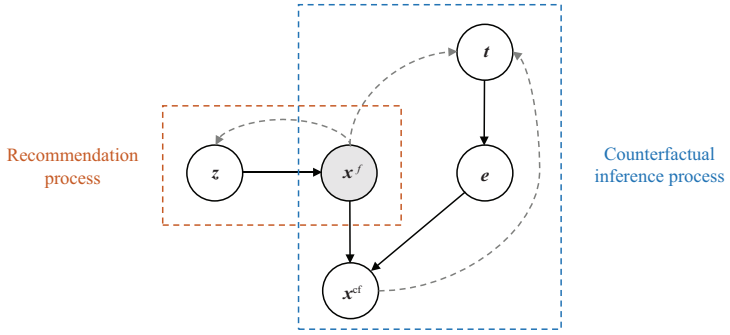

Figure 2 (Color online) Graphical model of LCER. The gray circle represents the observed variable. The solid and dashed arrows denote the generation and inference processes of latent variables, respectively.

Importance $\boldsymbol{t}_u$ depicts the effect of items with regard to CEs, where items with higher importance are more likely to be selected as a member of the counterfactual set. In this way, the overall generative process can be modeled as (illustrated in Figure 2)

$$\begin{aligned} p\left(\boldsymbol{x}_u^{\mathrm{cf}}, \boldsymbol{x}_u^f\right) &= \iiint p\left(\boldsymbol{x}_u^{\mathrm{cf}}, \boldsymbol{x}_u^f, \boldsymbol{z}_u, \boldsymbol{e}_u, \boldsymbol{t}_u\right) \mathrm{d}\boldsymbol{z}_u \mathrm{d}\boldsymbol{e}_u \mathrm{d}\boldsymbol{t}_u \\ &= \iiint \underbrace{p\left(\boldsymbol{x}_u^{\mathrm{cf}} | \boldsymbol{x}_u^f, \boldsymbol{e}_u\right) p\left(\boldsymbol{e}_u | \boldsymbol{t}_u\right) p\left(\boldsymbol{t}_u\right)}_{\mathcal{CI}:\ p\left(\boldsymbol{x}_u^{\mathrm{cf}} | \boldsymbol{x}_u^f\right)} \underbrace{p\left(\boldsymbol{x}_u^f | \boldsymbol{z}_u\right) p\left(\boldsymbol{z}_u\right)}_{\mathcal{RS}:\ p\left(\boldsymbol{x}_u^f\right)} \mathrm{d}\boldsymbol{z}_u \mathrm{d}\boldsymbol{e}_u \mathrm{d}\boldsymbol{t}_u. \end{aligned} \tag{3}$$

The entire process can be categorized into recommendation process $\mathcal{RS}$ and counterfactual inference process $\mathcal{CI}$. The terms in $\mathcal{RS}$ relate to the modeling of factual feedback $\boldsymbol{x}_u^f$. To solve this generative model, it is required to estimate the posterior distribution of $\boldsymbol{z}_u$ and attempt to reconstruct the factual feedback. Intuitively, it can be viewed as exploring the potential interests of users and predicting users' preferences. From this perspective, it can be regarded as recommendation process. The terms in $\mathcal{CI}$ relate to the modeling of counterfactual feedback $\boldsymbol{x}_u^{\mathrm{cf}}$ given factual feedback $\boldsymbol{x}_u^f$, and it can be viewed as a counterfactual inference process. It captures the intuition that counterfactual feedback can be inferred only if given users' factual feedback, which aims to provide CEs based on personalized recommendations. Following this intuition, we perform two processes separately in practice.

3.1.1 *Recommendation process*

To learn the general recommendation mechanism across different users, amortized variational inference [32] has been widely used to approximate the true posterior $p\left(\boldsymbol{z}_u | \boldsymbol{x}_u^f\right)$ [34–36]. Let $q\left(\boldsymbol{z}_u | \boldsymbol{x}_u^f\right)$ denote the variational posterior, and the evidence lower bound (ELBO) on the marginal log-likelihood of factual preference can be derived as

$$\log p\left(\boldsymbol{x}_u^f\right) = \log \int p\left(\boldsymbol{x}_u^f | \boldsymbol{z}_u\right) p\left(\boldsymbol{z}_u\right) \mathrm{d}\boldsymbol{z}_u \geqslant \mathbb{E}_{q\left(\boldsymbol{z}_u | \boldsymbol{x}_u^f\right)}\left[\log p\left(\boldsymbol{x}_u^f | \boldsymbol{z}_u\right)\right] - \mathrm{KL}\left(q\left(\boldsymbol{z}_u | \boldsymbol{x}_u^f\right) || p\left(\boldsymbol{z}_u\right)\right). \tag{4}$$

The posterior $q\left(\boldsymbol{z}_u | \boldsymbol{x}_u^f\right)$ and likelihood $p\left(\boldsymbol{x}_u^f | \boldsymbol{z}_u\right)$ can be parameterized by neural network with corresponding parameters ϕ and θ to make computationally efficient. In this way, $q_\phi(\boldsymbol{z}_u | \boldsymbol{x}_u^f)$ and $p_\theta(\boldsymbol{x}_u^f | \boldsymbol{z}_u)$ are also called encoder and decoder respectively.

Accompanied by maximizing the ELBO, users' factual preferences are fitted, and recommendation results can be derived. Specifically, the first term can be regarded as a reconstruction of factual feedback, and $p\left(\boldsymbol{x}_u^f | \boldsymbol{z}_u\right)$ outputs the interaction likelihood among the current user and all items. Considering the recommendation process as a random function g, the reconstructed preference can be denoted as $g\left(\boldsymbol{x}_u^f\right) \in \mathbb{R}^{M \times 1}$, where each element $g_i\left(\boldsymbol{x}_u^f\right)$ represents the interaction likelihood between user u and item i. By sorting the interaction likelihood of uninteracted items, the top-K recommendation list $\mathcal{R}_u$ can be derived and displayed to user u.

