CauseRec: Counterfactual User Sequence Synthesis for Sequential Recommendation

Shengyu Zhang*,
Zhejiang University, China
sy_zhang@zju.edu.cn

Dong Yao*,
Zhejiang University, China
yadongai@zju.edu.cn

Zhou Zhao†,
Zhejiang University, China
zhaozhou@zju.edu.cn

Tat-seng Chua
National University of Singapore
dcsccts@nus.edu.sg

Fei Wu†
Zhejiang University, China
wufei@zju.edu.cn

ABSTRACT
Learning user representations based on historical behaviors lies at the core of modern recommender systems. Recent advances in sequential recommenders have convincingly demonstrated high capability in extracting effective user representations from the given behavior sequences. Despite significant progress, we argue that solely modeling the observational behaviors sequences may end up with a brittle and unstable system due to the noisy and sparse nature of user interactions logged. In this paper, we propose to learn accurate and robust user representations, which are required to be less sensitive to (attack on) noisy behaviors and trust more on the indispensable ones, by modeling counterfactual data distribution. Specifically, given an observed behavior sequence, the proposed CauseRec framework identifies dispensable and indispensable concepts at both the fine-grained item level and the abstract interest level. CauseRec conditionally samples user concept sequences from the counterfactual data distributions by replacing dispensable and indispensable concepts within the original concept sequence. With user representations obtained from the synthesized user sequences, CauseRec performs contrastive user representation learning by contrasting the counterfactual with the observational. We conduct extensive experiments on real-world public recommendation benchmarks and justify the effectiveness of CauseRec with multi-aspects model analysis. The results demonstrate that the proposed CauseRec outperforms state-of-the-art sequential recommenders by learning accurate and robust user representations.

KEYWORDS
Sequential Recommendation, User Modeling, Contrastive Learning, Counterfactual Representation

ACM Reference Format:
Shengyu Zhang, Dong Yao, Zhou Zhao, Tat-seng Chua, Fei Wu. 2021. CauseRec: Counterfactual User Sequence Synthesis for Sequential Recommendation. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR ’21), July 11–15, 2021, Virtual Event, Canada. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3404835.3462908

1 INTRODUCTION
Due to the overwhelming data that people are facing on the Internet, personalized recommendation has become vital for retrieving information and discovering content. Accurately characterizing and representing users plays a vital role in a successful recommendation framework. Since users’ historical interactions are sequentially dependent and by nature time-evolving, recent advances [40, 42, 43, 48, 57–60, 63, 71, 74, 82] pay attention to sequential recommendation, which captures the current and recent preference by exploiting the sequentially modeled user-item interactions.

A sequential recommender aims to predict the next item a user might interact with based on the historical interactions. The challenging and open-ended nature of sequence modeling lends itself to a variety of diverse models. Traditional methods mainly exploit Markov chains [15] and factorization machines [22, 49] to capture lower-order sequential dependencies. Following these works, the higher-order Markov Chain and RNN (Recurrent Neural Network) [18, 21, 66] are proposed to model the complex high-order sequential dependencies. More recently, MIND is proposed to transform the historical interactions into multiple interest vectors using the capsule network [51]. ComiRec [5] differs from MIND by leveraging the attention mechanism and introducing a factor to control the balance of recommendation accuracy and diversity.

Despite significant progress made with these frameworks, there are some challenges demanding further explorations. A vital challenge comes from the noisy nature of implicit feedback. Due to the ubiquitous distractions that may affect the users’ first impressions (such as caption bias [39], position bias [24], and sales promotions), there are inconsistencies between users’ interest and their clicking behaviors, known as the natural noise [45]. Another challenge relates to the deficiency of existing methods in confronting data sparsity problem in recommender systems where users in general
only interact with a limited number of items compared with the item gallery which can easily reach 100 million in large live systems. Therefore, solely modeling the observational behavior sequences that can be both sparse and noisy may end up with a brittle system that is less satisfactory. To this end, learning accurate and robust users’ user representations is essential for recommender systems.

In this paper, we propose Counterfactual User Sequence Synthesis for Sequential Recommendation, abbreviated as CauseRec. The essence of CauseRec in confronting the data sparsity problem is to model the counterfactual data distribution rather than the observational sparse data distribution where the latter can be a subset of the former one, as shown in Figure 1. We mainly aim to answer the counterfactual question, “what the user representation would be if we intervene on the observed behavior sequence?”. Specifically, given the observed behavior sequence, we identify indispensable/disposable concepts at both the fine-grained item level and the abstract interest level. A concept indicates a certain aspect of user interest/preference. We perform counterfactual transformations on both the item-level and the interest-level user concept sequences. We obtain counterfactually positive user representation by replacing indispensable concepts, and counterfactually negative user representation by replacing dispensable concepts. To learn accurate and robust user representations, we propose to conduct contrastive learning between: 1) the observational and the counterfactual user representations; and 2) the user representations and the target items. Contrast with such out-of-distribution hard negatives potentially makes the learned representations robust since they are less sensitive to dispensable/noisy concepts. Contrast with such out-of-distribution positives potentially makes the learned representations accurate since they will trust more on the indispensable/disposable concepts at both the fine-grained item level and the abstract interest level. A concept indicates a certain aspect of user interest/preference. To learn accurate and robust user representations, we propose to conduct contrastive learning between: 1) the observational and the counterfactual user representations; and 2) the user representations and the target items. Contrast with such out-of-distribution hard negatives potentially makes the learned representations robust since they are less sensitive to dispensable/noisy concepts. Contrast with such out-of-distribution positives potentially makes the learned representations accurate since they will trust more on the indispensable/disposable concepts that are better representing user’s interest.

