MuSeNet: Multi-Scenario Learning for Repeat-Aware Personalized Recommendation

Senrong Xu
State Key Laboratory for Novel Software Technology, Nanjing University
China
srxu@smail.nju.edu.cn

Liangyue Li
Alibaba Group
China
liliangyue.lly@alibaba-inc.com

Yuan Yao
State Key Laboratory for Novel Software Technology, Nanjing University
China
y.yao@nju.edu.cn

Zulong Chen
Han Wu
Quan Lu
Alibaba Group
China
{zulong.czl,zhaiyue.wh,luquan.lq}@alibaba-inc.com

Hanghang Tong
University of Illinois Urbana-Champaign
USA
htong@illinois.edu

ABSTRACT

Personalized recommendation has been instrumental in many real applications. Despite the great progress, the underlying *multi-scenario* characteristics (e.g., users may behave differently under different scenarios) are largely ignored by existing recommender systems. Intuitively, modeling different scenarios properly could significantly improve the recommendation accuracy, and some existing work has explored this direction. However, these work assumes the scenarios are explicitly given, and thus becomes less effective when such information is unavailable. To complicate things further, proper scenario modeling from data is challenging and the recommendation models may easily overfit to some scenarios. In this paper, we propose a multi-scenario learning framework, MuSeNet, for personalized recommendation. The key idea of MuSeNet is to learn multiple implicit scenarios from the user behaviors, with a careful design inspired by the causal interpretation of recommender systems to avoid the overfitting issue. Additionally, since users' repeat consumptions account for a large part of the user behavior data on many e-commerce platforms, a repeat-aware mechanism is integrated to handle users' repurchase intentions within each scenario. Comprehensive experimental results on both industrial and public datasets demonstrate the effectiveness of the proposed approach compared with the state-of-the-art methods.

CCS CONCEPTS

• **Information systems → Recommender systems**.

WSDM '23, February 27–March 3, 2023, Singapore, Singapore.

ACM ISBN 978-1-4503-9407-9/23/02…$15.00
https://doi.org/10.1145/3539597.3570414

KEYWORDS

Recommender system; Scenario learning; Causal interpretation; Repeat intention

ACM Reference Format:
Senrong Xu, Liangyue Li, Yuan Yao, Zulong Chen, Han Wu, Quan Lu, and Hanghang Tong. 2023. MuSeNet: Multi-Scenario Learning for Repeat-Aware Personalized Recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM '23), February 27–March 3, 2023, Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3539597.3570414

1 INTRODUCTION

Personalized recommendation has been increasingly instrumental for connecting users with items that match users' interests in various real-world applications [15, 26, 32, 37, 41]. Despite the progress, the underlying *multi-scenario* characteristics of users' *behaviors* (e.g., browse, click, purchase, etc.) have been largely ignored by the vast majority of existing recommender systems. An illustrative example of the multi-scenario characteristics of an online travel platform (OTP) is shown in Fig. 1. We assume that there are two different scenarios (i.e., business travel and leisure travel) for hotel reservation, which can be summarized from the users' hotel booking data. Intuitively, users may have different preferences in different scenarios. For example, company employees on business would choose business hotels near downtown for convenience while tourists may prefer enjoyable vacation hotels (e.g., seaside hotels as shown in Fig. 1). Knowing the background scenarios and tailoring the corresponding recommendation models properly could improve the recommendation accuracy, and some recent work [17, 19, 25, 28] has started initial explorations in this direction. However, existing work assumes that the scenarios are explicitly given (e.g., different recommendation scenes in homepage, add-to-cart page, etc.), and thus becomes less effective when such information is unavailable or unclear. Although we can manually define the scenarios, a sub-optimal division might even hinder the downstream recommendation tasks. Therefore, a natural solution is to jointly learn the implicit scenarios from data along with the recommendation task.

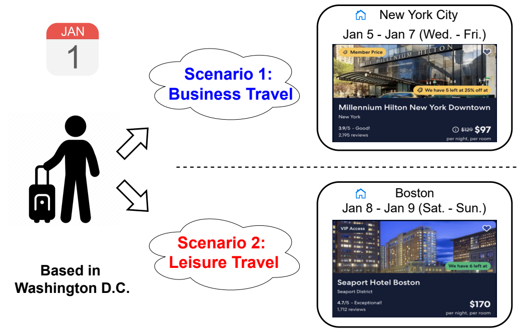


Figure 1: An illustrative example of multiple scenarios of an online travel platform. Users may have different preferences under different scenarios (e.g., business travel vs. leisure travel).

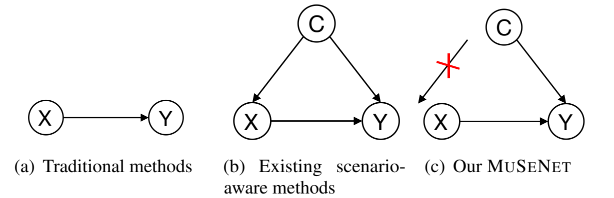


Figure 2: Causal diagrams for recommendation methods. (a) Traditional methods that directly predict interaction Y depending on data sample X. (b) Existing methods that construct scenarios C (e.g., based on pre-defined rules) as the common cause for both X and Y. (c) Our method that learns the scenarios from data and cuts off the confounding effect caused by $C \to X \to Y$.

However, learning meaningful scenarios from data for recommendation is non-trivial, mainly due to the fact that neural networks are prone to capturing unreliable correlations [8, 23, 38]. Such phenomenon becomes even worse in the multi-scenario learning case. To understand this, we may view each scenario as a training sample, and hence recommendation models may overfit to scenarios since there are usually a limited number of scenarios and some scenarios may be highly weighted (i.e., having many training samples). Consider the following example related to Fig. 1. The model may easily learn the correlation between A: "the user is a company employee" and B: "the user tends to book a business hotel", from the business travel scenario. However, it is also possible for a company employee to take a leisure trip, making the above correlation untenable in other scenarios. In this work, we call such issue as *scenario overfitting*, indicating that recommendation models may overfit to some scenarios with unreliable correlations, and thus give wrong predictions on the other scenarios.