3.1.2 *Counterfactual inference process*

Given factual preference $\boldsymbol{x}_u^f$, the counterfactual preference $\boldsymbol{x}_u^{\mathrm{cf}}$ can be inferred via

$$\log p\big(\boldsymbol{x}_u^{\mathrm{cf}}|\boldsymbol{x}_u^f\big) = \log \iint p\big(\boldsymbol{x}_u^{\mathrm{cf}}|\boldsymbol{x}_u^f, \boldsymbol{e}_u\big)p(\boldsymbol{e}_u|\boldsymbol{t}_u)p(\boldsymbol{t}_u)\mathrm{d}\boldsymbol{e}_u\mathrm{d}\boldsymbol{t}_u \geqslant \mathbb{E}_{p(\boldsymbol{t}_u)p(\boldsymbol{e}_u|\boldsymbol{t}_u)} \log p\big(\boldsymbol{x}_u^{\mathrm{cf}}|\boldsymbol{x}_u^f, \boldsymbol{e}_u\big), \tag{5}$$

where $\boldsymbol{e}_u$ indicates whether interacted items belong to the counterfactual set. To further differentiate the importance of interacted items in terms of CEs, we introduce a learnable importance variable $\boldsymbol{t}_u$. The prior $p(\boldsymbol{t}_u)$ describes the initial distribution of $\boldsymbol{t}_u$, and the learning of $\boldsymbol{t}_u$ corresponds to approximate the posterior $p\big(\boldsymbol{t}_u|\boldsymbol{x}_u^f, \boldsymbol{x}_u^{\mathrm{cf}}\big)$. Furthermore, $p\big(\boldsymbol{x}_u^{\mathrm{cf}}|\boldsymbol{x}_u^f, \boldsymbol{e}_u\big)$ can reflect the properties of counterfactual feedback and will be tailored in Subsection 3.2.

By solving (5), we can obtain the ultima counterfactual feedback $\boldsymbol{x}_u^{\mathrm{cf}}$ and corresponding counterfactual indicator $\boldsymbol{e}_u$. The elements with a value of one in $\boldsymbol{e}_u$ indicate that the corresponding items belong to the counterfactual set. Therefore, the resulting counterfactual set $\mathcal{C} = \{j|e_{uj} = 1\}$ is exactly the generated CE. Note that the counterfactual preference for each user is not unique. For instance, several items will be recommended to each user after the recommendation process, whereas there is at least one counterfactual preference for each recommended item. Therefore, we can use $\mathcal{X}_u^{\mathrm{cf}}$ to denote the set of counterfactual preferences.

3.2 A concrete implementation

In the above subsection, we provide a general formulation of the joint distribution of factual and counterfactual feedback. Specifying the specific form of each probability can reflect the properties that factual and counterfactual feedback should have. To deepen the understanding of the entire process, we describe a concrete implementation of the proposed LCER in this subsection.

3.2.1 *Concrete implementation of recommendation process*

Considering the characteristics of implicit feedback, the factual feedback $\boldsymbol{x}_u^f$ is assumed to be sampled from multinomial distribution with probability $\pi(\boldsymbol{z}_u)$, i.e.,

$$\boldsymbol{x}_u^f \sim \mathrm{Mult}(N_u, \pi(\boldsymbol{z}_u)), \tag{6}$$

where $\boldsymbol{z}_u \sim \mathcal{N}(0, \boldsymbol{I}_d)$ is the latent representation of user u with Gaussian prior distribution, and $N_u = \sum_{i\in\mathcal{N}_u} x_{ui}^f$ is the total number of interactions. To further combine the powerful representation ability of neural networks, we use two multilayer perceptrons are utilized to approximate the variational posterior $q_\phi\big(\boldsymbol{z}_u|\boldsymbol{x}_u^f\big)$ and multinomial likelihood $p_\theta\big(\boldsymbol{x}_u^f|\boldsymbol{z}_u\big)$, respectively.

In this way, the recommendation process is designated as Mult-VAE [32]. We do not explore other likelihood, distribution or network architecture in this paper, but directly use Mult-VAE as an illustration because of its simplicity and effectiveness.

3.2.2 *Concrete implementation of counterfactual inference process*

To give a concrete description of the counterfactual inference process, we connect each counterfactual preference with a recommended item. In other words, we attempt to generate CEs for each recommended item.

Limited by the characteristics of implicit feedback in recommendation scenarios, each item has only two states for each user: interacted or uninteracted. Each interacted item contains only two states in terms of counterfactual set, belonging to or not. To identify whether each interacted item is a part of the counterfactual set, we introduce a binary indicator $\boldsymbol{e}_u \in \{0,1\}^{M\times 1}$, e.g., $e_{uj} = 1$ if item j belongs to counterfactual set[4)]. However, the importance of different items for generating CEs should be distinguished, which cannot be displayed in such a binary setting. To differentiate the importance of interacted items in terms of CEs, we introduce a learnable importance $\boldsymbol{t}_u \in \mathbb{R}^{M\times 1}$. The value of $\boldsymbol{t}_u$ is limited in $[0, 1]$, and its prior distribution $p(\boldsymbol{t}_u)$ is assumed to be a uniform distribution. According to the

4) Although not all items are interacted by user, we still endow an indicator for all items. The reason is that it is simple and straightforward to set the indicator to a fixed length, and this indicator acts on binary factual feedback; thus, uninteracted items will be filtered.

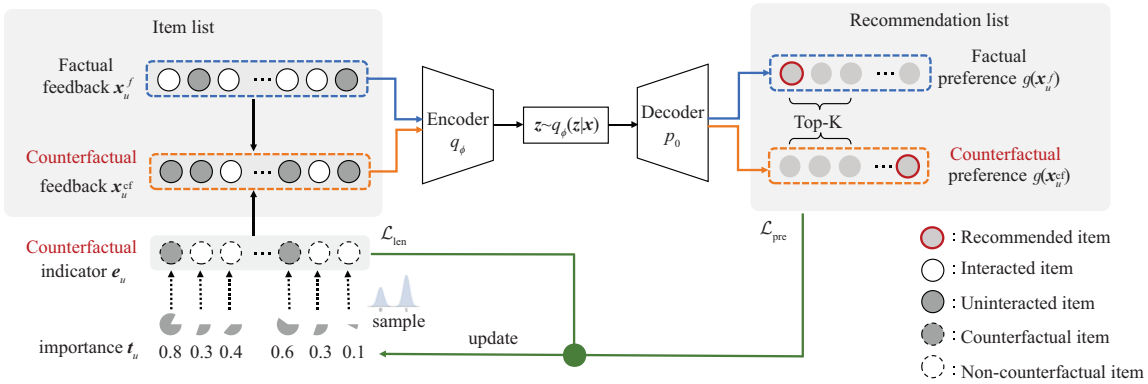

Figure 3 (Color online) Illustration of the counterfactual inference process. In this figure, CE is generated for user u and one of the recommended items in the top-K recommendation list. The selected recommended item is denoted by a gray circle with a red border. Blue and orange lines represent the original recommendation and counterfactual recommendation procedures, respectively, after removing the counterfactual set from the original interactions.

importance $\boldsymbol{t}_u$, the indicator $\boldsymbol{e}_u$ can be derived to indicate whether items belong to the counterfactual set. Formally, it is assumed to be sampled from multivariate Bernoulli distribution with probability $\boldsymbol{t}_u$:

$$\boldsymbol{e}_u \sim \text{Bernoulli}(\boldsymbol{t}_u), \tag{7}$$

where each element e_{uj} in $\boldsymbol{e}_u$ follows a Bernoulli distribution with probability t_{uj}.