We conduct in-depth experiments to validate the effectiveness of the proposed CauseRec architectures on various public recommendation datasets. With a naive deep candidate generation (or matching) architecture as the baseline method, CauseRec outperforms SOTA sequential recommenders for deep candidate generation. We conduct comprehensive model analysis to uncover how different building blocks and hyper-parameters affect the performance of CauseRec. Case studies further demonstrate that CauseRec can help learn accurate user representations. To summarize, this paper makes the following key contributions:

- We propose to model the counterfactual data distribution (besides the observational data distribution) to confront the data sparsity problem for recommendation.
- We devise the CauseRec framework which learns accurate and robust user representations with counterfactual transformations on both fine-grained item-level and abstract interest-level, and with various contrastive objectives.
- We conduct extensive experiments and show that with a naive deep candidate generation architecture as the baseline, CauseRec can outperform SOTA sequential recommenders.

2 RELATED WORKS

2.1 Sequential Recommendation

Sequential recommendation can be traced back to leveraging Markov-chain [14, 15] and factorization machines [22, 49]. To capture long-term and multi-level cascading dependencies, deep learning based techniques (e.g., RNNs [10, 21, 47, 66] and CNNs [55, 75]) are incorporated into sequential modeling. DNNs are known to have enticing representation capability and have the natural strength to capture comprehensive relations [76] over different entities (e.g., items, users, interactions). Recently, there are works that explore advanced techniques, e.g., memory networks [53], attention mechanisms [56, 79], and graph neural networks [9, 26, 31, 36, 81] for sequential recommendation [6, 23, 29, 54, 61, 67, 72]. Typically, MIND [32] adopts the dynamic routing mechanism to aggregate users’ behaviors into multiple interest vectors. ComiRec [5] differs from MIND by leveraging the attention mechanism for user representations and proposes a factor for the trade-off between recommendation diversity and accuracy. Different from the above works that solely model the observational user sequences, we take a step further to model the counterfactual data distributions. By contrasting the user representations of the observation with the counterfactual, we aim to learn user encoders that can better confront out-of-distribution user sequences and learn accurate and robust user representations.

2.2 Contrastive Learning for Recommendation

A growing number of attempts have been made to exploit the complementary power of self-supervised learning (e.g., contrastive learning) and deep learning, with domains varying from computer vision [12, 17, 68], natural language generation [7, 77], to graph embedding [46]. However, how to consolidate the merits of contrastive learning into recommendation remains largely unexplored in the literature. Recently, Sun et al.[33] adopt noise contrastive estimation [16] to transfer the knowledge from a large natural language corpus to recommendation-specific content that is sparse on long-tail publishers and thus learning effective word representations. CLRec [83] bridges the theoretical gap between contrastive learning objective and traditional recommendation objective, e.g., sampled softmax loss, as well as more advanced inverse propensity weighted (IPW) loss. They show that directly performing contrastive learning can help to reduce exposure bias. CP4Rec [69] and S3-Rec [84]...
integrates Bert structure and contrastive learning objective for user pretraining, which require a fine-tuning stage. Compared with these works, we design model-agnostic and non-intrusive frameworks that help any baseline model learn more effective user representations in an end-to-end manner. Such representations are more accurate and robust by contrasting the original user representation with counterfactually positive samples and counterfactually negatives samples.

2.3 Counterfactual for Recommendation

Causality and counterfactual reasoning have attracted great attentions in various domains [11, 78, 80]. Previous counterfactual frameworks in recommendation focus on debiasing the learning-to-rank problems. A rigorous counterfactual learning framework, i.e., PropDCG [1], is proposed to overcome the distorting effect of presentation bias. The position bias and the clickbait issue are investigated in [2, 64] and [62], respectively. The Inverse Propensity Score [52, 70] method obtains unbiased estimation by sample re-weighting based on the likelihood of being logged. Another line of works encapsulates the uniform data into recommendation by learning imputation models [73], computing propensity [52], using knowledge distillation [13, 37], and directly modeling the uniform data [4, 28, 38, 50]. Different from these works, we focus on denoising user representation learning and considers the retrospsect question, i.e., ‘what the user representation would be if we intervene on the observed behavior sequence?’. Technically, we propose several counterfactual transformations based on the identification of indispensable/disposable concepts and devise several contrasting objectives for learning accurate and robust user representations.