In this paper, we propose a new recommendation framework, MuSeNet, which is able to learn multiple implicit scenarios from the existing user behaviors. Specifically, MuSeNet consists of two key components, i.e., the *scenario learning module* and the *repeat intention modeling module*. The scenario learning module uncovers the latent scenarios of the data in a suitable metric subspace. To handle the scenario overfitting problem, we first explain it from a causal view, and observe that the scenario exerts a confounding effect on the prediction from the data samples to labels for a straightforward solution. We then propose to mitigate the confounding effect of scenarios by introducing the *do*-calculus [10], which encourages the recommendation model to learn relatively generic *scenario-agnostic* correlations in addition to the *scenario-specific* correlations. To further show the benefit of scenario learning, we integrate the repeat intention modeling module into MuSeNet. This module infers user's intention to repeat a consumed item before or to explore a novel item, which is motivated by the observation that users' repeat consumptions account for a large part of the user behavior data on many e-commerce platforms [34]. Finally, we perform extensive experiments to evaluate the effectiveness of MuSeNet. The quantitative results show that the proposed method outperforms the competitors by up to 7.33% on average in terms of AUC, and the qualitative analysis indicates that the learned scenarios exhibit well-formed cluster structure and bear nice interpretations.

The main contributions of this paper are summarized as follows:

- To the best of our knowledge, this is the first work to mine and model the implicit scenarios in an industry-level recommender system.
- We propose a recommendation framework that is able to jointly learn multiple latent scenarios and alleviate the confounding effect from unobserved scenarios.
- We integrate user's scenario-specific repeat intentions into our model under a probabilistic framework.
- Comprehensive experiments on both industrial and public datasets demonstrate the effectiveness of our proposed model.

2 MOTIVATION

2.1 Causal Interpretation of Recommendation

We adopt the structural causal model (SCM) [10], which is a directed acyclic graph, to represent the data generation process of a recommender system. Specifically, we draw the causal diagrams for typical recommendation methods in Fig. 2, where each causal diagram contains the causal relations among different variables in the system. For simplicity, we consider the following three variables during the data generation process of a recommender system:

- X represents the data samples (i.e., user-item pairs) in the recommender system, and each data sample typically includes user profile, user historical behaviors, item features, etc.;
- Y represents the interaction label for each data sample, indicating whether the user has clicked or consumed the item;
- C represents the unobserved scenario or environment that affects the user behaviors in the recommender system.

Fig. 2(a) shows the causal diagram of the traditional recommendation methods that directly model $P(Y|X)$ without considering the *scenario* variable. Since users may have different preferences in different scenarios, such traditional methods cannot enjoy the performance gain provided by a proper scenario modeling, which is also indicated by the existing work [17, 19, 28].

To include the scenario, existing work [17, 28] proposes to model the scenario C as a cause variable for both X and Y. That is, both data samples and the user-item interactions are generated given the current scenario. The corresponding causal diagram is shown in Fig. 2(b). However, these methods still have limitations.

- First, existing recommendation methods assume that C is manually given (e.g., some static division rules). However, such static division rules may be unavailable in practice. Even when they are available, it is usually undesired to simply apply the rules since the given C may be sub-optimal and thus even degenerate the performance of recommendation models.
- Second, the confounder C of X and Y may mislead the recommendation model to mainly rely on scenario-specific correlations between X and Y, and thus decrease the generalization ability. Specifically, the causal path $C \rightarrow X \rightarrow Y$ makes data samples rely on scenarios, and thus recommendation models tend to learn unreliable correlations that only hold in particular scenarios. On the contrary, our expectation is to uncover some scenario-agnostic correlations between data samples and labels in addition to scenario-specific correlations.

To address the above limitations, we propose a new recommendation framework whose causal diagram is shown in Fig. 2(c). First, instead of dividing scenarios relying on the given knowledge provided by humans, we propose to learn the scenarios from data[1]. Such scenario learning shares the same goal with the recommender system to improve its overall performance. Second, we propose to eliminate the causal relationship $C \rightarrow X$ as it may lead to degenerated generalization performance. Consequently, only the causal path $C \rightarrow Y$ exists for scenario-specific information, pushing the $X \rightarrow Y$ path to learn some scenario-agnostic information.

2.2 Problem Definition

For user $u \in \mathcal{U}$, where $\mathcal{U}$ is the set of all users, we can observe u's past *behaviors* with the items from the item pool $\mathcal{T}$. For each behavior, we have the user's basic profile information p_u (e.g., user id, gender, age, etc.), the item attributes b_t (e.g., item id, category id, etc.) for item t, and the context information o_u (e.g., location, time, etc.) when the recommendation happens. We can also extract the user's past behavior sequence $\mathcal{B}_u = \{b_1^u, b_2^u, \ldots\}$ ordered by time, where b_i^u indicates the attributes of the i-th item interacted by user u. We define a *data sample* x as the tuple of $[p_u, \mathcal{B}_u, b_t, o_u]$ for each user behavior. Given the existing data samples, the recommendation task is to predict the preference of a user towards a target item.

3 THE PROPOSED MUSENET

3.1 MuSeNet Framework

Based on Fig. 2(c), a reasonable solution is to guide the model to reveal both scenario-specific and scenario-agnostic correlations behind data, which contribute to stable predictions across different scenarios. As a result, we need to intervene X so that X can be free from the effect of $C \rightarrow X$. For example, randomized experiments can 'physically' intervene X by recollecting data from a prohibitively large quantity of random samples to remove the confounding bias. However, the efficiency and feasibility of such solutions are low because of the limited resources in practice.