Once obtained the indicator $\boldsymbol{e}_u$, the counterfactual set $\mathcal{C} = \{j|e_{uj} = 1\}$ can also be derived. Subsequently, the counterfactual feedback can be obtained as

$$\boldsymbol{x}_u^{\text{cf}} = (\mathbf{1}_{M\times 1} - \boldsymbol{e}_u) \odot \boldsymbol{x}_u^f, \tag{8}$$

where $\mathbf{1}_{M\times 1}$ denotes an $M \times 1$ vector whose elements are all 1, and $\odot$ represents element-wise product. Still considering the recommendation process as a random function g, the corresponding reconstructed counterfactual preference can also be denoted as $g(\boldsymbol{x}_u^{\text{cf}}) \in \mathbb{R}^{M\times 1}$, where each element $g_i(\boldsymbol{x}_u^{\text{cf}})$ represents the counterfactual interaction likelihood.

The ultimate goal of fitting counterfactual preference is to generate CEs. Thus, we use the goal of CEs to approximate the log-likelihood of counterfactual preference $\log p(\boldsymbol{x}_u^{\text{cf}}|\boldsymbol{x}_u^f, \boldsymbol{e}_u)$, which further guide the learning of importance $\boldsymbol{t}_u$. The overall framework is shown in Figure 3.

Goals of CEs. To be an eligible CE, the counterfactual set $\mathcal{C}$ should possess two properties: good quality and short length. On the one hand, the recommendation result should be flipped, i.e., the recommended item should topple out of the top-K recommendation list once the counterfactual set is removed from interactions. This goal can be achieved by constraining that user u's counterfactual preference for recommended item i should be smaller than a specific threshold. It is straightforward to define the threshold as the counterfactual preference of $(K+1)$-th item. However, the learning of importance $\boldsymbol{t}_u$ is influenced by $(K+1)$-th item in this way, which is unnecessary. Therefore, the threshold is defined as a factual preference for $(K+1)$-th item in the recommendation list, which can also reflect the objective of CEs to some extent and will not influence the learning of importance $\boldsymbol{t}_u$. It can be formally described as follows:

$$g_i(\boldsymbol{x}_u^{\text{cf}}) < g_{K+1}(\boldsymbol{x}_u^f), \tag{9}$$

where $g_i(\boldsymbol{x}_u^{\text{cf}})$ denotes the counterfactual preference for recommended item i, and $g_{K+1}(\boldsymbol{x}_u^f)$ represents the factual preference for $(K+1)$-th item in recommendation list.

On the other hand, to ensure the understandability and conciseness of CEs, the number of items in the counterfactual set should be as small as possible. It can be evaluated directly by

$$\mathcal{L}_{\text{len}} = \sum_j e_{uj}. \tag{10}$$

Optimization. It is worth noting that there are two challenges during the counterfactual inference process. One of the challenges is that the Bernoulli sampling process of $\boldsymbol{e}_u$ is not differentiable, so that importance $\boldsymbol{t}_u$ cannot be directly optimized during training. To address this problem, we use a

Algorithm 1 Learning algorithm of counterfactual inference process

Input: User u, factual feedback $\boldsymbol{x}_u^f$ of u, recommended item i to be explained, recommendation model g.
Output: Counterfactual set $\mathcal{C}$.
Compute the factual preference $g_{K+1}(\boldsymbol{x}_u^f)$ for $(K+1)$-th recommended item through g, given $\boldsymbol{x}_u^f$;
Initialize the importance $\boldsymbol{t}_u$;
while not convergence **do**
  Sample the counterfactual indicator $\boldsymbol{e}_u$ based on $\boldsymbol{t}_u$ according to Gumbel Softmax reparameterization;
  Obtain counterfactual feedback $\boldsymbol{x}_u^{\text{cf}}$ after removing the counterfactual set via (8);
  Get the counterfactual preference $g_i(\boldsymbol{x}_u^{\text{cf}})$ of item i through g, given $\boldsymbol{x}_u^{\text{cf}}$;
  Update importance $\boldsymbol{t}_u$ according to (12);
end while
if $g_i(\boldsymbol{x}_u^{\text{cf}}) < g_{K+1}(\boldsymbol{x}_u^f)$ **then**
  $\mathcal{C} = \{j | e_{uj} = 1\}$;
else
  $\mathcal{C} = \emptyset$;
end if
return $\mathcal{C}$.

reparameterization method, Gumbel Softmax [37], which allows the gradients to backpropagate through a discrete sampling process. Inspired by [26], we use hinge loss to relax the the constraint $g_i(\boldsymbol{x}_u^{\text{cf}}) < g_{K+1}(\boldsymbol{x}_u^f)$ so as to make it differentiable:

$$\mathcal{L}_{\text{pre}} = \max\left(0, \alpha + g_i\left(\boldsymbol{x}_u^{\text{cf}}\right) - g_{K+1}\left(\boldsymbol{x}_u^f\right)\right), \tag{11}$$

where α is a hyperparameter, which can be regarded as a slack variable to constrain the intensity of change in the preference for recommended items. It is worth noting that we use hinge loss here to relax the above constraint; therefore, it is not guaranteed that the above constraint is satisfied. Thus, it requires a post process to check whether $g_i\left(\boldsymbol{x}_u^{\text{cf}}\right) < g_{K+1}\left(\boldsymbol{x}_u^f\right)$ indeed. Additionally, there is a gap between counterfactual preference and factual preference of $(K+1)$-th item; thus, constraining $g_i\left(\boldsymbol{x}_u^{\text{cf}}\right) < g_{K+1}\left(\boldsymbol{x}_u^f\right)$ cannot definitely make sure recommended item i topple out of top-K recommendation list after removing counterfactual set. Fortunately, this problem can be alleviated very well by increasing α, which has been verified in experimental results.

Thus the overall objective can be formalized as

$$\mathcal{L} = \mathcal{L}_{\text{pre}} + \lambda \cdot \mathcal{L}_{\text{len}} = \max\left(0, \alpha + g_i\left(\boldsymbol{x}_u^{\text{cf}}\right) - g_{K+1}\left(\boldsymbol{x}_u^f\right)\right) + \lambda \cdot \sum_j e_{uj}, \tag{12}$$

where λ is the trade-off coefficient. The learning algorithm is summarized in Algorithm 1.

In this way, the learning of importance is under the guidance of the goal of CEs to ensure faithfulness of importance. Additionally, CEs can be learned with the learning of importance. By adjusting the value of λ, we can choose to generate more concise CEs or CEs with higher quality.

4 Experiments

In this section, a series of experiments are conducted on four datasets to verify the effectiveness of the proposed LCER, both quantitatively and qualitatively. As introduced in Subsection 3.2, the recommendation process is designated as Mult-VAE [32]; thus, we mainly show the performance of the counterfactual inference process.