3 METHODS

3.1 Problem Formulation

In the view of sequential recommendation, datasets can be formulated as \( D = \{ (x_{u,t}, y_{u,t}) \}_{u=1,2,...,N,t=1,2,...,T_u} \), where \( x_{u,t} = \{ y_{u,1:t-1} \} \) denotes a user’s historical behaviors prior to the \( t \)-th behavior \( y_{u,t} \) and arranged in a chronological order, and \( T_u \) denotes the number of behaviors for the user \( u \). The goal of sequential recommendation is to predict the next item \( y_{u,t} \) given the historical behaviors \( x_{u,t} \), which can be formulated as modeling the probability of all possible items:

\[
p(y_{u,t} = y|x_{u,t}) ,
\]

We will drop the sub-scripts occasionally and write \((x, y)\) in place of \((x_{u,t}, y_{u,t})\) for simplicity. Let \( X \) denote the set of all possible click sequences, i.e. \( x \in X \) and each \( y \in Y \) represent a clicked item, while \( Y \) is the set of all possible items.

Since the number of items \(|Y|\) can easily reach 100 million, industrial recommender systems consist typically of two phases, i.e., the matching phase and the ranking phase, due to concerns on system latency. The matching (also called deep candidate generation) phase focuses on retrieving Top N candidates for each user, while the ranking phase further sorts the N candidates by typically considering more fine-grained user/item features and incorporating complex modeling architectures. In this paper, we mainly conduct experiments in the matching stage (e.g., comparing with SOTA matching models).

3.2 A Naive Matching Baseline

The paradigm of a matching model includes a user encoder \( f_\theta(x) \in \mathbb{R}^d \), which takes the user’s historical behavior sequence as input and output one (or more) dense vector representing the user’s interests, and an item encoder \( g_\phi(y) \in \mathbb{R}^d \) that represents the items in the same vector space as the user encoder. We denote all the trainable parameters in the system as \( \Theta \), which includes the parameters in \( f_\theta \) and \( g_\phi \). With the learned encoders and the extracted item vectors, i.e., \( \{g_\phi(y)\}_{y \in Y} \), a k-nearest-neighbor search service, e.g., Faiss [27], will be deployed for Top-N recommendation. Specifically, at serving time, an arbitrary user behavior sequence \( x \) will be transformed into a vectorial representation \( f_\theta(x) \) and top \( N \) items with the largest matching scores will be retrieved as Top-N candidates. Such matching scores are typically computed as inner product \( \phi_\theta(x, y) = \langle f_\theta(x), g_\phi(y) \rangle \) or cosine similarity. In a nutshell, the learning procedure can be formulated as the following maximum likelihood estimation:

\[
\arg\min_\theta \frac{1}{|D|} \sum_{(x,y) \in D} - \log p_\theta(y | x) ,
\]

where \( p_\theta(y | x) = \frac{\exp \phi_\theta(x, y)}{\sum_{y' \in Y} \exp \phi_\theta(x, y')} \).

In the matching phase, it can be infeasible to sum over all possible items \( y' \) as in the denominator. Here we adopt the sample softmax objective [5, 25]. To demonstrate the effectiveness of the proposed CauseRec architecture, we utilize a naive framework as the baseline. Specifically, the item encoder \( g_\phi(y) \) is a plain lookup embedding matrix where the \( n \)-th vector represents the item embedding with item id \( n \). The user encoder \( f_\theta(x) \) aggregates the embeddings of historically interacted items using global average pooling and then transforms the aggregated embedding into the same embedding space as item embeddings using multi-layer perceptrons (MLP):

\[
f_\theta(x) = \text{MLP}(\frac{1}{t-1} \sum_{i=1}^{t-1} g_\phi(y_i)) .
\]

3.3 The CauseRec Architecture

In this section, we give an brief illustration on the intuition and overall schema/pipeline of the CauseRec architecture, which is depicted in Figure 2, and introduce the building blocks in detail.

3.3.1 Overall Schema. The essence of CauseRec is to answer the retrospsect question, ‘what the user representation would be if we intervene on the observed behavior sequence?’ The counterfactual transformation in CauseRec relates to the ‘intervention on the observed behavior sequence’. For answering ‘what the user representation would be,’ we introduce an important inductive bias that makes the intervention work as expected. Specifically, we first identify indispensable/disposable concepts in the historical behavior sequence. An indispensable concept indicates a subset of one behavior sequence that can jointly represent a meaningful aspect of the user’s interest. A dispensable concept indicates a noisy subset that is less meaningful/important in representing an aspect of interest. We introduce the details in Section 3.3.2.
Given the identified concepts, a representative counterfactual transformation is designed to build out-of-distribution counterfactual user sequences. Here comes the inductive bias, i.e., counterfactual sequences constructed by replacing the dispensable concepts in the original user sequence should still have similar semantics to the original one. Here semantics refer to the characteristics of user interests/preferences. Therefore, replacing indispensable concepts in the original user sequence should incur a preference deviation from the resulted user representation to the original user representation. We denote these resulted user representations as counterfactually negative user representations. We note that such negatives are hard negatives where hard refers to that other dispensable concepts stay the same as the original user sequence and negatives means that the semantics of the user sequence should be different. In contrast, replacing dispensable concepts in the original user sequence should incur no preference change in representations. We denote these resulted user representations as counterfactually positive user representations. Different contrastive learning objectives are proposed to learn accurate and robust user representations that are less sensitive to the (attack on the) dispensable/noisy concepts and that trust more on the indispensable concepts that better represent the user’s interest. Details can be found in Section 3.3.5.