Alternatively, we can use the do-calculus to cut off the connection of X with its parents. Specifically, we formulate $P(Y|do(X))$ as the objective function:

$$
\begin{aligned}
P(Y|do(X)) &= \sum_c P(Y|do(X), c)P(c|do(X)) \\
&= \sum_c P(Y|X, c)P(c|do(X)) \\
&= \sum_c P(Y|X, c)P(c),
\end{aligned} \tag{1}
$$

where the first equality is given by the law of total probability, and the next two equalities are due to the rule of backdoor criterion [20], since the only backdoor path $X \leftarrow C \rightarrow Y$ has been blocked when conditioning on node C by $do(X)$. However, since $\{c\}$ is unobserved in most cases, the difficulty of optimizing the above equation lies in how to endow $\{c\}$ with meaningful semantics in data, thus obtaining a tractable objective from Eq. (1).

In order to solve the problem, the variational distribution $Q(C|X)$ is introduced into Eq. (1), which is a variational approximation to the prior $P(C)$. By embodying $Q(C|X)$ with semantics underlying the data, we can optimize the network parameters with scenarios uncovered in the observed data. Specifically, we have the lower bound of Eq. (1) that we need to maximize as follows:

$$
\begin{aligned}
&\log \sum_c P(Y|X, c)P(c) \\
&= \log \sum_c Q(c|X)P(Y|X, c)\frac{P(c)}{Q(c|X)} \\
&\geq \sum_c Q(c|X)\log P(Y|X, c)\frac{P(c)}{Q(c|X)} \\
&= \mathbb{E}_{Q(C|X)} \log P(Y|X, C) - D_{KL}[Q(C|X)||P(C)],
\end{aligned} \tag{2}
$$

where the second step of derivation uses the Jensen's inequality. In the above equation, the variational distribution $Q(C|X)$ connects the scenarios and observed data through the first predictive term, and the divergence between $Q(C|X)$ and $P(C)$ is penalized with the second Kullback-Leibler (KL) term.

3.1.1 Scenario Learning. The remaining problem is to compute the variational distribution $Q(C|X)$ and we use a scenario learner to approximate it in this work. Specifically, $Q(C|X)$ can be considered as a latent variable which identifies the scenario c for each data sample. We propose to learn $Q(C|X)$ via prototypical clustering [16], whose key step is to assign each data sample to a cluster/scenario by computing the distance between the data sample and the scenario prototype. We consider both soft version and hard version of the scenario assignment for scenario learning.

- **Soft version.** In soft version, each data sample can be assigned to all the scenarios with certain probabilities. In this case, the scenario learner produces a distribution $Q_s(c|x)$ over various scenarios for each x based on a softmax over distances (e.g., Euclidean distance) to the scenario prototypes in the metric space ω:

$$
Q_s(c|x) = P_\omega(\mathbf{c}|\mathbf{x}) = \frac{\exp(-d(f_\omega(\mathbf{x}), \mathbf{c}))}{\sum_{\mathbf{c'}} \exp(-d(f_\omega(\mathbf{x}), \mathbf{c'}))}, \tag{3}
$$

where $d(\cdot)$ is the distance function, $\mathbf{c}$ is the learnable vector representing scenario prototypes, $\mathbf{x}$ is the embedding of data sample x and function $f_\omega(\cdot)$ denotes the projection operation with the transformation matrix ω.

[1] We do not draw it in the figure for brevity.

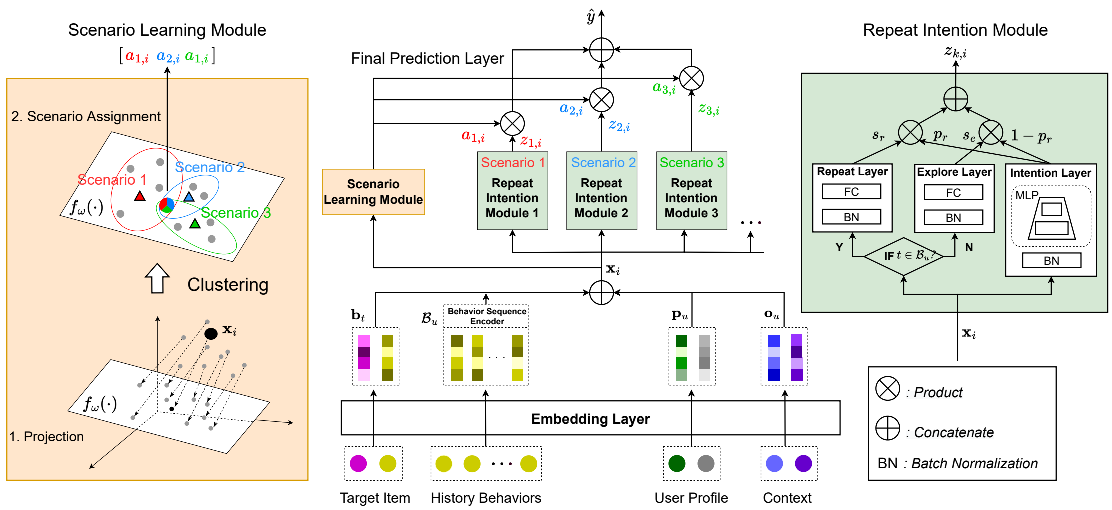


Figure 3: Overall architecture of the proposed MuSeNet model.

- **Hard version.** In hard version, each data sample is assigned to one of the identified scenarios and we choose the scenario whose prototype is closest to the data sample in the metric space. That is, we can identify the scenarios as well as the assignment of each data sample by

$$Q_h(c|x) = \begin{cases} 1 & if\ c = arg\max_{c'} Q_s(c'|x), \\ 0 & otherwise. \end{cases} \tag{4}$$

Remarks. Note that we measure the distance between the data samples and their respective scenario prototypes in the metric space ω. The reason for using this metric space is that different feature dimensions should be weighted differently in determining the cluster structure, which might be beneficial for the predictive task. This is also in alignment with the work on joint factor analysis and latent clustering [40].

