CoDeR: Counterfactual Demand Reasoning for Sequential Recommendation

Shuai Tang*, Sitao Lin*, Jianghong Ma†, Xiaofeng Zhang†

Harbin Institute of Technology(Shenzhen), Shenzhen, China
tangshuai@stu.hit.edu.cn, 220810134@stu.hit.edu.cn, majianghong@hit.edu.cn, zhangxiaofeng@hit.edu.cn

Abstract

Sequential recommendation systems aim to predict the next item based on users' historical interactions. While traditional methods focus on learning feature representations or user preferences, they often struggle with detecting subtle demand shifts in short sequences, especially when these shifts are obscured by noise or biases. To address these issues, we propose CoDeR (Counterfactual Demand Reasoning), a novel framework designed to handle demand shifts in sequential recommendations with greater precision. CoDeR features two key modules: (1) the User Demand Extraction module, which utilizes self-attention mechanisms and demand graphs to identify and model demand shifts from minimal user interactions; and (2) the Counterfactual Demand Reasoning module, which employs causal effect analysis and backdoor adjustment techniques to distinguish true demand shifts from noisy or biased signals. Our approach represents the first application of counterfactual reasoning to sequential recommendation systems. Comprehensive experiments on three real-world datasets demonstrate that CoDeR significantly outperforms existing baselines.

Introduction

Sequential recommendation (SR) is a key component of modern E-commerce, aiming to predict the next item a user will engage with by analyzing historical sequential data. This approach captures user behavior patterns through dependencies and interactions in their browsing or purchasing history. Research typically focuses on either learning detailed feature representations from this data or extracting higher-level semantic insights, such as user preferences, to improve recommendation tasks. These tasks include next item recommendation (Lu, Wu, and Yuan 2023; Cho et al. 2023; Yue et al. 2024) and next basket recommendation (Li et al. 2023), among others. A brief review of existing SR approaches is presented below.

Prior Works. Previous research on preference (interest, intention or demand)-oriented approaches has focused on understanding user preferences from historical sequential data, such as browsing or purchase activities. These approaches typically learn latent preferences from interaction data, train item feature representations and align them with item representations for recommendations (Feng et al. 2019; Hidasi et al. 2015; Tang and Wang 2018). Among these methods, attention-based techniques (Li et al. 2017; Ying et al. 2018; Liu et al. 2018) excel at discerning preferences, while chain-based methods like FPMC (Rendle, Freudenthaler, and Schmidt-Thieme 2010) and GRU4Rec (Hidasi et al. 2016) address sequential item dependencies. To further capture these dependencies in longer sequential data, several graph neural network (GNN) models (Li et al. 2015; Veličković et al. 2017) capture complex item relationships over longer sequences (Qiu et al. 2019; Wu et al. 2019; Xu et al. 2019a; Zhang et al. 2020). Despite advancements of preference-oriented approaches (Zhou et al. 2020; Liu et al. 2021; Zhu et al. 2020), DAGNN (Yang et al. 2022) posits that demand is a higher-level concept built upon preferences. DAGNN introduces a demand estimation module that captures both global and session-specific demands, enhancing recommendation accuracy for both common and rare items.

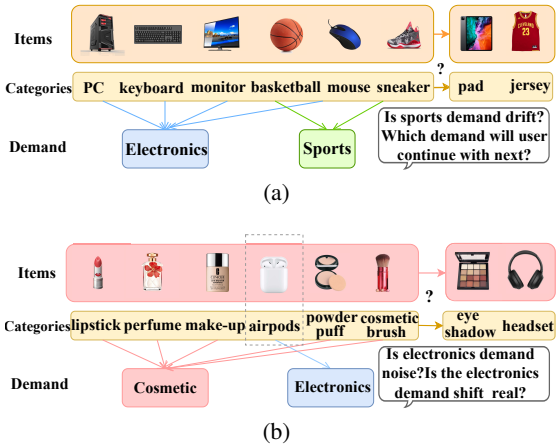

Figure 1: Motivation example.

The SOTA demand-driven approaches like DAGNN appear to overlook the fact that demand can vary even within a short period (i.e., a few clicks) and is sometimes hard to distinguish from random noise caused by accidental clicks or position bias. Specifically, there are two main challenges:

*These authors contributed equally.
†Corresponding authors.

First, users may change their demands after initial browsing, necessitating the recommendation of items under the new demand. Extracting new demand from just a few clicks is difficult, because they are shorter than typical sequential data. For instance, as illustrated in Fig.1(a), a user initially browses Electronics products, but two clicked items, i.e. a basketball and sneakers, indicate a demand shift from Electronics to Sports. ***Second***, these few clicks might include skewed actions such as clickbait (Wang et al. 2021b) and position bias (Jagerman, Oosterhuis, and de Rijke 2019), which may not accurately reflect the true underlying demands. Therefore, it is crucial to differentiate such noisy "demand" from the true varying demand. Fig.1(b) demonstrates a "noisy" demand shift where user continues to browse Cosmetic products despite an incidental click on "airpods". Thus, recommending the "headset" would be inappropriate as Electronics is not the true demand shift.

To address the aforementioned issues, we propose the **Co**unterfactual **De**mand **R**easoning graph neural network **(CoDeR)** model, which includes two main modules: **(1)** User Demand Extraction module, which identifies ***demand shifts using only a few clicked items***. This module initially leverages the self-attention mechanism to capture the interdependencies among items in user sequential data. Subsequently, a demand graph is constructed based on the demand scores of items to learn demand-aware item embeddings. **(2)** Counterfactual Demand Reasoning module, which ***differentiates "noisy" demand shifts from true demand shifts***, and to investigate the causal relationship between demand shifts and the recommended items. To achieve this, a causal relation graph is established, positing demand drift as a confounding factor that creates a backdoor path. By utilizing backdoor adjustment, we can obstruct the backdoor path and mitigate the influence of confounding factors. The key contributions are summarized as follows.