3.3.2 Identification of Indispensable/Dispensable concepts. To identify indispensable/ dispensable concepts, we propose to first extract concept proposals and compute the proposal scores.

Item-level Concepts. Inspired by instance discrimination [17], a straightforward while workable solution is to treat each item in the behavior sequence as an individual concept since each item has its unique fine-grained characteristics. In this way, we obtain the concept sequence $C = X \in \mathbb{R}^{T \times d}$, where $X = \phi(x_{t+1})$ denotes the vectorial representations of the behavior sequence. In essence, concept scores indicate to what extent these concepts are important to represent the user’s interest. Since there is no groundtruth for one user’s real interest, we use the target item $y_{t+1}$ as the indicator:

$$p^{\text{item}}_i = \phi(y_{i}),$$

where $c_i$ indicates the representation of $i$th concept in $C$, and $y$ indicates the representation of the target item. $\phi$ is the similarity function, and we empirically use dot product for its effectiveness in the experiment. $p^{\text{item}}_i$ is thus the score for the $i$th concept.

Interest-level Concepts. However, such a solution may incur redundancy in concepts since some items may share similar semantics, and might deteriorate the capability of modeling higher-order relationships between items. To this end, besides the item-level concepts, we introduce interest-level concepts by leveraging the attention mechanism [56] to extract interest-level concepts. Formally, $X \in \mathbb{R}^{T \times d}$, we obtain the following attention matrix:

$$A = \text{softmax} \left( W_2 \tanh \left( W_1 X^T \right) \right)^T,$$

where $W_1 \in \mathbb{R}^{d \times d}$ and $W_2 \in \mathbb{R}^{K \times d}$ are trainable transformation matrices. $K$ is thus the number of concepts that is pre-defined. $A$ is of shape $\mathbb{R}^{T \times K}$. We obtain the concept sequence as the following:

$$C = A^T X.$$

Since interest-level concepts and the target item are not naturally embedded in the same space, we compute the concept score by the weighted sum of item-level scores:

$$p^{\text{interest}} = A^T \phi(y, X, y).$$

For both item-level concepts and interest-level concepts, we treat the top half concepts with the highest scores as indispensable concepts and the remaining half concepts as dispensable concepts. This strategy is mainly designed to prevent the number of indispensable or dispensable concepts from being too small. We leave finding more effective solutions as future works as illustrated in Section 5.

3.3.3 Counterfactual Transformation. The proposed counterfactual transformation aims to construct out-of-distribution user sequences by transforming the original user sequence for one user. We here
use **user sequence** to generally denote the concept sequence, which can be either the commonly known item-level concept sequence, *i.e.*, the original user behavior sequence, or the interest-level concept sequence. Based on the inductive bias described in Section 3.3.1, we propose to replace the identified indispensable/disposable concepts at the rate of $r_{rep}$ to construct counterfactually negative/positive user sequences, respectively. We note that directly dropping indispensable/disposable concepts also seems feasible, but replacement has the advantage of not affecting overall sequence length and the relative positions of remaining concepts. We maintain a first-in-first-out queue as a concept memory for each level and use dequeued concepts as substitutes. We enqueue the concepts extracted from the current mini-batch. We denote the user sequence with indispensable/disposable concepts being replaced as counterfactually negative/positive user sequence.

### 3.3.4 User Encoders

We note that item-level concept representations can be trained along with the item embedding matrix in most recommendation frameworks. Therefore, the above item-level concept identification and counterfactual transformation processes can be performed without any modification on the user encoder in the original baseline model, *i.e.*, a model-agnostic and non-intrusive design. We denote the architecture solely considering item-level concepts as CauseRec-Item. CauseRec-Item obtains counterfactually positive/negative user representations $\{x^{i,m}_i\}_{m=1}^M / \{x^{-i,n}_i\}_{n=1}^N$ from counterfactually item-level concept sequences using the original user encoder $f_0$.

We denote the architecture solely considering interest-level concepts as CauseRec-Interest. Different from CauseRec-Item, interest-level concepts are constructed with learnable parameters, *i.e.*, $W_1$ and $W_2$ in Equation 6. Therefore, CauseRec-Interest is an intrusive design, and the inputs to the user encoder should be the interest-level concept sequence rather than the behavior sequence at the item-level. We note that there are no further modifications, and the architecture of the user encoder can stay the same as in the original baseline model. CauseRec-Interest obtains counterfactually positive/negative user representations from counterfactually interest-level concept sequence using the original user encoder $f_0$.

We denote the architecture that considers counterfactual transformation on both the item-level concept sequence and the interest-level concept sequence as CauseRec-H(ierarchical). CauseRec-H is also an intrusive design with interest-level concepts as the inputs of the user encoder. Different from CauseRec-Interest, CauseRec-H further considers counterfactual transformations performed on item-level concepts. The counterfactually transformed item-level sequence will be forwarded to construct interest-level concept sequence using Equation 6-7. We note that counterfactual transformations will not be performed on these two levels simultaneously, which might introduce unnecessary noises. In other words, each counterfactual user representation is constructed with transformation on sequence solely from one level.