3.1.2 Repeat-aware Mechanism. Next, we take the repeat intention of users into consideration. We propose a new repeat-aware mechanism, and assimilate it into Eq. (2). The repeat-aware mechanism first infers the user's repeat intention, i.e., whether the user desires to repeat a consumed item before or explore a new item that might be of interest; it then predicts the user's affinity to the item conditioned on the inferred intention. Particularly, we transform $P(Y|X,C)$ as:

$$\begin{aligned} P(Y|X,C) =& P(r|X,C)P(Y|r,X,C) \\ &+ P(e|X,C)P(Y|e,X,C), \end{aligned} \tag{5}$$

where r and e denote repeat mode and explore mode, respectively. Then, absorbing Eq. (5) into Eq. (2), our objective function becomes:

$$\begin{aligned} &\mathbb{E}_{Q(C|X)} \log[P(r|X,C)P(Y|r,X,C) \\ &+ P(e|X,C)P(Y|e,X,C)] - D_{KL}[Q(C|X)||P(C)]. \end{aligned} \tag{6}$$

Note that we use the repeat-aware mechanism as an illustration to show the benefit of scenario learning. Other scenario-specific behavior patterns can also be similarly encoded in our framework (e.g., users may prefer business hotels during business trips, users may prefer a nearby hotel late at night, etc.), and we leave them as future work.

3.2 Instantiation of MuSeNet

Next, we present how we initialize the MuSeNet framework with a scenario learning module and a repeat intention modeling module. The workflow of MuSeNet is shown in Fig. 3.

Let $\Lambda = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K\}$ denote the learned prototype vectors of K scenarios, and Θ denote the other parameters (i.e., the transformation matrix ω in the scenario learning module and network parameters φ in the repeat intention modeling module). The scenario assignment for each data sample x is a categorical variable represented by a K-dimensional vector $\mathbf{a}$, and we use $a_{k,i}$ to indicate the k-th value of vector $\mathbf{a}$ for the i-th data sample. We use $\mathcal{X}$ to denote the training dataset, and use $N = |\mathcal{X}|$ to denote the number of training samples.

3.2.1 Scenario Learning Module. There are two learnable parameters related with the scenario learner: the transformation matrix ω and the scenario prototype Λ. Here, we consider ω first. For the soft version in Eq. (3), ω can be directly updated by back-propagation since the distribution $Q_s(C|X)$ is continuous. However, for the hard version in Eq. (4), the assignment probability $a_{k,i}$ is discrete, which cannot be back-propagated directly. To solve this issue, we adopt the Gumbel-Softmax trick [14] and assume

$$a_{k,i} = \frac{\exp((\log \pi_{k,i} + g_{k,i})/\tau)}{\sum_{k'=1}^{K} \exp((\log \pi_{k',i} + g_{k',i})/\tau)}, \tag{7}$$

where $\pi_{k,i} = \frac{\exp(-d(f_\omega(\mathbf{x}_i),\mathbf{c}_k))}{\sum_{k'=1}^{K}\exp(-d(f_\omega(\mathbf{x}_i),\mathbf{c}_{k'}))}$, $g_{k,i}$ are i.i.d. samples drawn from distribution Gumbel$(0,1)$, and τ is a temperature parameter to control the discreteness of $\mathbf{a}_i$. As τ approaches 0, $\mathbf{a}_i$ becomes nearly indistinguishable from a one-hot vector. We set τ to 0.1 by default.

As for the scenario prototype Λ, given ω, the objective is to minimize the distance expectation from each data sample to the corresponding scenario prototypes as follows,

$$\mathcal{L}_C(\Lambda, \Theta) = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} a_{k,i} d(f_\omega(\mathbf{x}_i), \mathbf{c}_k). \tag{8}$$

In other words, we aim to adjust the scenario prototype vectors to minimize the above loss given the current assignment probability $\mathbf{a}$.

3.2.2 Repeat Intention Modeling Module. This module consists of three components: an intention inference layer that computes the probability of the user's repeat intentions, i.e., repeat or explore; a repeat layer that outputs affinity score of the user towards the target item conditioned on the repeat intention; and an exploration layer that outputs affinity score under the explore intention. We use multi-layer perceptrons (MLP) for these layers. For example, the output from the intention inference layer is $p_r = \sigma(MLP([p_u, \mathcal{B}_u, o_u]))$, where $\sigma(\cdot)$ is the sigmoid activation function, p_r is the probability that the user intends to repeat, and $1 - p_r$ indicates the exploration probability. Similarly, the output score s_r from the repeat layer and s_e from the explore layer are both from a MLP layer with four types of input features (i.e., $[p_u, \mathcal{B}_u, b_t, o_u]$). The final output of x_i from the k-th repeat intention module is

$$z_{k,i} = \mathbb{I}[t \in \mathcal{B}_u] p_r s_r + (1 - \mathbb{I}[t \in \mathcal{B}_u])(1 - p_r) s_e, \tag{9}$$

where $\mathbb{I}[t \in \mathcal{B}_u]$ is an indicator function that tells whether the target item t is in the user's past behavior sequence or not. The score $z_{k,i}$ indicates the user's preferences towards the target item considering the user's repeat intentions within the k-th scenario.

One issue about the repeat intention modeling is that, the ground-truth label y only supervises the final preference prediction, which might not be sufficient to supervise the inference of the user's repeat intentions. Therefore, we additionally propose an auxiliary loss function to supervise such intention prediction p_r, as follows,

$$\mathcal{L}_A(\Lambda, \Theta) = -\frac{1}{N} \sum_{i=1}^{N} (r_i - \sum_{k=1}^{K} p_r^k a_{k,i})^2, \tag{10}$$

where r_i is the auxiliary label indicating the fraction of items the user consumed before. The auxiliary label characterizes the user's actual proportion of repeat behaviors. The second term in the parenthesis is the predicted repeat propensity across all the scenarios where p_r^k is the repeat probability of the k-th scenario.