4.1 Experimental setting

Datasets. To evaluate the effectiveness of LCER, we perform experiments on four publicly available datasets: MovieLens 100K (ML 100K), MovieLens 1M (ML 1M)[5], Alishop[6], and Epinions[7]. Among them, MovieLens is related to the movie domain, Alishop and Epinions corresponds to the online product domain, and the user/item scale and sparsity of these datasets vary a lot. The distinction among datasets can better demonstrate the effectiveness of LCER.

To construct implicit feedback, only ratings larger than 3.5 for ML 100K, ML 1M, and Epinions are retained, whereas interaction data in Alishop exactly satisfies implicit feedback setting. For all datasets,

5) https://grouplens.org/datasets/movielens/.
6) https://jianxinma.github.io/disentangle-recsys.html.
7) http://www.trustlet.org/downloaded_epinions.html.

Table 1 Statistics of the preprocessed datasets

Dataset	♯ Users	♯ Items	♯ Interactions	Density (%)
ML 100K	938	1447	55361	4.079
ML 1M	6034	3502	575235	2.722
Alishop	10668	20591	767493	0.350
Epinions	20634	115798	460451	0.019

users whose number of interactions is less than five are removed. For each dataset, we randomly select 100 test users, then LCER is carried out to generate CEs for each recommendation of these test users[8]. More detailed statistical information about these datasets is summarized in Table 1.

Metrics. We used four quantitative metrics to measure the effectiveness of CEs generated by LCER: validity, length, probability of sufficiency (PS), and probability of necessity (PN). Suppose there are N_t test users and K items are recommended for each test user. For each user u and a corresponding recommended item i, the set of items belonging to CEs is $\mathcal{C}_{u,i}$. (1) Validity (Val) measures the ratio of recommendations for which a CE can be found. Formally, Validity is generally defined as $\frac{\sum_{u=1}^{N_t}\sum_{i=1}^{K}\mathbb{I}(\mathcal{C}_{u,i}\neq\emptyset)}{N_t\times K}$, where $\mathbb{I}(\cdot)$ is an indicator function. It is expected to find explanations for as many recommendations as possible, so higher validity is better. (2) Length (Len) measures the average number of items in CEs, which can be expressed as Length $= \frac{\sum_{u=1}^{N_t}\sum_{i=1}^{K}|\mathcal{C}_{u,i}|}{\sum_{u=1}^{N_t}\sum_{i=1}^{K}\mathbb{I}(\mathcal{C}_{u,i}\neq\emptyset)}$, where $|\mathcal{C}_{u,i}|$ denotes the number of items in CE $\mathcal{C}_{u,i}$. A lower length is better because CEs are expected to be simple and concise so that users can understand better. (3) PS and (4) PN are used to measure the probability of a cause being sufficient cause or necessary cause [9], respectively. Following [26], sufficiency and necessity are in terms of a CE generated for user u and a recommended item i. Then, PS and PN evaluate the ratio of CEs that satisfy sufficiency and necessity, respectively. Specifically, sufficiency is defined as if items in CE were the only items interacted by the user, the item i would still be recommended. In contrast, necessity is defined as if items in CE were removed from interactions, the item i would not be recommended. To compute PS and PN, two feedbacks should be constructed. Let $\tilde{\boldsymbol{x}}_u$ denotes feedback after removing items in CEs (referring to necessity), $\boldsymbol{x}_u^*$ denotes feedback containing only items in CEs (referring to sufficiency), and corresponding recommendation set is $\tilde{\mathcal{R}}_u$ and $\mathcal{R}_u^*$ respectively. Among them, $\tilde{\boldsymbol{x}}_u$ can be computed via (8) directly, $\boldsymbol{x}_u^*$ can be computed through $\boldsymbol{x}_u^* = \boldsymbol{e}\odot\boldsymbol{x}_u$, then $\tilde{\mathcal{R}}_u$ and $\mathcal{R}_u^*$ can be obtained by feeding corresponding feedback into recommendation model g. Thus the formalization of PS is

$$\mathrm{PS} = \frac{\sum_{u=1}^{N_t}\sum_{i=1}^{K}\mathbb{I}(i\in\mathcal{R}_u^*)}{\sum_{u=1}^{N_t}\sum_{i=1}^{K}\mathbb{I}(\mathcal{C}_{u,i}\neq\emptyset)}. \tag{13}$$

The denominator is the number of CEs generated by algorithm, and the numerator denotes the number of CEs which are sufficient. Similarly, PN can be formalized as

$$\mathrm{PN} = \frac{\sum_{u=1}^{N_t}\sum_{i=1}^{K}\mathbb{I}(i\notin\tilde{\mathcal{R}}_u)}{\sum_{u=1}^{N_t}\sum_{i=1}^{K}\mathbb{I}(\mathcal{C}_{u,i}\neq\emptyset)}. \tag{14}$$

Additionally, the harmonic mean of PS and PN is also shown to exhibit the performance, which is $F_{\mathrm{NS}} = \frac{2\cdot\mathrm{PN}\cdot\mathrm{PS}}{\mathrm{PN}+\mathrm{PS}}$. As discussed in [22], necessity and sufficiency are complementary to judge the effectiveness of explanations. Thus not only PS and PN are expected to be higher, but also their harmonic mean.

Baselines. To demonstrate the performance, LCER is compared with an item-based explanation method and two state-of-the-art item-based CE methods. The research on CE in the field of recommendation is just emerging; thus, research is still few in number. Among them, Prince [17], Accent [7] and method proposed by Kaffes et al. [18] (called KaffesCE) focus on item-based CEs. However, Prince generated CEs depending on the graph structure, whose application scenario is limited. Thus, KaffesCE and Accent are used as baselines in the following experiments. Except for CE methods, FIA [23] is a method that provides item-based explanations based on the concept of counterfactual reasoning. However, explanations generated by FIA do not ensure that the recommendation result would change if one interaction was removed. Therefore, they are not CEs. We choose FIA as a baseline to compare the

8) When performing recommendation, there are 100 users for ML 100K, 500 users for ML 1M, 2000 users for Alishop, and 2000 users for Epinions in validation and test set respectively.