- This work represents the ***first*** attempt to explore user demand shifts in sequential recommendation through the application of a counterfactual reasoning model.
- We propose an innovative User Demand Extraction module to detect user demand shifts by encoding user sequences with item, category, and positional embeddings, utilizing self-attention mechanisms to capture dependencies, and constructing a demand graph to refine representations and accurately model demand dynamics.
- We introduce a novel Counterfactual Demand Reasoning module to differentiate true demand shifts from "noisy" demand shifts by leveraging causal effect analysis. This involves constructing a causal relation graph, where backdoor adjustment is utilized to block the backdoor path and eliminate the influence of confounding factors.
- Comprehensive experiments conducted on three real-world datasets demonstrate that the proposed approach outperforms existing baselines.

Related Work

Sequential Recommendation Methods

Sequential recommendation predicts users' next actions based on their historical behavior. The Markov Decision Process (MDP) (Shani et al. 2005) models item transition probabilities, while FPMC (Rendle, Freudenthaler, and Schmidt-Thieme 2010) combines Matrix Factorization (MF) and Markov Chains (MC) to capture sequential relationships. Deep learning advances have introduced methods that leverage complex dependencies in sequential data (Yuan et al. 2019; Zhou et al. 2018, 2019; Sun et al. 2019; Tang and Wang 2018), such as GRU-based models (Hidasi et al. 2016) for effective pattern modeling.

Addressing interest drift, (Tan, Xu, and Liu 2016) suggests data augmentation, and model like STAMP (Liu et al. 2018) uses attention mechanisms to discern and integrate user interests. Self-attention methods (Kang and McAuley 2018; Xu et al. 2019b; Ren et al. 2020) enhance performance with transformer architectures, while graph representation learning approaches (Gori, Monfardini, and Scarselli 2005; Li et al. 2015; Xu et al. 2019a; Wu et al. 2019; Wang et al. 2020b; Chen and Wong 2020) and models like DeepWalk (Perozzi, Al-Rfou, and Skiena 2014), LINE (Tang et al. 2015), and node2vec (Grover and Leskovec 2016) capture nonlinear item relationships and semantic connections.

Causal Recommendation Methods

Counterfactual reasoning has gained attention in various research fields (Feng et al. 2021; Chen et al. 2019). The causal reasoning framework (Pearl 2019) uses causal graphs to identify true causal relationships in data. Early work in causal recommendation focused on reducing biases in learning-to-rank problems, with methods like (Agarwal et al. 2019) employing gradient optimization and propensity-weighted ranking for unbiased results.

In sequential recommendations, causal reasoning helps address biases. For instance, (Wang et al. 2020a) uses causal graphs to improve click-through rate prediction by excluding exposure effects. (Gupta et al. 2021) and (Zheng et al. 2023) introduce frameworks and methods to reduce popularity bias. Techniques like the Inverse Propensity Score method (Schnabel et al. 2016; Yang et al. 2018) and counterfactual data augmentation (Chen et al. 2022; Peng, Song, and Liu 2023) further enhance unbiased estimation and address data sparsity.

Problem Formulation

Let $V = \{v_1, v_2, ..., v_{|s|}\}$ denote a sequence of $|s|$ items, and let the corresponding item category sequence be denoted as $C = \{c_1, c_2, \ldots, c_{|s|}\}$. Assume there are M demand spaces, and each session may encompass one or several underlying demands within this space. Let $\mathcal{V}$ and $\mathcal{C}$ denote the item set and its category set, respectively, while $\mathcal{S}$ represents the set of sequences. Given a sequence $s = \{(v_1, c_1), (v_2, c_2), ..., (v_{|s|}, c_{|s|}) | v_i \in \mathcal{V}, c_i \in \mathcal{C}, i \in \{1, ..., |s|\}\}$, the goal of SR problem is to predict the probability $P(v_t|s)$ that the next item v_t will be recommended, given the preceding s_{t-1}. This problem is formulated as follows:

$$\arg\max_{v_i \in \mathcal{V}} p(v_t = v_i \,|s_{t-1}), \tag{1}$$

where a candidate item with the highest probability is to be predicted as the next item.

The demand shift can be formally quantified by assessing the variation in demand within a sequence s over time. This is expressed as follows:

$$D_{drift} = {D^m}_{t+1} - {D^m}_t, \tag{2}$$

where D_t^m denotes the demand representation in the m-th demand space at time t. Note that D_{drift} contains both true shift and "noisy" shift which will be further differentiated in the next section.

To account for the confounding effects of demand drift on user demand and recommendations, the probability is estimated using backdoor adjustment, expressed as:

$$P(I|do(D)) = \sum_{D_{drift}} P(I|D, D_{drift})P(D_{drift}), \tag{3}$$

where $do(D)$ represents an intervention on demand.

By incorporating the causal effect of demand shift into the final recommendations, the overall model predicts the next item with the highest probability, determined as:

$$\arg\max_{v_i \in \mathcal{V}} \left[p(v_t = v_i|s_{t-1}, P(I|do(D))\right]. \tag{4}$$

The Proposed CoDeR Model

User Demand Extraction Module

This module aims to extract the underlying user demand from the input sequential data. It starts with a submodule that models demand within the sequence, which serves as the basis for the subsequent submodule. A demand graph is then created to refine item representations based on the demand insights provided by the initial submodule.

User Demand Modeling. We start by encoding user sequences $S = \{s_i|s_i = (v_i, c_i)\}$, where each element s_i consists of an item v_i and its associated category c_i. To represent these elements, we create two embedding matrices: the item embedding matrix $V = [v_1, v_2, ..., v_{|s|}] \in R^{n_v \times |s|}$ and the category embedding matrix $C = [c_1, c_2, ..., c_{|s|}] \in R^{n_v \times |s|}$, where $v_i \in \mathcal{V}$ and $c_i \in \mathcal{C}$ are denoted as n_v- dimensional embeddings.

We then integrate positional embeddings p_i into our model, where $p_i \in R^{n_p}$ is the n_p-dimensional positional embedding for the position index i. The final input representation for each sequence element x_i is computed as the sum of the item embedding, category embedding, and positional embedding, expressed as:

$$x_i = v_i + c_i + p_i, i \in \{1, 2, ..., |s|\}, \tag{5}$$

where $v_i \in R^{n_v \times |s|}$ denotes the item embedding for item v_i, $c_i \in R^{n_c \times |s|}$ denotes the category embedding and $p_i \in R^{n_p \times |s|}$ denotes the positional embedding.