#### 3.3.5 Learning Objectives

Besides the original recommendation loss $L_{matching}$ described near Equation 2, we propose several contrastive learning objectives that are especially designed for learning accurate and robust user representations.

**Contrast between Counterfactual and Observation.** As discussed in Section 3.3.1, a **robust** user representation should be less sensitive to (possible attack on) dispensable concepts. Therefore, the user representations learned from counterfactual sequences with indispensable concepts transformed should be intuitively pushed away from the original user representation. Similar in spirit, an **accurate** representation should trust more on indispensable concepts. Therefore, user representations learned from counterfactual sequences with dispensable concepts transformed should be intuitively pulled closer to the original user representation. Under these intuitions, we derive inspiration from the recent success of contrastive learning in CV [17, 68] and NLP [7], we use triplet margin loss to measure the relative similarity between samples:

$$L_{co} = \sum_{m=1}^M \sum_{n=1}^N \max\left\{d(x^i, x^{i,m}) - d(x^i, x^{-i,n}) + \Delta_{co}, 0\right\},$$

where $x^i$ denotes the original user representation. We set the distance function $d$ as the L2 distance since user representations generated by the same user encoder are in the same embedding space. We empirically set the margin $\Delta_{co} = 1$.

**Contrast between Interest and Items.** The above objective considers the user representation side solely, and we further capitalize on the target item $y_i$, which also enhances the user representation learning. Formally, given the L2-normalized representation of the target item $\tilde{y}$ and user representation $\tilde{x}$, we have:

$$L_{ii} = \sum_{m=1}^M \sum_{n=1}^N \max\left\{0, \tilde{x}^{i,m} \cdot \tilde{y} + \Delta_{ii}\right\},$$

This objective also has the advantage of preventing the user encoder from learning trivial representations for counterfactual user sequences. We set the margin $\Delta_{ii} = 0.5$ in the experiment. Finally, the loss for training the whole framework can be written as:

$$L_{cause} = L_{matching} + \lambda_1 L_{co} + \lambda_2 L_{ii}.$$  

During testing/serving, only the backbone model that generates the user representation is needed. The identification of indispensable/disposable concepts and the counterfactual transformation processes are disregarded. Noteworthy, the computation of proposal scores which depends on the target item does not belong to the backbone model and is not required during testing.

### 4 EXPERIMENTS

We conduct experiments on real-world public datasets and mainly aim to answer the following three research questions:

- **RQ1**: How does CauseRec perform compared to the base model and various SOTA sequential recommenders?
- **RQ2**: How do the proposed building blocks and different hyperparameter settings affect CauseRec?
- **RQ3**: How do user representations benefit from modeling the counterfactual world and contrastive representation learning?

#### 4.1 Experimental Setup

To demonstrate the generalization capability on learning users’ representations of the proposed CauseRec architecture, we employ an
evaluation framework [5, 35, 41] where models should confront unseen user behavior sequences. Specifically, the users of each dataset are split into training/validation/test subset by the proportion of 8 : 1 : 1. For training sequential recommenders, we incorporate a commonly used setting by viewing each item in the behavior sequence as a potential target item and using behaviors that happen before the target item to generate the user’s representation, as defined in Section 3.1. For evaluation, only users in the validation/test set are considered, and we choose to generate users’ representations on the first 80% behaviors, which are unseen during training. Such a framework can help justify whether models can learn accurate and robust user representations that can generalize well. We mainly focus on the matching phase of recommendation and accordingly choose the datasets, comparison methods, and evaluation metrics.

Datasets We consider three challenging recommendation datasets, of which the statistics are shown in Table 1.

| Dataset        | #Users | #Items | #Interactions | #Density |
|----------------|--------|--------|---------------|----------|
| Amazon Books   | 459, 133 | 313, 966 | 8, 898, 041  | 0.00063  |
| Yelp           | 31, 668  | 38, 048  | 1, 561, 406   | 0.00130  |
| Gowalla        | 52, 643  | 91, 599  | 2, 984, 108   | 0.00084  |

Comparison Methods We mainly consider sequential recommenders for comparison since models are required to confront unseen behaviors for each user. Therefore, factorization-based and graph-based methods are not considered. The compared state-of-the-art models are listed as the following:

- **POP.** A naive baseline that always recommends items with the most number of interactions.
- **YouTube DNN** [8]. A successful industrial recommender that generates one user’s representation by pooling the embeddings of historically interacted items.
- **GRU4Rec** [21]. An early attempt to introduce recurrent neural networks into recommendation.
- **MIND** [32]. The first framework that extracts multiple interest vectors for one user based on the capsule network.
- **ComiRec-DR** [5]. A recently proposed SOTA framework following MIND to extract diverse interests using dynamic routing and incorporate a controllable aggregation module to balance recommendation diversity and accuracy.
- **ComiRec-SA** [5]. ComiRec-SA differs from ComiRec-DR by using self-attention to model interests.

Evaluation Metrics We employ three broadly used numerical criteria for the matching phase, i.e., Recall, Normalized Discounted Cumulative Gain (NDCG), and Hit Rate. We report metrics computed on the top 20/50 recommended candidates. Higher values indicate better performance for all metrics.