Table 2 Quantitative performance of LCER and baselines under different top-K recommendation results

Dataset	Method	Top-5					Top-10				
		Val (%)	Len	PS (%)	PN (%)	F_{NS} (%)	Val (%)	Len	PS (%)	PN (%)	F_{NS}(%)
ML 100K	FIA	–	**1.000**	35.54	27.47	30.61	–	**1.000**	42.52	23.73	30.47
	KaffesCE	68.0	1.691	47.06	36.76	41.28	65.8	1.948	55.92	39.36	46.20
	Accent	87.8	3.431	39.64	60.82	47.99	86.0	3.857	47.67	48.84	48.25
	LCER	**100.0**	4.461	**55.56**	**61.41**	**58.34**	**88.5**	3.361	**66.63**	**51.89**	**58.34**
	Improvement	13.90%	–	18.06%	0.97%	21.57%	2.91%	–	19.15%	6.24%	20.91%
ML 1M	FIA	–	**1.000**	25.80	34.40	29.49	–	**1.000**	34.10	28.60	31.11
	KaffesCE	56.8	1.482	41.20	57.04	47.84	56.3	1.622	49.02	58.97	53.54
	Accent	97.0	3.303	34.43	58.35	43.31	96.2	3.418	45.43	44.59	45.01
	LCER	**97.2**	3.685	**59.67**	**64.20**	**61.85**	**96.8**	4.092	**70.04**	**63.84**	**66.80**
	Improvement	0.21%	–	44.83%	10.03%	29.29%	0.62%	–	42.88%	8.26%	24.77%
Alishop	FIA	–	**1.000**	9.80	33.40	15.15	–	**1.000**	15.30	28.80	19.98
	KaffesCE	60.8	1.457	19.08	57.24	28.62	59.1	1.387	25.72	52.45	34.51
	Accent	99.2	3.028	15.93	57.46	24.94	98.1	3.060	23.45	45.36	30.91
	LCER	**99.6**	3.135	**31.53**	**64.26**	**42.29**	**99.0**	3.021	**42.12**	**55.45**	**47.88**
	Improvement	0.40%	–	65.25%	11.83%	47.76%	0.92%	–	63.76%	5.72%	38.74%
Epinions	FIA	–	**1.000**	32.26	58.06	41.47	–	**1.000**	33.55	50.00	40.15
	KaffesCE	58.2	1.306	40.21	58.08	47.52	58.1	1.361	45.27	52.67	48.69
	Accent	58.4	1.527	38.36	71.92	50.03	57.8	1.528	36.85	60.03	45.67
	LCER	**100.0**	2.074	**49.53**	**73.23**	**58.97**	**99.5**	1.551	**53.33**	**62.60**	**57.60**
	Improvement	71.23%	–	23.18%	1.82%	17.87%	71.26%	–	17.80%	4.28%	18.30%

quantitative performance[9)].

Parameter settings. In our experiments, we consider top-5,10 ($K = 5, 10$) recommendations because users usually focus only on fewer items that are ranked higher. Unless otherwise specified, K is set as five. The parameters of all baselines we compared with are either adopted from their original papers or determined by experiments. In particular, the replacement item in Accent is set as $(K+1)$-th item in the original recommendation list to ensure consistency. The hyperparameters α and λ are tuned according to grid search. Concretely, margin α is tuned amongst {0.02, 0.04, 0.06, 0.08, 0.1}, and trade-off coefficient λ is selected from {0.01, 0.02, 0.03, 0.04, 0.05}. Without specification, we show the results of ML 100K with $\alpha = 0.1$ and $\lambda = 0.01$, ML 1M with $\alpha = 0.04$ and $\lambda = 0.02$, Alishop with $\alpha = 0.08$ and $\lambda = 0.03$, as well as Epinions with $\alpha = 0.1$ and $\lambda = 0.01$.

4.2 Quantitative performance

In this subsection, we investigate the performance of LCER quantitatively. Table 2 presents the experimental results of LCER and baselines on four datasets. The best and second results are marked in bold and underlined, respectively. It is worth noting that there is no single metric that can reflect the quality of CEs, and all metrics need to be considered together. From Table 2, we have following observations, which are consistent under top-5 and top-10 recommendation:

• Although FIA can find the shortest-length explanation for all recommendation results, the quality cannot be guaranteed. It cannot also determine what the appropriate length for an explanation should be. Item-based explanation methods struggle to adjust the length of each individual explanation. Additionally, mediocre-quality explanations presented to users may backfire and lead to user dissatisfaction. Thus, we mainly focus on performance comparison among CE methods as follows.

• The validity of LCER outperforms all baselines, where validity reflects the proportion of recommendations for which a CE can be found. The validity of KaffesCE is the lowest across all datasets, indicating that the heuristic strategy only works well on the part of recommendations. Thus, CEs need to be analyzed case by case. Benefiting from the influence score computed for every interacted item, Accent can distinguish the importance of interacted items and give guidance for following the search process. Therefore, it can generate CEs for most recommendations on most datasets, except for Epinions. The possible

9) Unfortunately, FIA and Accent can hardly be applied to recommender systems where the amount of model parameters depends on the number of users/items. Thus, in our experiments, we approximate the influence score of an item by the difference between prediction scores before and after removing the item from the interaction set.

Table 3 Performance comparison with baselines when their lengths are close[a)]

Method	Val (%)	Len	PS (%)	PN (%)	F_{NS} (%)
KaffesCE	58.2	**1.306**	40.21	58.08	47.52
LCER	**79.2**	1.315	**48.54**	**63.59**	**55.06**
Accent	58.4	**1.527**	38.36	**71.92**	50.03
LCER	**100**	1.535	**49.68**	68.39	**57.55**

a) The best result is marked in bold.

reason for performing poorly on Epinions is the huge number of items, such that the influence score may not be discriminative enough to guide the search process. The validity of the proposed LCER is almost 100% across all datasets, reflecting the superiority of learning the importance of interacted items under the guidance of objective loss, compared to precomputation.

• Better CEs can be generated at the cost of the length, where length indicates the average number of items in CEs. In real-world scenarios, users prefer meaningful but slightly long explanations rather than short but worthless explanations. Although the length of KaffesCE is the smallest across all datasets, it can only generate CEs for part of recommendations. In contrast, CEs generated by Accent are slightly longer, but the validity improves a lot. It is noteworthy that the proposed LCER can generate more sufficient and necessary CEs for almost every recommendation, with a slightly longer counterfactual set or even with the same length.

• Both PS and PN of the proposed LCER show great superiority across all datasets. Comparing KaffesCE and Accent, KaffesCE shows better sufficiency stably, whereas Accent exhibits better necessity. In our opinion, the underlying reason is that the influence score in Accent is computed based on counterfactual reasoning, which focuses more on the necessity of CEs [22]. Consequently, a blind focus on necessity may result in less sufficient CEs. Furthermore, the PS and PN of the proposed LCER perform better than KaffesCE and Accent across all datasets, especially PS. As introduced in [22], a good explanation should be sufficient and necessary simultaneously. LCER can greatly improve the sufficiency of CEs while ensuring the necessity of CEs. This phenomenon demonstrates the superiority of a learning-based strategy for generating CEs compared with a search strategy.

We further present the quality of explanations compared with baselines when their lengths are close (with minor errors). The results of Epinions are presented in Table 3. We can observe that under similar lengths, LCER mostly achieves comparative performance with baselines. The PN of Accent is pretty high; the possible reason is that Accent computes importance of interacted items based on the thought of necessity exactly. Additionally, existing CE methods cannot adjust the length of explanations, and their results are unique. In contrast, our method can adjust the ratio of quality to length in the goal (i.e., loss function) of CE, validating the flexibility of our method.