To obtain a more precise representation of demand for each user sequence, we employ a self-attention mechanism (Vaswani et al. 2017) following the embedding layer. This mechanism is designed to capture the interdependencies among different items within the user sequence. Specifically, the self-attention mechanism operates on the feature representation X^L, and details are provided hereinafter.

Finally, we project the transformed sequence representation X^L into predefined demand spaces using the transformation $D^m = W_d^m X^L$ for $m \in \{1, \ldots, M\}$.

To represent multiple demands within a sequence, we aggregate the generated demand representations along the transformed sequence dimension. This aggregation yields a comprehensive demand representation vector for each demand space as:

$$d^m = \log \sum_{i=1}^{|s|} \exp(D^m[:, i]), \tag{6}$$

where the i-th column d_i^m of D^m is the demand representation of the i-th item in the m-th demand space.

The demand score is calculated as follows. For each element in the sequence s and a target item v_t, we derive the demand score vector $z^m = [z_1^m, z_2^m, \cdots, z_{|s|}^m] \in R^{|s|}$.

The final demand scores are then obtained by applying a dot product operation:

$$z_i^m = \sigma(\frac{< d^m, W^k x_i >}{\sqrt{n_d}}), \tag{7}$$

where $W^k \in R^{n_d \times n_c}$ is the learnable weight matrix, σ is sigmoid function to normalize the score, demand score z_i^m denotes the contribution of x_i to the m-th demand d^m.

Demand Graph Constructing. This submodule is crucial for identifying demand shifts within a sequence. We will explain the process of constructing the demand graph and how it updates both demand and item representations.

Given a session $s = \{s_i|s_i = (v_i, c_i)\}$, we construct a directed demand graph $G^m = (V, E^m)$ to capture the sequence order of items. In this graph, E^m represents the directed edges in the m-th demand space graph, with an edge $e_{v_i,v_j} = 1$ between items v_i and v_j if $j = i + 1$, thus preserving the sequence order. Each node $v_i \in V$ is assigned a weight corresponding to its demand score z_i^m, which is derived from the previous submodule.

To update node representations, we employ a message passing mechanism inspired by gate control mechanisms (Hochreiter and Schmidhuber 1997; Sutskever, Vinyals, and Le 2014) and graph attention networks (Veličković et al. 2017). In this approach, the demand score of each node functions as a gating unit, regulating the flow of information from neighboring nodes. For each node v_i in $G^m = (V, E^m)$, we define the aggregation of 1-hop neighboring nodes N_{v_i} in the l-th layer of our GNN as:

$${h_i}^{m,l} = agg(\{v_j^{m,l-1}, j \in N_{v_i}\}), \tag{8}$$

$$agg(\cdot) = \frac{1}{|N_{v_i}|} \sum_{j \in N_{v_i}} k_j^m \odot e_{v_i,v_j}^m v_j^{m,l-1}, \tag{9}$$

where N_{v_i} is the set of neighboring nodes of v_i, and k_j^m represents the gating value of node v_j in the demand graph G^m. Note the node representation $\boldsymbol{v}_i^m$ in G^m is updated as:

$$v_i^{m,l} = \sigma(W^{m,l} concat(v_i^{m,l-1}, h_i^{m,l}) + b_m^l), \tag{10}$$

where $W^{m,l} \in R^{n \times 2n}$, $b_m^l \in R^n$ are learnable parameters.

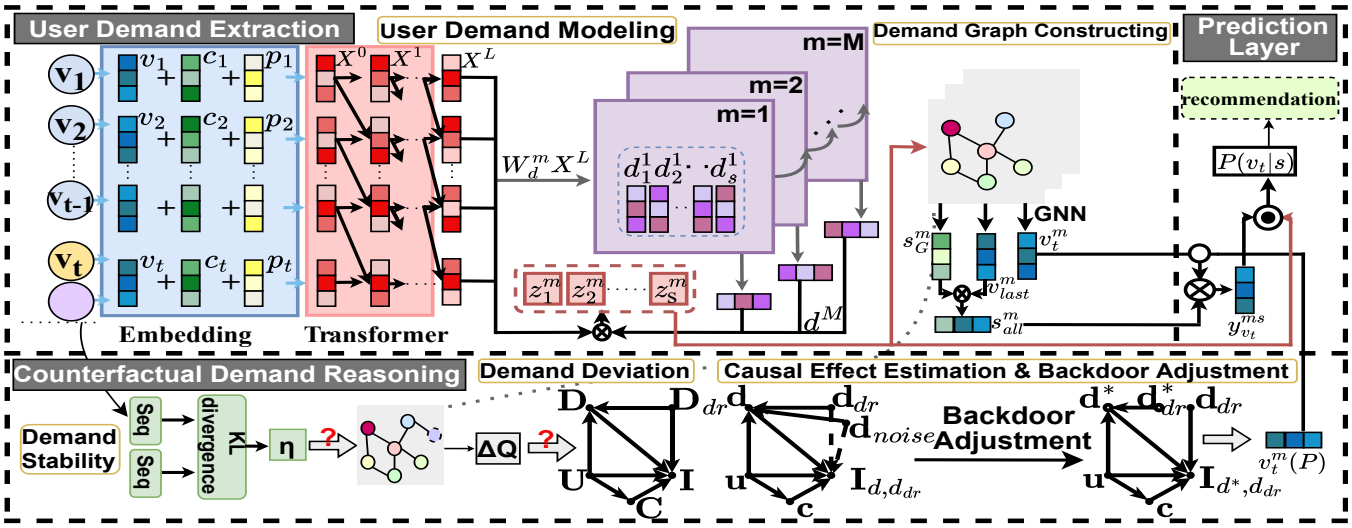

Figure 2: **Overall architecture of CoDeR.** It includes two modules: User Demand Extraction, which identifies demand shifts from limited interactions, and Counterfactual Demand Reasoning, which distinguishes true demand shifts from "noisy" ones.

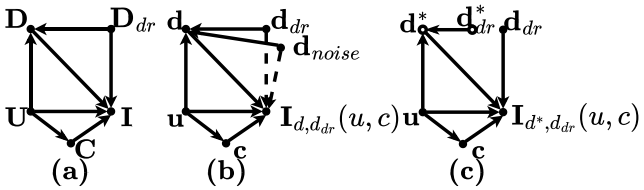

Figure 3: The built causal graph based on user demand for SR, where $*$ denotes the reference value.