Implementation Details We use Adam [30] for optimization with learning rate of 0.003/0.005 for Books/Yelp and Gowalla, $\beta_1 = 0.9$, $\beta_2 = 0.99$, $\epsilon = 1 \times 10^{-8}$, weight decay of $1 \times 10^{-5}$. We train CauseRec-Item for (maximum) 10 epochs and CauseRec-Interest/CauseRec-H for (maximum) 30 epochs with mini-batch size 1024. All models are with embedding size 64. We set hyper-parameters $\lambda_1 = \lambda_2 = 1$ and do not tune them with bells and whistles. As illustrated in Section 3.2, the item encoder is a plain embedding lookup matrix, and the user encoder is a three-layer perceptron with hidden size 256. We set $N = 8$, $M = 1$, $t_{rep} = 0.5$ for CauseRec-Item/-Interest and $N = 16$, $M = 2$ for CauseRec-H to accommodate transformation on two levels, as illustrated in Section 3.3.4. We set $K = 20$ for CauseRec-Interest/-H.

### 4.2 Performance Analysis (RQ1)

The comparison results of CauseRec with SOTA sequential recommenders are listed in Table 2. We report three architectures of CauseRec including CauseRec-Item (CauseItem), CauseRec-Interest (CauseIn), and CauseRec-Hierarchical (CauseH), as described in Section 3.3.4. In a nutshell, we observe a clear improvement of these architectures over various comparison methods and across three different metrics. Notably, CauseRec-H improves the previous SOTA ComiRec-SA/DR by +0.0299 (relatively 22.1%) concerning NDCG@50 on the Amazon Books dataset and +0.0179 (relatively 8.64%) concerning Recall@20 on the Gowalla dataset. Among the comparison methods, ComiRec mostly yields the best performance by modeling multiple interests for a given user. However, only modeling the noisy historical behaviors might result in diverse but noisy interests that may not accurately represent users, finally leading to inferior results. GRU4Rec achieves comparably good results with ComiRec on the Gowalla dataset. GRU4Rec can effectively model the sequential dependency between items in the behavior sequence. However, it might be more likely to suffer from the noises due to the strict step-by-step encoding process. In contrast, CauseRec architectures confront the noises within users’ behaviors by pushing the user representation away from counterfactually negative user representations and pulling it closer to counterfactually positive user representations. Besides, these results demonstrate the generalization capability of CauseRec on confronting out-of-distribution user sequences by modeling the counterfactual world.

Among three CauseRec architectures, CauseRec-Item is a model-agnostic and non-intrusive design, which means it can be applied to any other sequential recommender without any modification on the original user encoder, and solely functions in the training stage without sacrificing inference efficiency. CauseRec-Interest constructs interest-level concepts by grouping items that may belong to a certain interest (e.g., chocolate and cake belong to sweets) into one holistic concept. Compared to CauseRec-Item, CauseRec-Interest has the advantage of reducing concept redundancy and modeling higher-order relationships between items, and thus improving CauseRec-Item. To combine the merits of CauseRec-Interest and CauseRec-Item, CauseRec-Hierarchical considers both
interest-level and item-level concepts in counterfactual transformation. CauseRec-H achieves the best results, which shows that counterfactual transformation on item-level concepts still yields some unique advantages, such as modeling fine-grained preferences. For example, people might not generally like all sweets and prefer cake to chocolate.

4.2.1 Ablation Studies. We are interested in the CauseRec-Item architecture due to its strengths of being: easy to implement (model-agnostic), efficient in serving (non-intrusive), and effective. To obtain a better understanding of different building blocks in CauseRec-Item, we consider surgically removing some components and construct the following architectures. The results on Yelp and Gowalla datasets are shown in Table 3.