3.2.3 Final Prediction. The final predicted score of the user interacting with the target item in the hard version is simply $\hat{y}_i = \sigma(z_{k,i})$ if the data sample x_i belongs to the k-th scenario. With the soft version of the multi-scenario learning, the final score is the linear combination of the output from each of the repeat intention module weighted by the soft cluster membership, i.e., $\hat{y}_i = \sigma(\sum_{k=1}^{K} a_{k,i} z_{k,i})$.

We collect training data $\mathcal{X}$, which consists of users' engagement logs as triad (u, t, y), i.e., user u interacts with target item t if label $y = 1$. Our main goal is to accurately predict the interaction label y, and thus we minimize the following loss function with cross-entropy for this purpose:

$$\mathcal{L}_T(\Lambda, \Theta) = -\frac{1}{N} \sum_{(u,t,y) \in \mathcal{X}} (y \log \hat{y} + (1 - y) \log(1 - \hat{y})). \tag{11}$$

3.2.4 Training and Optimization. The final goal of model training is to maximize the objective in Eq. (6) given the cluster structure. The first expectation term in Eq. (6) can be optimized by $\mathcal{L}_T(\Lambda, \Theta)$, which predicts label y. For the second KL term, we assume that the prior $P(C)$ obeys a uniform distribution (i.e., $P(C) = 1/K$), and re-write the cost function of $D_{KL}[Q(C|X)||P(C)]$ as:

$$\begin{aligned} \mathcal{D}_{KL}(\tilde{\Lambda}, \Theta) &= \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} a_{k,i} \log \frac{a_{k,i}}{P(C)} \\ &= \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} a_{k,i} (\log a_{k,i} + \log K). \end{aligned} \tag{12}$$

Since MuSeNet incorporates the repeat intention prediction as an auxiliary task, we can jointly train both two tasks (i.e., Eq. (10) and Eq. (11)) in a linear combination with a hyperparameter β. Putting all together, we have the final optimization objective:

$$\begin{aligned} &\min_{\tilde{\Lambda}, \Theta} \mathcal{L}_T(\tilde{\Lambda}, \Theta) + \mathcal{D}_{KL}(\tilde{\Lambda}, \Theta) + \beta \mathcal{L}_A(\tilde{\Lambda}, \Theta), \\ &\text{s.t.} \quad \tilde{\Lambda} = arg \min_{\Lambda} \mathcal{L}_C(\Lambda, \Theta). \end{aligned} \tag{13}$$

This is a standard bi-level optimization problem and we have two blocks of variables, namely, the outer variables Θ and the inner variables Λ. Inspired by [6], we can apply the algorithm of alternating optimization to minimize the above loss by performing the following two steps iteratively.

Step 1: *Update* Λ *by fixing* Θ. In this case, the prototypes of scenarios can be updated in an online fashion as

$$\mathbf{c}_k^{t+1} = (1 - \alpha) \mathbf{c}_k^t + \alpha \frac{\sum_{i=1}^{S} a_{k,i} \mathbf{x}_i}{\sum_{i=1}^{S} a_{k,i}}, \tag{14}$$

where t in the superscript is the training steps, S is the batch size, and α controls how much we should discount the prototypes from the last step. Here, we fix α to 0.2 by default.

Step 2: *Update* Θ *by fixing* Λ. Given Λ, this step can be done directly by back-propagation in both soft version and hard version of multi-scenario learning.

3.3 Analysis and Discussions

3.3.1 Scenario Learning as a Linear Model. Here, we present a simple but useful analysis which can offer more insight into the nature of the scenario learning. When we use Euclidean distance as the measure of $d(\cdot)$, the model in Eq. (3) can be turned into a linear model with a particular parameterization [18] as:

$$-\|f_\omega(\mathbf{x}) - \mathbf{c}_k\|_2^2 = -f_\omega(\mathbf{x})^T f_\omega(\mathbf{x}) + 2\mathbf{c}_k^T f_\omega(\mathbf{x}) - \mathbf{c}_k^T \mathbf{c}_k. \tag{15}$$

Since the first term in the above equation is independent w.r.t. the scenario c_k, it does not affect the probabilities after softmax. Thus, we can focus on the latter two terms and let

$$2\mathbf{c}_k^T f_\omega(\mathbf{x}) - \mathbf{c}_k^T \mathbf{c}_k = \mathbf{w}_k^T f_\omega(\mathbf{x}) + b_k, \tag{16}$$

where $\mathbf{w}_k = 2\mathbf{c}_k$, $b_k = -\mathbf{c}_k^T \mathbf{c}_k$.

Now, we can observe that the scenario learning can be interpreted as a linear model, which is a reasonable choice and commonly adopted by existing deep clustering models [3, 4, 29]. The reason is that the necessary non-linearity parts can be learned within the previous embedding process.

3.3.2 *Comparison with Correlation* $P(Y|X)$.

In the beginning, we point out that traditional methods are designed to directly estimate the correlation $P(Y|X)$ and may suffer from the heterogeneity of unknown scenarios. Here, we give a further analysis to offer more insights into the difference between $P(Y|X)$ and $P(Y|do(X))$.

We first rewrite $P(Y|X)$ with the following steps:

$$\begin{aligned} P(Y|X) &= \sum_c P(Y, c|X) \\ &= \sum_c P(Y|X, c)P(c|X) \\ &\propto \sum_c P(Y|X, c)P(X|c)P(c), \end{aligned} \tag{17}$$

where the first step is because of the definition of marginal distribution, and the latter two steps are the results of the Bayes' theorem.

We can observe from the above equation that both conditional probability terms $P(Y|X, c)$ and $P(X|c)$ are implicit, since scenario c is unknown for recommender models. In this case, deep models are expected to spontaneously build conditional probability formulas corresponding to scenarios, but this goal is difficult to achieve because neural networks have trouble in fitting product relationships without the explicit design [22].