Given that each item contributes differently to the recommendations, we design a soft-attention mechanism to place greater emphasis on recent items in the sequence. The attention scores and the resulting demand representations s_G for a sequence are computed as follows:

$$\beta_i = \alpha^T \sigma(W_1 v_i + W_2 v_{last} + b), \tag{11}$$

$$s_G = \sum_{i=1}^{|s|} \beta_i v_i. \tag{12}$$

Counterfactual Demand Reasoning Module

This module differentiates true demand shifts from "noisy" ones and explores the causal relationship between demand shifts and SR outcomes. To achieve this, we first construct a causal graph to model the impact of user demand and its shifts on item recommendations(Fig.3), using modularity gain of the demand graph to approximate demand shifts and KL divergence to quantify "noisy" shifts.Intervention techniques, specifically backdoor adjustment, is then applied to mitigate confounding effects. Details of each submodule is described below.

Causal Effect Estimation. Fig.3 illustrates the causal relation graph, where nodes represent variables and edges denote causal relationships. For instance, the edge from D to I suggests that D has a causal effect on I. The transition from Fig.3(a) to Fig.3(b) depicts the process of distinguishing "noisy" demand shifts from true demand shifts. Meanwhile, the changes from Fig.3(b) to Fig.3(c) represent the process of counterfactual reasoning. A detailed explanation of the causal relationship graph is provided as follows:

- D and U denote demands and users, respectively. C represents the user sequence, I is the recommendation item list, and D_{dr} represents user demand drift.
- $U \rightarrow D$: The demand is influenced by the user. For example, college students majoring in computer science have a demand for purchasing textbooks w.r.t. computer science.
- $U \rightarrow C$: Users' interests and personal characteristics (e.g., geography, occupation, etc.) determine the distribution of user sequences. For instance, users in tropical regions are less likely to interact with items related to down jackets.
- $U \rightarrow I$: The recommendation item list is highly correlated with the user's preferences, establishing a direct causal relationship between the user's preferences and the recommendation outcomes.
- $C \rightarrow I$: The distribution of user sequences reflects the user's interests and preferences, thereby influencing the final recommendation results. Additionally, as users interact with click sequences, their preferences may shift, leading to changes in the recommendation results.
- $D_{dr} \rightarrow D$: User demand drift impacts user demand.
- $(D, U, C, D_{dr}) \rightarrow I$: The recommendation list is affected by the demand, the user, sequence, and demand drift.

The causal effect quantifies the alteration in the response variable when the treatment variable transitions from its reference value to its expected value, referred to as the total effect (TE) (Pearl 2022; Wang et al. 2020a). Formally, the TE of D, D_{dr} on I under $U = u, C = c$ is defined as:

$$TE = I_{d,d_{dr}}(u,c) - I_{d^*,d^*_{dr}}(u,c), \tag{13}$$

where $U = u$, $C = c$ is unchanged, and $I_{d^*,d_{dr^*}}(u,c) =$

$I(U=u, C=c, D=d^*, D_{dr}=d^*_{dr})$ represents the reference situation with D, D_{dr} set to d^*, d^*_{dr}, respectively.

We then conduct counterfactual inference to reduce the direct effect of demand drift and mitigate "noisy" demand shifts from true demand shifts. According to the causality theory, the TE can be divided into the natural direct effect (NDE) and total indirect effect (TIE). In this task, TIE is treated as the true demand shift, whereas NDE can be approximated as the "noisy" demand shift. To this end, we calculate NDE and subtract it from TE to get TIE. As shown in Fig.3(c), the effect of D_{dr} on D is blocked by setting D to the reference situation d^*, calculated as:

$$NDE = I_{d^*,d_{dr}}(u,c) - I_{d^*,d^*_{dr}}(u,c). \tag{14}$$

Then, we can achieve the debiasing inference by TIE as:

$$TIE = TE - NDE = I_{d,d_{dr}}(u,c) - I_{d^*,d_{dr}}(u,c). \tag{15}$$

According to TIE, we can assess whether the user's demand shift affects the user's demand. Since the reference situation is unobservable, we design an estimation strategy with two terms: demand stability and demand deviation. If a user has stable demand and significant deviation occurs, we consider a demand shift.

Demand Stability. To calculate the stability property of a user's demand, we employ the symmetric KL divergence inspired by (Wang et al. 2021a). The historical interaction sequence is divided into two parts, and the distribution of the item set for each part is calculated as $d^n_u = [p^n_u(g_1), p^n_u(g_2), ..., p^n_u(g_N)]$, where $p_u(g_n)$ is defined as:

$$p_u(g_n) = \sum_{i\in\mathcal{V}} p(g_n\,|i\,)p(i\,|u\,) = \frac{\sum_{i\in H_u} q^i_{g_n}}{|H_u|}, \tag{16}$$

where $\mathcal{V}$ represents the set of all items, H_u is the set of items in the sequence for user u, and $q^i_{g_n}$ indicates whether item i belongs to class group g_n or not. We quantify the divergence between these two distributions using KL divergence as:

$$\eta = KL(d^1_u\,|d^2_u\,) + KL(d^2_u\,|d^1_u\,) \tag{17}$$

$$= \sum_{n=1}^{N} p^1_u(g_n)\log\tfrac{p^1_u(g_n)}{p^2_u(g_n)} + \sum_{n=1}^{N} p^2_u(g_n)\log\tfrac{p^2_u(g_n)}{p^1_u(g_n)}, \tag{18}$$

where d^1_u and d^2_u are the divided user's historical interaction sequence.

In summary, η indicates the likelihood of a user's interest shift across items, with a higher η value suggesting a greater chance of demand change.