A naive matching baseline described in Section 3.2. Base Model

| Datasets | Metric | POP | Y-DNN | GRU4Rec | MIND | ComiSA | ComiDR | CauseItem | CauseIn | CauseH | Improv. |
|----------|--------|-----|-------|---------|------|-------|-------|-----------|---------|--------|--------|
| Books    | R@20   | 0.0137 | 0.0457 | 0.0406 | 0.0486 | 0.0549 | 0.0531 | 0.0582 | 0.0593 | 0.0623 \* | 13.3% |
|          | R@50   | 0.0240 | 0.0731 | 0.0650 | 0.0764 | 0.0847 | 0.0811 | 0.1001 | 0.0993 | 0.1018 \* | 20.2% |
|          | NDCG@20 | 0.0226 | 0.0767 | 0.0680 | 0.0793 | 0.0899 | 0.0918 | 0.0985 | 0.1006 | 0.1051 \* | 14.5% |
|          | NDCG@50 | 0.0394 | 0.1208 | 0.1037 | 0.1223 | 0.1356 | 0.1352 | 0.1628 | 0.1619 | 0.1655 \* | 22.1% |
|          | HR@20  | 0.0302 | 0.1029 | 0.0894 | 0.1062 | 0.1140 | 0.1201 | 0.1280 | 0.1303 | 0.1370 \* | 14.1% |
|          | HR@50  | 0.0523 | 0.1589 | 0.1370 | 0.1610 | 0.1720 | 0.1758 | 0.2078 | 0.2062 | 0.2113 \* | 20.2% |
| Yelp     | R@20   | 0.0016 | 0.0506 | 0.0454 | 0.044 | 0.0534 | 0.0472 | 0.0570 | 0.0580 | 0.0591 \* | 10.7% |
|          | R@50   | 0.003 | 0.1048 | 0.0937 | 0.0943 | 0.1101 | 0.0935 | 0.1163 | 0.1175 | 0.1182 \* | 7.36% |
|          | NDCG@20 | 0.0065 | 0.1582 | 0.1447 | 0.1414 | 0.1728 | 0.1453 | 0.1812 | 0.1806 | 0.1830 \* | 5.90% |
|          | NDCG@50 | 0.0129 | 0.2887 | 0.2673 | 0.2699 | 0.3025 | 0.2612 | 0.3179 | 0.3180 | 0.3210 \* | 6.12% |
|          | HR@20  | 0.0152 | 0.3015 | 0.2826 | 0.2681 | 0.3249 | 0.2775 | 0.3416 | 0.3391 | 0.3426 \* | 5.45% |
|          | HR@50  | 0.0268 | 0.5131 | 0.4853 | 0.4866 | 0.5324 | 0.4629 | 0.5583 | 0.5576 | 0.5605 \* | 5.28% |
| Gowalla  | R@20   | 0.0028 | 0.1127 | 0.1273 | 0.1218 | 0.1277 | 0.1153 | 0.1315 | 0.1355 | 0.1359 \* | 6.42% |
|          | R@50   | 0.0054 | 0.1926 | 0.2043 | 0.2049 | 0.2072 | 0.1831 | 0.2238 | 0.2204 | 0.2251 \* | 8.64% |
|          | NDCG@20 | 0.0073 | 0.2378 | 0.2803 | 0.2565 | 0.2736 | 0.2534 | 0.2747 | 0.2825 | 0.2842 \* | 1.39% |
|          | NDCG@50 | 0.0135 | 0.3638 | 0.4002 | 0.3888 | 0.4019 | 0.3621 | 0.4123 | 0.4113 | 0.4221 \* | 5.03% |
|          | HR@20  | 0.0104 | 0.3443 | 0.3814 | 0.3627 | 0.3838 | 0.3429 | 0.3918 | 0.3995 | 0.4042 \* | 5.32% |
|          | HR@50  | 0.0224 | 0.5010 | 0.5251 | 0.5301 | 0.5288 | 0.5355 | 0.5596 | 0.5553 | 0.5697 \* | 6.39% |

Table 2: Comparison results of three CauseRec architectures with SOTA sequential recommenders designed for the matching phase. CauseItem/CauseIn/CauseH stand for CauseRec-Item/-Interest/-Hierarchical, respectively. The symbol \* indicates the improvements over the strongest baseline (underlined) are statistically significant (p < 0.05) with one-sample t-tests.
We attribute this phenomenon to that small 𝑟.

We ablate the number of counterfactually positive/negative user representations, i.e., 𝑀 and 𝑁, as defined in Section 3.3.4. As shown in Table 4, we can observe that 1) increasing the number (from 𝑁 = 𝑀 = 1 to 𝑁 = 𝑀 = 8) not necessarily leads to better performance. 2) particularly increasing the number of counterfactually negative user representations (from 𝑁 = 𝑀 = 1 to 𝑁 = 8, 𝑀 = 1) can be useful. This result is reasonable in the sense that such representations can be interpreted as hard negatives since each of the corresponding counterfactual user sequences contains dispensable items staying the same as the original user sequence, and hard negatives are known to be helpful. 3) particularly increasing the number of counterfactually positive user representations (from 𝑁 = 𝑀 = 1 to 𝑁 = 1, 𝑀 = 8) may introduce noises since each of them contains some randomly sampled items that can be falsely interpreted as “positive”. Therefore one counterfactually positive user representation can be enough for learning accurate user embeddings.

4.2.2 Analysis on the number of counterfactual user representations. We ablate the number of counterfactually positive/negative user representations, i.e., 𝑀 and 𝑁, as defined in Section 3.3.4. As shown in Table 4, we can observe that 1) increasing the number (from 𝑁 = 𝑀 = 1 to 𝑁 = 𝑀 = 8) not necessarily leads to better performance. 2) particularly increasing the number of counterfactually negative user representations (from 𝑁 = 𝑀 = 1 to 𝑁 = 8, 𝑀 = 1) can be useful. This result is reasonable in the sense that such representations can be interpreted as hard negatives since each of the corresponding counterfactual user sequences contains dispensable items staying the same as the original user sequence, and hard negatives are known to be helpful. 3) particularly increasing the number of counterfactually positive user representations (from 𝑁 = 𝑀 = 1 to 𝑁 = 1, 𝑀 = 8) may introduce noises since each of them contains some randomly sampled items that can be falsely interpreted as “positive”. Therefore one counterfactually positive user representation can be enough for learning accurate user embeddings.