Next, if recommendation models resolve to explicitly identify scenarios, the only difference between $P(Y|X)$ in Eq. (17) and $P(Y|do(X))$ in Eq. (1) is that $P(Y|X)$ has an additional term: $P(X|C)$ (i.e., the causal path $C \rightarrow X$ in Fig. 2(b)). Then, we can put it into Eq. (2) and similarly obtain

$$\begin{aligned} &\log \sum_c P(Y|X, c)P(X|c)P(c) \\ &\geq \mathbb{E}_{Q(C|X)}[\log P(Y|X, C) - \log \frac{Q(C|X)}{P(X|C)P(C)}]. \end{aligned} \tag{18}$$

From Eq. (18), we can see that the additional $P(X|C)$ is introduced in the second KL term, which fundamentally affects the model training. For example, suppose $C = c$ is a scenario and $X = x$ is an input sample deviating from the prototype of c, the value of the KL term is close to negative infinity since $P(x|c)$ is a small value tending to 0. That is to say, the lower bound to maximize becomes not as tight as before, which can hinder the training process.

4 EXPERIMENTS

4.1 Experimental Setup

Datasets. We conduct experiments on two datasets: one proprietary dataset collected from an industrial level online travel platform's recommender system (*OTP*) and a public dataset[2] (*ADS*). *OTP* is extracted from four weeks of users' behavior logs where those impressed and clicked samples are labeled as positive and those impressed but not clicked are labeled as negative. The dataset is further split into training (first 21 days) and testing (last 7 days) sets. *ADS* is a public dataset released by Alimama for the purpose of click-through rate (CTR) prediction of ads. We divide the data into seven equal parts according to timestamps, and use the first six parts for training and the rest part for testing. The statistics of the two datasets are shown in Table 1.

[2]https://tianchi.aliyun.com/dataset/dataDetail?dataId=56

Table 1: Statistics of the datasets.

Datasets	*OTP*	*ADS*
# of users	2,532,809	1,141,729
# of items	357,696	846,811
# of samples	68,526,336	26,557,961
click-through rate	7.25%	5.14%
repeat rate	18.9%	2.62%

Evaluation Metrics. We use AUC, GAUC (Group AUC) [46], and Logloss [35] to measure the performance of the models. GAUC measures the average AUC per ranking result. Logloss measures the distance between the predicted CTR and the ground truth. In addition, for both AUC and GAUC, we use the RelaImpr metric as defined in [39, 45], which has been widely used in industry to show the relative improvement for the evaluated models.

Comparison Methods. Since we focus on the CTR prediction task, we compare the proposed model with the following models.

- *BaseModel*. A baseline model with embedding & MLP architecture.
- *Wide&Deep* [7]. Wide&Deep is a widely used deep model for CTR prediction in industrial applications. It consists of a wide model for memorization and a deep model for generalization.
- *NCF* [13]. NCF proposes to use neural networks to substitute the inner product used in collaborative filtering so as to better learn the prediction function from user-item interactions.
- *DIN* [45]. DIN is a CTR prediction model that captures relationships between the target item and user's historical behavior sequence via the attention mechanism.
- *RepeatNet* [21]. RepeatNet incorporates repeat and explore modes into models with an encoder-decoder structure.
- *DCN-V2* [35]. DCN-V2 designs a special cross layer to achieve element-wise and feature-wise feature interactions.
- *PLE* [31]. PLE is a multi-task recommendation model that considers both shared components and specific components among different tasks, and uses a progressive layered extraction mechanism to effectively transfer the knowledge.
- *RevMan* [17]. RevMan models multiple sale scenarios for insurance recommendation with interactions among scenarios.

The former six baselines are single-scenario models (corresponding to Fig. 2(a)), i.e., they do not consider the scenarios behind the data. The latter two baselines are multi-scenario models but extract scenarios based on pre-defined rules (corresponding to Fig. 2(b)). Since such explicit scenarios are unavailable in our datasets, we manually segment the dataset into different scenarios. For *OTP*, we setup four scenarios based on whether the data sample is a same-day and/or same-city hotel booking. For *ADS*, we use the user's shopping-level attributes (i.e., high-frequency buyers, mid-frequency buyers, or low-frequency buyers) in the dataset to define three scenarios.

Implementations. We implement all the methods in the Tensorflow distributed training framework with 30 workers. The learning rate is set to 0.001 with Adam optimizer. The hyperparameters K and β in Eq. (8) and Eq. (13) are set to 4 and 0.5 by default, respectively. We will empirically evaluate these two parameters.

Table 2: Comparison results of different methods on *OTP* and *ADS*. The proposed MuSeNet achieves the best results.

	OTP					*ADS*		
Model	AUC	RelaImpr	GAUC	RelaImpr	Logloss	AUC	RelaImpr	Logloss
BaseModel	0.7521	0.00%	0.7321	0.00%	0.2238	0.6171	0.00%	0.1963
Wide&Deep	0.7613	3.64%	0.7340	0.83%	0.2247	0.6190	1.71%	0.1957
NCF	0.7544	0.91%	0.7326	0.22%	0.2290	0.6208	3.16%	0.1952
DIN	0.7617	3.82%	0.7334	0.56%	0.2401	0.6206	3.05%	0.1953
RepeatNet	0.7614	3.68%	0.7343	0.94%	0.2235	0.6197	2.23%	0.1958
DCN-V2	0.7622	4.01%	0.7336	0.65%	0.2193	0.6201	2.58%	0.1957
PLE	0.7636	4.56%	0.7345	1.03%	0.2186	0.6187	1.38%	0.1960
RevMan	0.7627	4.22%	0.7341	0.85%	0.2191	0.6173	0.18%	0.1962
Hard Version	0.7690	6.73%	0.7386	2.81%	**0.2169**	0.6215	3.77%	0.1952
Soft Version	**0.7706**	**7.33%**	**0.7392**	**3.06%**	0.2171	**0.6234**	**5.40%**	**0.1950**

4.2 Experimental Results

4.2.1 Effectiveness Comparison. We first compare the overall effectiveness of different methods and show the results in Table 2. On *OTP*, the proposed MuSeNet generally outperforms its competitors on all the three evaluation metrics (i.e., AUC, GAUC, and Logloss). The best competitor in single-scenario models (i.e., DCN-V2) and the best one in multi-scenario models (i.e., PLE) achieve AUC gains of 4.01% and 4.56%, respectively. This indicates that even using the scenarios we manually defined, PLE is still better than the best single-scenario competitor, demonstrating the benefit of modeling multiple scenarios. Further, by learning implicit scenarios automatically and modeling the repeat intentions for each scenario, MuSeNet (soft) is 2.77% better than PLE in terms of AUC, achieving 7.33% AUC gains. We have similar observations for the other metrics.