Demand Deviation. To calculate demand deviation, we compute the modularity gain (Blondel et al. 2008) of the demand graph, which evaluates whether there is a significant deviation in user demand. The Louvain algorithm (Lu, Halappanavar, and Kalyanaraman 2015) is applied to compute the modularity. The modularity gain Q_G is determined as:

$$Q_G = \frac{1}{2\,|E^m|}\sum_{i,j}[A_{ij} - \frac{k_i k_j}{2\,|E^m|}]\delta(v_i, v_j), \tag{19}$$

where $|E^m|$ is the number of edges, A_{ij} is the adjacency matrix, and k_i represents the sum of weights of all edges connected to node v_i. $\delta(v_i, v_j)$ equals 1 if and only if v_i and v_j belong to the same community.

When a new candidate item is added to the demand graph Q, its impact on modularity is assessed to detect shifts in user demand. If the modularity falls below a threshold γ, it suggests that the item does not align with current demand. Conversely, if modularity exceeds γ, user demand remains stable. The modularity gain ΔQ is quantified as follows:

$$\Delta Q = [\frac{\sum in + k_{i,in}}{2\,|E^m|} - (\frac{\sum tot + k_i}{2\,|E^m|})^2] - [\frac{\sum in}{2\,|E^m|} - (\frac{\sum tot}{2\,|E^m|})^2 - (\frac{k_i}{2\,|E^m|})^2], \tag{20}$$

where $\sum in$ denotes the sum of weights of internal edges within community c, while $\sum tot$ is the sum of weights of all edges connected nodes in c, including both internal and external edges. In addition, k_i is the sum of weights of all edges connected to node v_i, and $k_{i,in}$ is the sum of weights of edges connecting node v_i to community c.

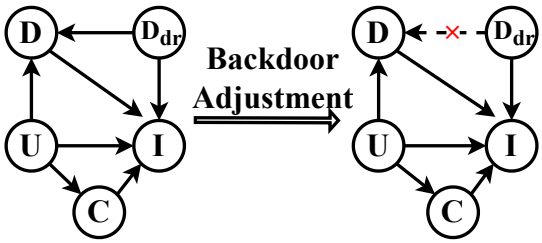

Figure 4: Backdoor adjustment to reduce the confounding effects on the causal relationship.

Applying Backdoor Adjustment for Confounding Bias. Based on the Structural Causal Model (SCM) (Pearl 2019), the causal graph in Fig.4 shows D_{dr} as a confounder, creating a backdoor path $I \leftarrow D_{dr} \rightarrow D$, introducing bias in the causal path from D_{dr} to D. To mitigate this, we apply backdoor adjustment (Pearl 2019) to block the path, reducing confounding effects. Existing models estimating conditional probabilities may suffer from correlation errors due to confounding bias, which we correct by intervening on D by employing backdoor adjustment, recalibrating the estimated probabilities. This precess is mathematically represented as:

$$P\,(I\,|do(D=d), C\,) = P_{G'}\,(I\,|D, D_{dr}, C\,) \tag{21a}$$

$$= \sum_u P_{G'}\,(I\,|D, D_{dr}, C, u\,)P_{G'}\,(u\,|D, D_{dr}, C\,) \tag{21b}$$

$$= \sum_u P_{G'}\,(I\,|D, D_{dr}, C, u\,)P_{G'}\,(u) \tag{21c}$$

$$= \sum_u P\,(I\,|D, D_{dr}, C, u\,)P\,(u\,), \tag{21d}$$

where G' denotes the causal graph after the backdoor adjustment, as shown in the right section of Fig.4. The derivation of Eq.21 is as follows:

- Eq.21b is derived from $Bayes' theorem$.

Dataset	#Item	#Category	#Training	#Testing
Diginetica	43,137	996	103,906	25,976
Tmall	17,095	698	28,004	7,001
Tafeng	14,637	1,638	80,552	20,138

Table 1: Statistics of experimental datasets.

- Eq.21a and Eq.21c result from the application of $do-calculus$ (Pearl 2019) on D. The backdoor path $I \leftarrow D_{dr} \rightarrow D$ is blocked by $do(D = d)$, thus mitigating the bias in user demand. Thus, $P_{G'}(u|D, D_{dr}, C)$ simplifies to $P_{G'}(u)$.
- Eq.21d follows from maintaining the same prior distribution on both graphs (G and G'). The relationship $(D, U, C, D_{dr}) \rightarrow I$ remains unchanged by removing the edge $D_{dr} \rightarrow D$.

To estimate $\sum_u P(I|D, D_{dr}, C, u)$ and $P(u)$, we adopt a dual approach. First, for estimating $\sum_u P(I|D, D_{dr}, C, u)$, we calculate the probability of recommending a candidate item list given a specific user demand, considering their demand, interaction history, and observed demand drift. Second, estimating $P(u)$ involves determining the probability distribution of users across different demand types.

By employing counterfactual inference with backdoor adjustment and intervention, the confounding bias from demand drift is effectively mitigated, minimizing its impact on recommendation outcomes.

Prediction

Prediction Layer. The prediction layer evaluates the probability of recommending a candidate item to a user. It first obtains sequence embeddings s_G and target item embeddings v_{last} from the user demand GNN submodule. These embeddings are then concatenated and transformed linearly to form the combined input s_{all} for the prediction layer:

$$s_{all} = W^h concat(s_G, v_{last}), \tag{22}$$

where $W^h \in R^{n \times 2n}$ is the learnable network parameters.

The matching degree $y^s_{v_t}$ between the candidate item v_t and the combined input s_{all} is then calculated as:

$$y^s_{v_t} = \langle s_{all}, v_t \rangle. \tag{23}$$

Finally, the probability $P(v_t|s)$ of recommending a candidate item v_t is given as:

$$p(v_t|s) = \text{softmax}(\sum_{m=1}^{M} z^m_t y^{m,s}_{v_t}), \tag{24}$$

where z^m_t denotes the demand score of item v_t, and $y^{m,s}_{v_t}$ represents the matching degree of the overall demand representation in user sequences in the m-th demand space.

Recommendation Strategy. We first assess whether a user has stable demand characteristics. If a stable-demand user shows signs of demand shift, we apply backdoor adjustment to mitigate confounding bias from demand drift. Specifically, the user's KL divergence value η is calculated using Eq.17. If η is below a threshold μ set to the mean KL divergence across users, the user is considered stable. Next, if the modularity gain calculated from Eq.20, falls below the threshold γ, after adding candidate items, this indicates a demand shift. For such users, backdoor adjustment (Eq.21) is applied to reduce the impact of noisy demand shifts and enhance recommendation performance.