4.3 Qualitative performance

In this subsection, we demonstrate the performance of LCER qualitatively from two perspectives: raising users' understanding and revealing biases of the recommendation model. First, we show some realistic explanations generated by LCER and baselines, from which differences in CEs generated by different methods are perceived. Moreover, we count the frequency and compute the average importance of each item being a part of CEs. Accordingly, CEs can also be used to reveal the unintended biases of the recommendation model.

Raising users' understanding of recommendation results. To give a qualitative feel of the CEs generated by LCER and baselines, we provide some anecdotal examples in Figures 4(a) and (b) for ML 100K and Alishop, respectively. For each dataset, two examples are displayed and category information is used to validate CEs. Categories in ML 100K reflect the genres of movies, whereas categories in Alishop are some numbers whose realistic meaning is not given. From Figures 4(a) and (b), we have several observations. Search strategies used in KaffesCE and Accent may fail to generate CEs sometimes, whereas LCER can always find CEs (as shown in Example 1 of Figure 4(a) and Example 3 of Figure 4(b)). When the lengths of CEs are the same, CEs generated by the proposed LCER are more similar to recommended items in terms of categories (as shown in Example 1 of Figure 4(a) and Example 1 of Figure 4(b)). Although CEs generated by LCER and Accent all belong to Category 3, the specific meaning of LCER is closer to the recommended item than Accent. Sometimes, LCER can generate better CEs at the expense of length (as shown in Example 2 of Figure 4(a)). When search strategies cannot find

Example 1:
Recommended movie: L.A. Confidential
Category: Drama / Mystery / Thriller / Crime

Methods	Counterfactual Explanation	Category
LCER	The Full Monty	Drama / Comedy
	Gattaca	Drama / Sci-Fi / Thriller
KaffesCE	Rosewood	Drama / Action / History
	The Full Monty	Drama / Comedy
Accent	none	none

Example 2:
Recommended movie: Schindler's List
Category: Drama / History / War

Methods	Counterfactual Explanation	Category
LCER	The English Patient	Drama / Romance / War
	The Killing Fields	Drama / History / War
	Patton	Drama / Biography / War
KaffesCE	The English Patient	Drama / Romance / War
	The Shawshank Redemption	Drama / Crime
Accent	Absolute Power	Drama / Thriller / Crime
	Pulp Fiction	Drama / Comedy / Crime

(a)

Example 1:
Recommended product: frozen cuttlefish
Category: 3

Methods	Counterfactual Explanation	Category
LCER	beef offal	3
	beef rolls	3
KaffesCE	beef offal	3
	screen protector	5
Accent	beef offal	3
	orange	3

Example 2:
Recommended product: pencil
Category: 2

Methods	Counterfactual Explanation	Category
LCER	notebook	2
	tablet case	5
	notebook	2
	rollerball pen	2
KaffesCE	none	none
Accent	toothpaste	4
	notebook	2
	… (10 products in total)	…
	tablet case	5

(b)

Figure 4 (Color online) An illustration of CEs for (a) ML 100K and (b) Alishop.

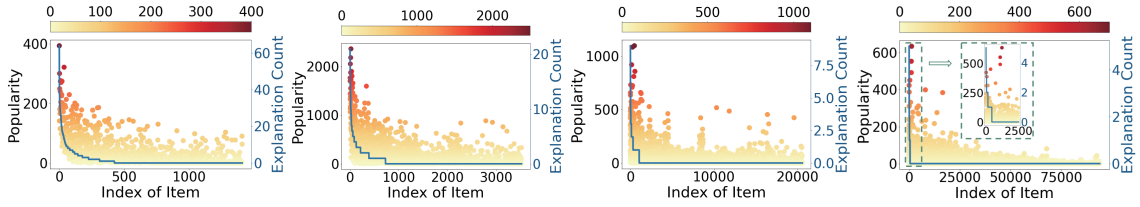

Figure 5 (Color online) Relationship between explanation count and popularity of each item. (a) ML 100K; (b) ML 1M; (c) Alishop; (d) Epinions. The solid line indicates the explanation count sorted in descending order, and the scatters describe the popularities of corresponding items. Note that some popular items do not belong to any CEs in Epinions. This is because only 100/2000 test users are selected to demonstrate the performance; thus, some popular items may not be interacted by these users due to the extreme sparsity of Epinions.

CEs or the length of CEs is too long, LCER may still find a smaller and more meaningful CE (as shown in Example 3 of Figure 4(b)). Thus, the above observations can intuitively confirm the effectiveness of CEs generated by LCER.

Revealing biases of recommendation model. As described in Section 3, not only whether an item belonging to CEs can be obtained by LCER, but its importance in terms of CEs is also revealed. To gain a deeper understanding of the generated CEs, we first count the number of times that each item is a part of CEs (called "explanation count" hereinafter). The relationship between explanation count and popularity is then shown in Figure 5. It shows that items that appear more frequently in CEs are more popular. For every item belonging to CEs, we compute the average importance (called "mean importance" hereinafter). Figure 6 shows the relationship between mean importance and popularity. Similarly, most items with a higher importance of being a part of CE are more popular. Additionally, there are two phenomena different from Figure 5. First, the small set of items, with the highest importance being a part of CEs, are not the most popular items. This is because these items belong to a fraction of CEs or even only one CE with high importance, whereas popular items may belong to lots of CEs with varying importance. However, this phenomenon does not affect the judgment of the overall trend. Moreover, there is a sharp drop at importance 0.5, down to 0. This phenomenon is caused by the Gumbel Softmax reparameterization process [37], in which only items with importance greater than 0.5 are regarded as CEs.

The above experimental results reveal that the frequency and importance of items in CEs are highly correlate with item popularity. In other words, the recommended items are more related to popular items. This is because the CEs of the proposed LCER are generated under the guide of their influence

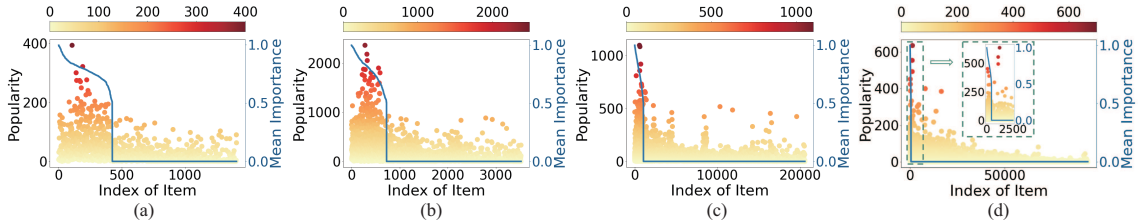

Figure 6 (Color online) Relationship between mean probability and popularity of each item. (a) ML 100K; (b) ML 1M; (c) Alishop; (d) Epinions. The remarks are similar to Figure 5.