4.2.3 Analysis on the Replace Ratio. We are interested in how the replacement ratio, i.e., 𝑟𝑒𝑝, in counterfactual transformation of CauseRec-Item affects the model performance. Table 5 shows the results by varying 𝑟𝑒𝑝. The best result is achieved with 𝑟𝑒𝑝 = 0.4/0.5 for the Yelp dataset and 𝑟𝑒𝑝 = 0.5 for the Gowalla dataset. Either too small or too large 𝑟𝑒𝑝 will lead to sub-optimal results. We attribute this phenomenon to that small 𝑟𝑒𝑝 will affect the capability of counterfactual learning and large 𝑟𝑒𝑝 will introduce more noises brought by randomly sampled items for replacement. 𝑟𝑒𝑝 makes a tradeoff between these two impacts.

4.2.4 Analysis on the number of interest-level concepts. Here we take an analysis on the number of interest-level concepts particularly for CauseRec-Interest architecture. Specifically, we ablate 𝑘 as described in Equation 6. As shown in Table 6, we observe a performance improvement with 𝑘 increasing (e.g., 4 → 10, 10 → 20). Each interest-level concept can be more coarse-grained when 𝑘 becomes smaller and more fine-grained when 𝑘 becomes larger. It can be hard to classify a coarse-grained concept as indispensable or dispensable. It may lead to also coarse-grained or even inaccurate transformations afterward, and thus hurting the performance. Too large 𝑘 (e.g., 30) may bring redundancies and noises (more random concepts introduced with a fixed replace ratio) to the framework and eventually leads to inferior results.

4.3 Case Study (RQ3)

To understand how the learned user representations benefit from the CauseRec framework, we plot randomly selected six users from the Amazon books dataset. We also plot ten corresponding items sampled from the test set for each user. Specifically, we perform
Figure 3: Visualization of randomly sampled users (shown as stars) with their interacted items (shown as points of the same color) from the Amazon Books dataset. We perform the t-SNE transformation on the representations learned by the base model (left) and CauseRec (right).

t-SNE transformation on the user/item representations learned by the base model (as shown in Figure 3a) and CauseRec-Item (as shown in Figure 3b). The connectivities of users and test items in the embedding space can help reflect whether the model learns accurate and robust user representations. From Figure 3b, we observe that users with their corresponding test items easily form clusters and show small intra-cluster distances and large inter-cluster distances. By jointly comparing the same users (e.g., 590695, and 586100) in Figures 3a and 3b, we can see that CauseRec-Item helps the user encoder learn representations that are closer to their corresponding test items. These results qualitatively demonstrate the effectiveness of CauseRec on learning accurate and robust user representations.

We also present a recommendation result from the Amazon Books test datasets in Figure 4. We list the historical behaviors, the top five books recommended by the base model and CauseRec-Item, and books interacted by the corresponding user in the test set. We mainly visualize the books’ covers and categories for better clarity. We note that the side information is generally not considered in training matching models (both the base model and CauseRec). As shown in Figure 4, we observe that CauseRec yields more consistent recommendation results to the books in the test set. Supposing historical behaviors consist of noisy ones, and behaviors in the test accurately reflect users’ interest for the current state, CauseRec successfully captures users’ interests, i.e., Children’s Books, and Literature&Fictions. In contrast, the base model is more likely to be affected by noisy behaviors that appear only a few times, such as the Biographies&Memories, and Education&Reference. These results further demonstrate that CauseRec can learn accurate and robust user representations that are less distracted by noisy behaviors.

5 CONCLUSION AND FUTURE WORK

In this work, we propose to model the counterfactual data distribution to confront the sparsity and noise nature of observed user interactions in recommender systems. The proposed CauseRec conditionally sample counterfactually positive and negative user sequences with transformations on the dispensable/indispensable concepts. We propose multiple structures (-item, -interest, -hierarchical) to confront both fine-grained item-level concepts and abstract interest-level concepts. Several contrastive objectives are devised to contrast the counterfactual with the observational to learn accurate and robust user representations. Among several proposed architectures, CauseRec-Item has the advantage of being non-intrusive, i.e., solely functioning at training while not affecting serving efficiency. With a naive matching baseline, CauseRec achieves a considerable improvement over it and SOTA sequential matching recommenders. Extensive experiments help to justify the strengths of CauseRec as being both simple in design and effective in performance.

This work can be viewed as an initiative to exploit the joint power of constitutive learning and counterfactual thinking for recommendation. We believe that such a simple and effective idea can be inspirational to future developments, especially in model-agnostic and non-intrusive designs. CauseRec-Item is compatible with various user encoders within most existing sequential recommenders. We choose a naive baseline to better demonstrate the effectiveness of this work, and we plan to explore its strengths in more models. Another future direction is whether more effective solutions of identifying indispensable/disposable concepts exist, including both the computation of concept scores and the determination of indispensable or disposable for each concept based on the scores. Lastly, we will explore the strengths of CauseRec for the ranking phase of recommendation. Counterfactual transformations designed with various auxiliary features and complex model architectures will open up new research possibilities.

6 ACKNOWLEDGMENTS

The work is supported by the National Key R&D Program of China (No. 2020YFC0832500), NSFC (61625107, 61836002, 62072397), Zhejiang Natural Science Foundation (LR19F020006), and Fundamental Research Funds for the Central Universities (2020QNA5024).