Model Loss

To train the model, we utilize the cross-entropy loss function, which is formulated as follows:

$$\mathcal{L}_p = -\frac{1}{|S|} \sum_{s \in S} \sum_{t=1}^{|V|} y_t \log p(v_t|s) + \lambda_1 ||\Theta||_2, \tag{25}$$

where S denotes the set of training sequences and y_t is the ground truth for the target item. And L_2 regularization is applied to mitigate overfitting, where λ_1 is a hyper-parameters that controls the strength of the L_2 regularization, and Θ is the set of model parameters.

Experimental Results

Datasets

In our experiments, we adopt three widely recognized real-world datasets—Diginetica, Tmall, and TaFeng[1]—to assess the performance of the proposed model. The detailed statistics of datasets are provided in Table 1.

Baselines

We select conventional methods: **POP**, **item-KNN** (Sarwar et al. 2001), and **FPMC** (Rendle, Freudenthaler, and Schmidt-Thieme 2010); RNN-based methods: **GRU4REC** (Hidasi et al. 2016) and **NARM** (Li et al. 2017); **STAMP**(Liu et al. 2018); graph-based methods: **SR-GNN** (Wu et al. 2019) and **LightGCN** (He et al. 2020); and the SOTA SR methods: **CORE** (Hou et al. 2022) and **DiffuRec** (Li, Sun, and Li 2023).

Performance Comparison

We present the experimental results in Table 2, where the best outcomes are highlighted in bold and the second-best are underlined. Several observations are made as follows:

Superior Performance: Our model surpasses all baselines by discerning true shifts in user demand from noise, thus improving the accuracy of recommendations.

Variation in NDCG Metrics: The increase in NDCG metrics for the Diginetica and Tmall datasets is less pronounced compared to the Tafeng dataset due to: (1) *E-commerce Dataset Characteristics:* Diginetica and Tmall contain substantial promotional data, resulting in user sequences with items less relevant to actual demands, thus recommended items aligned with true preferences may be ranked lower. (2) *Stability in Grocery Store Data:* The Tafeng dataset, from a grocery store, shows more stable user

[1]https://github.com/RUCAIBox/RecSysDatasets

Dataset	Metric	POP	item-KNN	FPMC	GRU4Rec	NARM	STAMP	SR-GNN	LightGCN	CORE	DiffuRec	Ours
Diginetica	R@10	0.923	24.192	18.922	19.190	34.154	32.454	36.744	25.023	41.860	35.546	**42.965**
	R@20	1.519	34.921	32.069	31.600	47.072	44.514	49.085	48.366	52.894	45.825	**54.180**
	R@40	2.498	46.060	48.135	46.945	60.211	57.465	61.598	60.655	63.318	55.419	**64.896**
	N@10	0.430	12.719	8.539	9.273	18.853	18.328	21.041	20.001	25.695	21.112	**26.346**
	N@20	0.579	15.427	11.839	12.395	22.112	21.368	24.156	24.126	28.642	23.712	**29.207**
	N@40	0.778	17.715	15.122	15.527	24.803	24.020	26.725	25.561	30.537	25.681	**31.365**
Tmall	R@10	0.070	4.542	20.507	7.858	20.000	21.532	22.804	19.482	21.738	22.180	**23.091**
	R@20	0.461	12.363	23.592	9.901	23.211	24.771	26.234	22.412	25.007	26.755	**27.269**
	R@40	2.698	13.598	26.747	12.680	26.662	28.084	30.742	25.409	28.351	31.331	**31.828**
	N@10	0.036	4.300	14.765	4.966	14.297	15.061	15.888	14.175	15.208	15.761	**16.213**
	N@20	0.135	6.464	15.540	5.482	15.108	15.851	16.722	14.919	16.007	16.923	**17.263**
	N@40	0.585	6.683	16.190	6.021	15.816	16.513	17.420	15.542	16.675	17.859	**18.195**
TaFeng	R@10	5.498	2.784	6.630	6.686	7.129	7.186	7.960	6.204	6.609	4.364	**8.201**
	R@20	6.710	3.889	9.337	9.811	10.040	10.154	11.080	10.095	10.596	5.957	**11.548**
	R@40	10.532	5.456	12.943	13.997	13.917	13.983	15.794	15.047	15.627	7.994	**16.405**
	N@10	3.251	1.384	3.826	3.682	4.027	4.132	4.499	3.998	3.209	2.623	**4.655**
	N@20	3.565	1.664	4.451	4.428	4.685	4.879	5.226	4.398	4.210	3.029	**5.525**
	N@40	4.338	1.984	5.263	5.238	5.540	5.658	6.108	4.984	5.235	3.440	**6.514**

Table 2: Model performance evaluation results on Diginetica, Tmall, and Tafeng datasets.

demands, allowing our model to effectively detect and correct demand biases. (3) *Impact of Dataset Time Span:* The short time span of these datasets likely contributes to this phenomenon, as user preferences, reducing the likelihood of preference shifts.

Efficacy of GNN-based Methods: The SR-GNN method excels among baselines, highlighting the potential of GNNs to capture complex item relationships in sequences. It indicates that, in addition to temporal ordering and long-term preferences, intricate relationships and transition patterns reflecting user demands are effectively learned.

Sequential Transition Patterns in Tmall Dataset: The FPMC method outperforms RNN-based methods in the Tmall dataset, indicating that sequential transition patterns between items are critical for effective SR.

Ablation Study

Dataset	Methods	R@20	R@40	N@20	N@40
Diginetica	w/o UDM	51.364	61.275	27.548	29.480
	w/o DGC	49.364	59.687	26.363	28.650
	w/o CDR	52.365	60.351	27.369	30.388
	CoDeR	**54.180**	**64.896**	**29.207**	**31.365**
Tmall	w/o UDM	26.307	30.830	16.265	17.600
	w/o DGC	24.651	28.266	15.632	16.316
	w/o CDR	26.537	30.365	16.354	17.365
	CoDeR	**27.269**	**31.828**	**17.263**	**18.195**
Tafeng	w/o UDM	10.955	15.263	4.686	5.501
	w/o DGC	10.383	12.920	4.596	5.374
	w/o CDR	11.336	15.939	5.139	6.099
	CoDeR	**11.548**	**16.405**	**5.525**	**6.514**

Table 3: Ablation study results on different datasets.