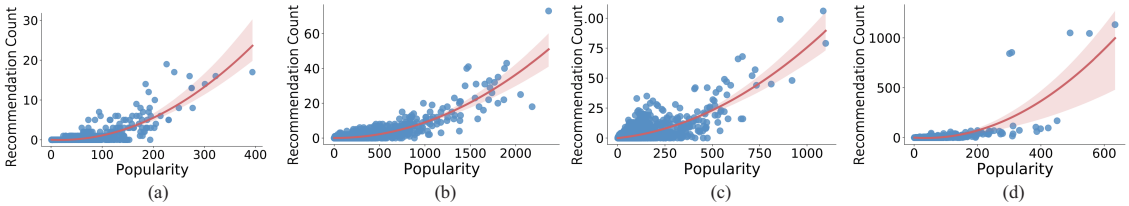

Figure 7 (Color online) Relationship between recommendation count and popularity of each item. Each scatter represents an item, and the red line represents the regression line of the scatter. (a) ML 100K; (b) ML 1M; (c) Alishop; (d) Epinions.

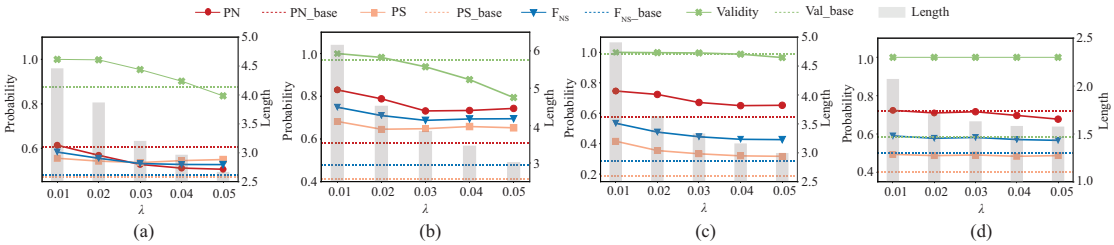

Figure 8 (Color online) Effect of λ. (a) ML 100K; (b) ML 1M; (c) Alishop; (d) Epinions. The solid lines in different colors indicate the trends of PN, PS, and F_{NS} of LCER; the dashed lines of corresponding color denote the best results in baselines; the histogram represents the trends of the length of LCER as λ increases.

on recommendation results. This phenomenon reflects the popularity bias of the recommendation model to some extent. To verify this conjecture, we also compute the number of times that each item is recommended (called "recommendation count") in the recommendation model. Figure 7 shows the relationship between recommendation count and item popularity. We observe that items that are recommended more are more popular, which is consistent with the above observations. Therefore, the conjecture is confirmed, i.e., CEs generated by LCER reveal the popularity bias of the recommendation model to some extent.

4.4 Effect of hyperparameters

In this subsection, we conduct a series of experiments to investigate the effect of hyperparameters.

Effect of λ. λ is the trade-off coefficient in the objective function, and it acts on the length constraint of CEs. Figure 8 shows the effect of λ on performance. Since λ acts on the length constraint, the effect of this constraint enhances as λ increases, thus decreasing the corresponding number of items in CEs. In contrast, the constraint of the recommendation result after removing CEs from original interactions weakens when λ increases; thus, metrics measuring the effectiveness of CEs decrease, i.e., validity, PS, PN, and F_{NS} all decline as λ increases.

Effect of α. For user u and a corresponding recommended item i, α constrains the intensity of change between factual preference of $(K+1)$-th item $g_{K+1}(\boldsymbol{x}_u^f)$ and preference of recommended item i after removing counterfactual set from interactions $g_i(\boldsymbol{x}_u^{\mathrm{cf}})$. Figure 9 shows the change in quantitative metrics for varying α. It is known that the positive impact of α contains two facets. However, hinge loss is used to relax constraint $g_i(\boldsymbol{x}_u^{\mathrm{cf}}) < g_{K+1}(\boldsymbol{x}_u^f)$ in LCER. Although hinge loss cannot guarantee that the constraint is satisfied, it can be alleviated by adding α. In this way, the validity of CEs increases. In experiments, the validity of LCER increases slightly as α increases, and the value of validity is almost 100% under each hyperparameter setting. Thus, the change in validity is not shown in Figure 9. However, even if $g_i(\boldsymbol{x}_u^{\mathrm{cf}}) < g_{K+1}(\boldsymbol{x}_u^f)$ is satisfied, it is not sure whether recommended item i is removed from top-K

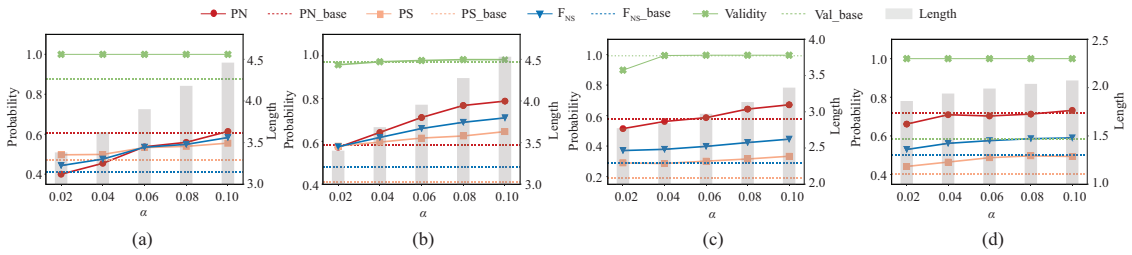

Figure 9 (Color online) Effect of α. (a) ML 100K; (b) ML 1M; (c) Alishop; (d) Epinions. The remarks are similar with Figure 8.

recommendation list after removing counterfactual set. This is because $g_{K+1}(\boldsymbol{x}_u^f)$ is not equal to $(K+1)$-th item in recommendation list after removal, i.e., $g_{K+1}(\boldsymbol{x}_u^{\mathrm{cf}})$. By adding α, it can also be alleviated to improve the sufficiency and necessity of CEs. In contrast, there is always a cost of length to generate better CEs. Thus, the number of items in CEs gets larger as α increases.

Therefore, the effectiveness and conciseness of CEs should be considered, and $\{\lambda, \alpha\}$ should be considered together so that generated CEs can be accepted furthest by users. In practical applications, meaningful but slightly longer explanations are more accepted by users instead of short but worthless explanations. Therefore, the choices of λ and α generate CEs that are as short as possible on the premise of being more effective.

5 Conclusion and future work

In this study, we propose LCER to improve trustworthiness and user satisfaction by providing more faithful CEs. In LCER, each interacted item is accompanied by a learnable importance weight, with which the consistency between the computation of importance and generation of CEs can be ensured. Moreover, the trade-off between quality and length of CEs can be customized. Extensive experiments demonstrate that the proposed LCER is superior to the state-of-the-art item-based CE methods for recommendation, both quantitatively and qualitatively.