On the public dataset *ADS*, MuSeNet still outperforms its competitors. Different from the results on *OTP*, the multi-scenario competitors (i.e., PLE and RevMan) only perform comparably with the single-scenario models. The possible reason is that the manually defined scenarios are unsuitable for this dataset. This also shows the difficulty and importance of identifying good scenarios. Our model, on the other side, still achieves the best performance over all competitors. This means that through multi-scenario learning, we can discover implicit scenarios that are friendly to the recommendation task. Note that the ranking results per request is unavailable on *ADS*, and thus we do not report the GAUC results.

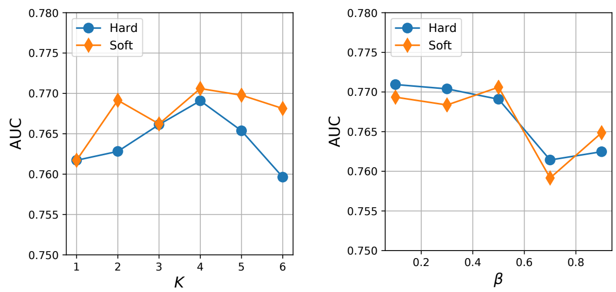
4.2.2 Ablation study. Next, we conduct an ablation study on the two key components of the proposed model, i.e., the scenario learning module and the repeat intention modeling module. The results are shown in Table 3 where we show the AUC and GAUC results on *OTP* for brevity. In the table, 'MuSeNet on predefined scenarios' means that we use the manually defined four scenarios as stated in the experimental setup instead of learning them. We can observe from the table that: (1) removing both modules achieves the worst performance, which demonstrates the importance of these two modules; (2) replacing scenario learning with pre-defined scenarios also brings drop on the performance, validating the superiority of automatically identifying the scenario structure from the data; (3) repeat intention modeling also helps improve the AUC score, emphasizing the benefit of modeling scenario-specific repeat intentions.

Table 3: Results of the ablation study on scenario learning (M1) and repeat intention modeling (M2). Both components are useful for improving the performance.

Model	AUC	GAUC
MuSeNet w/o **M1**&**M2**	0.7617	0.7334
MuSeNet on predefined scenarios	0.7650	0.7351
MuSeNet (hard) w/o **M2**	0.7667	0.7363
MuSeNet (soft) w/o **M2**	0.7678	0.7379
MuSeNet (hard)	0.7691	0.7386
MuSeNet (soft)	**0.7706**	**0.7392**

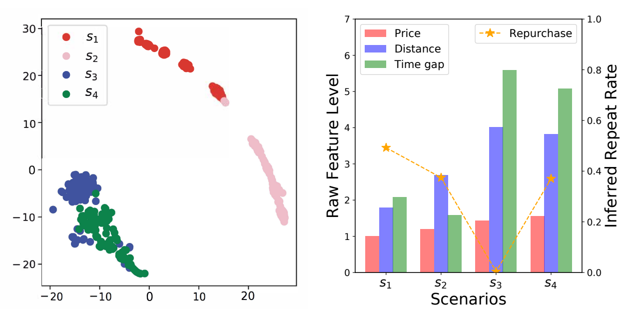
4.2.3 Parameter sensitivity. We next perform parameter sensitivity analysis on the two important hyperparameters of our model, i.e., K, the number of scenarios, and β, which controls the importance of the auxiliary loss function. The results are shown in Fig. 4 where we show the AUC results on *OTP* for brevity. For the parameter K, AUC increases with K and peaks at $K = 4$ and then decreases as shown in Fig. 4(a). The possible reason is that larger K makes the data sparser in each scenario, thus reducing the recommendation accuracy for the hard version MuSeNet. In contrast, MuSeNet (soft) is less sensitive to K as it may still make use of all the data samples. For the parameter β, AUC is stable in a large range from 0.02 to 0.5.

4.2.4 Visualization. Finally, we randomly choose 100 data samples with softmax probability greater than 0.8 from each scenario based on the results of MuSeNet (soft), and plot their embeddings in the Fig. 5(a) using t-SNE. We can see a clear cluster structure, which shows the effectiveness of scenario learning at identifying the data scenarios. In Fig. 5(b), we plot the average value of three most important raw features for each of the scenario on the *OTP* dataset, namely, the price for booking the hotel, the distance from the user to the hotel, and the time gap between booking and check-in date. We observe a clear distribution difference, where lower price, shorter distance and time gap are present in scenarios s_1 and s_2, in clear



(a) AUC vs. K (b) AUC vs. β

Figure 4: Parameter sensitivity analysis. Best result is achieved when $K = 4$, and the results are relatively stable in a wide range for β.



(a) Embeddings from four scenarios (b) Feature differences of the learned scenarios

Figure 5: Visualization Results on the learned scenarios. Different scenarios exhibit significant differences w.r.t. the raw features and repeat intention.

contrast with s_3 and s_4. This indicates that the first two scenarios might describe a short to medium distance trip, with less time to make plans; on the contrary, the latter two scenarios might characterize a long-distance leisure or business trip in the future (having sufficient time to make plans). The inferred repurchase intention is also plotted. For the latter two scenarios, we observe that s_4 has a much higher repeat rate compared to s_3, indicating that the repeat intention might separate these two scenarios.