To validate the effectiveness of our proposed modules, we conducted ablation studies by removing the following components: the User Demand Modeling (UDM) submodule, the Demand Graph Constructing (DGC) submodule, and the Counterfactual Demand Reasoning (CDR) module. The results of these experiments are summarized in Table 3.

Removal of UDM: Excluding the UDM submodule results in a significant drop in model performance, highlighting its critical role in capturing user interests and demand patterns, as well as in generalizing item categories in relation to user demands.

Removal of DGC: Excluding the DGC submodule significantly reduces performance in Recall and NDCG, highlighting the demand graph's importance in accurately learning and representing user demands at the item level.

Removal of CDR: For the TaFeng dataset, omitting the CDR module has a minimal impact on performance. This is likely due to the stable nature of user demands in grocery store transactions, which results in less frequent demand drift. Consequently, the CDR module's absence has a limited effect on the model's performance.

Conclusion

This study introduces CoDeR, a novel framework for enhancing sequential recommendation systems by identifying and differentiating user demand shifts. Traditional methods often struggle with subtle and noisy demand changes, but CoDeR addresses this by combining two key modules: the User Demand Extraction module, which uses self-attention and demand graphs to identify shifts from limited interactions, and the Counterfactual Demand Reasoning module, which employs causal analysis to distinguish true demand changes from noise. Experiments on three datasets show that CoDeR outperforms existing methods.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (Project No. 62202122 and No. 62073272), the Shenzhen Science and Technology Program under Grant No. GXWD20231130110308001 and No. JCYJ20240813104837050, and the Guangdong Basic and Applied Basic Research Foundation under Grant No. 2024A1515011949.

References

Agarwal, A.; Takatsu, K.; Zaitsev, I.; and Joachims, T. 2019. A general framework for counterfactual learning-to-rank. In *SIGIR*, 5–14.

Blondel, V. D.; Guillaume, J.-L.; Lambiotte, R.; and Lefebvre, E. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10): P10008.

Chen, L.; Zhang, H.; Xiao, J.; He, X.; Pu, S.; and Chang, S.-F. 2019. Counterfactual critic multi-agent training for scene graph generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 4613–4623.

Chen, T.; and Wong, R. C.-W. 2020. Handling Information Loss of Graph Neural Networks for Session-based Recommendation. In *SIGKDD*, 1172–1180.

Chen, X.; Wang, Z.; Xu, H.; Zhang, J.; Zhang, Y.; Zhao, W. X.; and Wen, J.-R. 2022. Data Augmented Sequential Recommendation based on Counterfactual Thinking. *TKDE*, 35(9).

Cho, J.; Hyun, D.; Lim, D. w.; Cheon, H. j.; Park, H.-i.; and Yu, H. 2023. Dynamic multi-behavior sequence modeling for next item recommendation. In *AAAI*, 4199–4207.

Feng, F.; Huang, W.; He, X.; Xin, X.; Wang, Q.; and Chua, T.-S. 2021. Should graph convolution trust neighbors? a simple causal inference method. In *SIGIR*, 1208–1218.

Feng, Y.; Lv, F.; Shen, W.; Wang, M.; Sun, F.; Zhu, Y.; and Yang, K. 2019. Deep session interest network for click-through rate prediction. *arXiv preprint arXiv:1905.06482*.

Gori, M.; Monfardini, G.; and Scarselli, F. 2005. A new model for learning in graph domains. In *IJCNN*, volume 2, 729–734.

Grover, A.; and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *SIGKDD*, 855–864.

Gupta, P.; Sharma, A.; Malhotra, P.; Vig, L.; and Shroff, G. 2021. Causer: Causal session-based recommendations for handling popularity bias. In *CIKM*, 3048–3052.

He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; and Wang, M. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *SIGIR*, 639–648.

Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; and Tikk, D. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*.

Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; and Tikk, D. 2016. Session-based recommendations with recurrent neural networks. In *ICLR*, 208–216.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*, 9(8): 1735–1780.

Hou, Y.; Hu, B.; Zhang, Z.; and Zhao, W. X. 2022. Core: simple and effective session-based recommendation within consistent representation space. In *SIGIR*, 1796–1801.

Jagerman, R.; Oosterhuis, H.; and de Rijke, M. 2019. To model or to intervene: A comparison of counterfactual and online learning to rank from user interactions. In *SIGIR*, 15–24.

Kang, W.-C.; and McAuley, J. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, 197–206. IEEE.

Li, J.; Ren, P.; Chen, Z.; Ren, Z.; Lian, T.; and Ma, J. 2017. Neural attentive session-based recommendation. In *CIKM*, 1419–1428.

Li, R.; Zhang, L.; Liu, G.; and Wu, J. 2023. Next Basket Recommendation with Intent-aware Hypergraph Adversarial Network. In *SIGIR*, 1303–1312.

Li, Y.; Tarlow, D.; Brockschmidt, M.; and Zemel, R. 2015. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*.

Li, Z.; Sun, A.; and Li, C. 2023. Diffurec: A diffusion model for sequential recommendation. *ACM Transactions on Information Systems*, 42(3): 1–28.

Liu, C.; Li, X.; Cai, G.; Dong, Z.; Zhu, H.; and Shang, L. 2021. Noninvasive Self-attention for Side Information Fusion in Sequential Recommendation. In *AAAI*, volume 35, 4249–4256.

Liu, Q.; Zeng, Y.; Mokhosi, R.; and Zhang, H. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In *SIGKDD*, 1831–1839.

Lu, H.; Halappanavar, M.; and Kalyanaraman, A. 2015. Parallel heuristics for scalable community detection. *Parallel Computing*, 47: 19–37.

Lu, X.; Wu, J.; and Yuan, J. 2023. Optimizing Reciprocal Rank with Bayesian Average for improved Next Item Recommendation. In *SIGIR*, 2236–2240.

Pearl, J. 2019. Causal inference in statistics: An overview. *Statistics surveys*, 3: 96–146.