In addition to the explainability of LCER, we are also able to discover underlying biases of recommendation models through LCER by analyzing the experimental results. It will be an interesting topic on how to improve recommender systems based on the result of LCER. Furthermore, the combination of generating explanations and improving recommendation performance in a holistic method is even more worthwhile to be considered.

Acknowledgements This work was partly supported by National Natural Science Foundation of China (Grant No. 62176020), National Key Research and Development Program of China (Grant No. 2020AAA0106800), Joint Foundation of the Ministry of Education (Grant No. 8091B042235), Beijing Natural Science Foundation (Grant No. L211016), Fundamental Research Funds for the Central Universities (Grant No. 2019JBZ110), and Chinese Academy of Sciences (Grant No. OEIP-O-202004).

References

1 Shirky C. It's not information overload. It's filter failure. 2008. https://mascontext.com/issues/information/its-not-information-overload-its-filter-failure
2 Zhang Y, Chen X. Explainable recommendation: a survey and new perspectives. FNT Inf Retrieval, 2020, 14: 1–101
3 Balog K, Radlinski F, Arakelyan S. Transparent, scrutable and explainable user models for personalized recommendation. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019. 265–274
4 Xin X, He X, Zhang Y, et al. Relational collaborative filtering: modeling multiple item relations for recommendation. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019. 125–134
5 Chen T, Yin H, Ye G, et al. Try this instead: personalized and interpretable substitute recommendation. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020. 891–900
6 Wang N, Wang H, Jia Y, et al. Explainable recommendation via multi-task learning in opinionated text data. In: Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, 2018. 165–174
7 Tran K H, Ghazimatin A, Saha Roy R. Counterfactual explanations for neural recommenders. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021. 1627–1631
8 Xue F, He X, Wang X, et al. Deep item-based collaborative filtering for top-n recommendation. ACM Trans Inf Syst, 2019, 37: 1–25
9 Pearl J, Mackenzie D. The Book of Why: the New Science of Cause and Effect. New York: Basic Books, 2018
10 Chou Y L, Moreira C, Bruza P, et al. Counterfactuals and causability in explainable artificial intelligence: theory, algorithms, and applications. Inf Fusion, 2022, 81: 59–83
11 Oh J M, Venters C C, Di C, et al. U1 snRNP regulates cancer cell migration and invasion in vitro. Nat Commun, 2020, 11: 1–9

12 Wachter S, Mittelstadt B, Russell C. Counterfactual explanations without opening the black box: automated decisions and the GDPR. Harvard J Law Technol, 2017, 31: 841
13 Mahajan D, Tan C, Sharma A. Preserving causal constraints in counterfactual explanations for machine learning classifiers. In: Proceedings of the NeurIPS Workshop on "Do the right thing": Machine Learning and Causal Inference for Improved Decision Making, 2019
14 Mothilal R K, Sharma A, Tan C. Explaining machine learning classifiers through diverse counterfactual explanations. In: Proceedings of the Conference on Fairness, Accountability, and Transparency, 2020. 607–617
15 Pawelczyk M, Broelemann K, Kasneci G. Learning model-agnostic counterfactual explanations for tabular data. In: Proceedings of the Web Conference 2020, 2020. 3126–3132
16 Freiesleben T. The intriguing relation between counterfactual explanations and adversarial examples. Minds Mach, 2022, 32: 77–109
17 Ghazimatin A, Balalau O, Roy R S, et al. Prince: provider-side interpretability with counterfactual explanations in recommender systems. In: Proceedings of the 13th International Conference on Web Search and Data Mining, 2020. 196–204
18 Kaffes V, Sacharidis D, Giannopoulos G. Model-agnostic counterfactual explanations of recommendations. In: Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization, 2021. 280–285
19 Laugel T, Lesot M J, Marsala C, et al. Issues with post-hoc counterfactual explanations: a discussion. In: Proceedings of the ICML Workshop on Human in the Loop Learning, 2019
20 Ribeiro M T, Singh S, Guestrin C. "Why should I trust you?" Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016. 1135–1144
21 Lundberg S M, Lee S I. A unified approach to interpreting model predictions. 2017. ArXiv:1705.07874
22 Mothilal R K, Mahajan D, Tan C, et al. Towards unifying feature attribution and counterfactual explanations: different means to the same end. In: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, 2021. 652–663
23 Cheng W, Shen Y, Huang L, et al. Incorporating interpretability into latent factor models via fast influence analysis. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019. 885–893
24 Koh P W, Liang P. Understanding black-box predictions via influence functions. In: Proceedings of International Conference on Machine Learning, 2017. 1885–1894
25 Nikolakopoulos A N, Karypis G. Recwalk: nearly uncoupled random walks for top-n recommendation. In: Proceedings of the 12th International Conference on Web Search and Data Mining, 2019. 150–158
26 Tan J, Xu S, Ge Y, et al. Counterfactual explainable recommendation. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021. 1784–1793
27 Zhong J, Negre E. Shap-enhanced counterfactual explanations for recommendations. In: Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing, 2022. 1365–1372
28 Xin Y, Jaakkola T. Controlling privacy in recommender systems. In: Proceedings of the 27th International Conference on Neural Information Processing Systems, 2014
29 Werder K, Ramesh B, Zhang R S. Establishing data provenance for responsible artificial intelligence systems. ACM Trans Manage Inf Syst, 2022, 13: 1–23
30 Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems. Computer, 2009, 42: 30–37
31 He X, Liao L, Zhang H, et al. Neural collaborative filtering. In: Proceedings of the Web Conference 2017, 2017. 173–182
32 Liang D, Krishnan R G, Hoffman M D, et al. Variational autoencoders for collaborative filtering. In: Proceedings of the Web Conference 2018, 2018. 689–698
33 Kingma D P, Welling M. Auto-encoding variational bayes. In: Proceedings of International Conference on Learning Representations, 2014
34 Yu X, Zhang X, Cao Y, et al. VAEGAN: a collaborative filtering framework based on adversarial variational autoencoders. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence, 2019. 4206–4212
35 Kim D, Suh B. Enhancing VAEs for collaborative filtering: flexible priors & gating mechanisms. In: Proceedings of the 13th ACM Conference on Recommender Systems, 2019. 403–407
36 Shenbin I, Alekseev A, Tutubalina E, et al. RecVAE: a new variational autoencoder for top-n recommendations with implicit feedback. In: Proceedings of the 13th International Conference on Web Search and Data Mining, 2020. 528–536
37 Jang E, Gu S, Poole B. Categorical reparametrization with gumble-softmax. In: Proceedings of International Conference on Learning Representations, 2017