Pearl, J. 2022. Direct and indirect effects. In *Probabilistic and causal inference: the works of Judea Pearl*, 373–392.

Peng, X.; Song, J.; and Liu, W. 2023. CALRec: Counterfactual Active Learning for Sequential Recommendation. In *ICAIBD*, 186–191.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *SIGKDD*, 701–710.

Qiu, R.; Li, J.; Huang, Z.; and Yin, H. 2019. Rethinking the item order in session-based recommendation with graph neural networks. In *CIKM*, 579–588.

Ren, R.; Liu, Z.; Li, Y.; Zhao, W. X.; Wang, H.; Ding, B.; and Wen, J.-R. 2020. Sequential recommendation with self-attentive multi-adversarial network. In *SIGIR*, 89–98.

Rendle, S.; Freudenthaler, C.; and Schmidt-Thieme, L. 2010. Factorizing personalized markov chains for next-basket recommendation. In *WWW*, 811–820.

Sarwar, B.; Karypis, G.; Konstan, J.; and Riedl, J. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW*, 285–295.

Schnabel, T.; Swaminathan, A.; Singh, A.; Chandak, N.; and Joachims, T. 2016. Recommendations as Treatments: Debiasing Learning and Evaluation. In *ICML*, 1670–1679.

Shani, G.; Heckerman, D.; Brafman, R. I.; and Boutilier, C. 2005. An MDP-based recommender system. *Journal of Machine Learning Research*, 6(9).

Sun, F.; Liu, J.; Wu, J.; Pei, C.; Lin, X.; Ou, W.; and Jiang, P. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *CIKM*, 1441–1450.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*.

Tan, Y. K.; Xu, X.; and Liu, Y. 2016. Improved recurrent neural networks for session-based recommendations. In *RecSys*, 17–22.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. Line: Large-scale information network embedding. In *WWW*, 1067–1077.

Tang, J.; and Wang, K. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *WSDM*, 565–573.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Wang, W.; Feng, F.; He, X.; Wang, X.; and Chua, T.-S. 2021a. Deconfounded recommendation for alleviating bias amplification. In *KDD*, 1717–1725.

Wang, W.; Feng, F.; He, X.; Zhang, H.; and Chua, T.-S. 2020a. "Click" Is Not Equal to "Like": Counterfactual Recommendation for Mitigating Clickbait Issue. *CORR*.

Wang, W.; Feng, F.; He, X.; Zhang, H.; and Chua, T.-S. 2021b. Clicks can be cheating: Counterfactual recommendation for mitigating clickbait issue. In *SIGIR*, 1288–1297.

Wang, Z.; Wei, W.; Cong, G.; Li, X.-L.; Mao, X.-L.; and Qiu, M. 2020b. Global context enhanced graph neural networks for session-based recommendation. In *SIGIR*, 169–178.

Wu, S.; Tang, Y.; Zhu, Y.; Wang, L.; Xie, X.; and Tan, T. 2019. Session-based recommendation with graph neural networks. In *AAAI*, volume 33, 346–353.

Xu, C.; Zhao, P.; Liu, Y.; Sheng, V. S.; Xu, J.; Zhuang, F.; Fang, J.; and Zhou, X. 2019a. Graph Contextualized Self-Attention Network for Session-based Recommendation. In *IJCAI*, volume 19, 3940–3946.

Xu, C.; Zhao, P.; Liu, Y.; Sheng, V. S.; Xu, J.; Zhuang, F.; Fang, J.; and Zhou, X. 2019b. Graph contextualized self-attention network for session-based recommendation. In *IJCAI*, volume 19, 3940–3946.

Yang, L.; Cui, Y.; Xuan, Y.; Wang, C.; Belongie, S.; and Estrin, D. 2018. Unbiased offline recommender evaluation for missing-not-at-random implicit feedback. In *RecSys*, 279–287.

Yang, L.; Luo, L.; Zhang, X.; Li, F.; Zhang, X.; Jiang, Z.; and Tang, S. 2022. Why do semantically unrelated categories appear in the same session? a demand-aware method. In *SIGIR*, 2065–2069.

Ying, H.; Zhuang, F.; Zhang, F.; Liu, Y.; Xu, G.; Xie, X.; Xiong, H.; and Wu, J. 2018. Sequential recommender system based on hierarchical attention network. In *IJCAI*, 3926–3932.

Yuan, F.; Karatzoglou, A.; Arapakis, I.; Jose, J. M.; and He, X. 2019. A simple convolutional generative network for next item recommendation. In *WSDM*, 582–590.

Yue, Z.; Wang, Y.; He, Z.; Zeng, H.; Mcauley, J.; and Wang, D. 2024. Linear Recurrent Units for Sequential Recommendation. In *WSDM*, 930–938.

Zhang, M.; Wu, S.; Gao, M.; Jiang, X.; Xu, K.; and Wang, L. 2020. Personalized graph neural networks with attention mechanism for session-aware recommendation. *TKDE*.

Zheng, Y.; Qin, J.; Wei, P.; Chen, Z.; and Lin, L. 2023. CIPL: Counterfactual Interactive Policy Learning to Eliminate Popularity Bias for Online Recommendation. *IEEE Transactions on Neural Networks and Learning Systems*.

Zhou, G.; Mou, N.; Fan, Y.; Pi, Q.; Bian, W.; Zhou, C.; Zhu, X.; and Gai, K. 2019. Deep interest evolution network for click-through rate prediction. In *AAAI*, volume 33, 5941–5948.

Zhou, G.; Zhu, X.; Song, C.; Fan, Y.; Zhu, H.; Ma, X.; Yan, Y.; Jin, J.; Li, H.; and Gai, K. 2018. Deep interest network for click-through rate prediction. In *SIGKDD*, 1059–1068.

Zhou, K.; Wang, H.; Zhao, W. X.; Zhu, Y.; Wang, S.; Zhang, F.; Wang, Z.; and Wen, J.-R. 2020. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *CIKM*, 1893–1902.

Zhu, N.; Cao, J.; Liu, Y.; Yang, Y.; Ying, H.; and Xiong, H. 2020. Sequential Modeling of Hierarchical User Intention and Preference for Next-item Recommendation. In *WSDM*, 807–815.