5 RELATED WORK

Personalized Recommendation. Two current focuses of personalized recommendation are modeling feature interactions and modeling users' sequential behaviors. For the former, Wide&Deep [7] combines the generalization and memorization features with MLP networks; DeepFM [12] learns both low-order and high-order feature interactions; DCN-v2 [35] proposes the so-called cross network structure to better capture second-order feature interactions. For the latter, DIN [45] introduces the attention mechanism to learn users' historical preferences; Atrank [44] considers various user behaviors and projects different types of behaviors into latent semantic spaces; HIFN [27] devises a self-supervised framework to capture long-term and short-term user interests in an adaptive manner.

Different from the above work, our focus is on the *multi-scenario* aspect of recommender systems. In terms of multi-scenario in single recommendation task, existing work mainly requires manual or explicit separation of scenarios. For example, RevMan [17] separates insurance sales data into four partitions, and builds an expert network for each partition; TreeMS [19] maintains a tree structure for explicit scenarios, and adapts the mixture of experts model to make recommendations; STAR [28] leverages the star topology among known scenarios to make CTR predictions. Such methods become less effective when the explicit scenarios and their relationships are unavailable. In this work, we propose to automatically learn the scenarios. Our work is also different from earlier work that clusters either users or items [9, 11]. We cluster the behaviors between users and items, and jointly optimize clustering and recommendation.

Causal Recommendation. Recently, causal inference has become a hot research topic in recommender systems, as it reveals the nature of data generation with the systems [24, 30, 36, 42, 43]. For example, ERM [24] uses causal inference to reduce data selection bias in the matrix factorization framework; MACR [36] and PDA [42] discuss how to handle the item popularity bias properly from the view of data generation; DICE [43] proposes a general framework of representation learning to decouple user preferences and hot items with causal embedding; CountER [30] introduces the counterfactual reasoning from causal inference to improve the explainability of recommender systems. Different from the above work, our focus is to analyze the scenarios underlying the data for better recommendation.

Repurchase Modeling. Repurchase modeling is studied in recommendation settings where users may re-consume the items [1]. For example, Chen et al. [5] propose fast algorithms to predict the short-term repurchase intentions; MPG [2] models the repurchase probability periodically for industry-level e-commerce data; RepeatNet [21] proposes an encoder-decoder structure with a repeat recommendation mechanism; Wang et al. [33] consider multiple influential factors and design an adaptive method to model the factors. In this paper, we connect repurchase modeling with scenario learning.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a MuSeNet framework that jointly learns the implicit scenarios from existing user behaviors and predicts the user preferences to items. We explain scenario learning from a causal view and propose to mitigate the confounding bias during scenario learning. For each learned scenario, we also use a repeat-aware mechanism to model the probability of repetitively consuming an item or exploring a new item. We conduct experiments on both industrial and public datasets, and the results show the effectiveness of our approach. In the future, we plan to explore how to automatically adjust the number of scenarios in the training process.

ACKNOWLEDGEMENTS

This work is supported by the National Natural Science Foundation of China (No. 62025202), and the Collaborative Innovation Center of Novel Software Technology and Industrialization. Hanghang Tong is partially supported by NSF (1947135). Yuan Yao is the corresponding author.

REFERENCES

[1] Ashton Anderson, Ravi Kumar, Andrew Tomkins, and Sergei Vassilvitskii. 2014. The dynamics of repeat consumption. In *WWW*. 419–430.

[2] Rahul Bhagat, Srevatsan Muralidharan, Alex Lobzhanidze, and Shankar Vishwanath. 2018. Buy it again: Modeling repeat purchase recommendations. In *SIGKDD*. 62–70.

[3] Deyu Bo, Xiao Wang, Chuan Shi, Meiqi Zhu, Emiao Lu, and Peng Cui. 2020. Structural deep clustering network. In *WWW*. 1400–1410.

[4] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. 2018. Deep clustering for unsupervised learning of visual features. In *ECCV*. 132–149.

[5] Jun Chen, Chaokun Wang, and Jianmin Wang. 2015. Will you" reconsume" the near past? fast prediction on short-term reconsumption behaviors. In *AAAI*.

[6] Tianyi Chen, Yuejiao Sun, and Wotao Yin. 2021. Tighter analysis of alternating stochastic gradient method for stochastic nested problems. *arXiv preprint arXiv:2106.13781* (2021).

[7] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Workshop on deep learning for recommender systems*. 7–10.

[8] Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *AAAI*. 67–73.

[9] Thomas George and Srujana Merugu. 2005. A scalable collaborative filtering framework based on co-clustering. In *ICDM*.

[10] Madelyn Glymour, Judea Pearl, and Nicholas P Jewell. 2016. *Causal inference in statistics: A primer*. John Wiley & Sons.

[11] Songjie Gong. 2010. A collaborative filtering recommendation algorithm based on user clustering and item clustering. *J. Softw.* 5, 7 (2010), 745–752.

[12] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *IJCAI*. 1725–1731.

[13] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. 173–182.

[14] Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *ICLR*.

[15] Chao Li, Zhiyuan Liu, Mengmeng Wu, Yuchi Xu, Huan Zhao, Pipei Huang, Guoliang Kang, Qiwei Chen, Wei Li, and Dik Lun Lee. 2019. Multi-Interest Network with Dynamic Routing for Recommendation at Tmall. In *CIKM*. 2615–2623.

[16] Junnan Li, Pan Zhou, Caiming Xiong, and Steven CH Hoi. 2021. Prototypical contrastive learning of unsupervised representations. In *ICLR*.

[17] Yu Li, Yi Zhang, Lu Gan, Gengwei Hong, Zimu Zhou, and Qiang Li. 2021. RevMan: Revenue-aware multi-task online insurance recommendation. In *AAAI*.

[18] Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. 2013. Distance-based image classification: Generalizing to new classes at near-zero cost. *IEEE transactions on pattern analysis and machine intelligence* 35, 11 (2013), 2624–2637.

[19] Xichuan Niu, Bofang Li, Chenliang Li, Jun Tan, Rong Xiao, and Hongbo Deng. 2021. Heterogeneous Graph Augmented Multi-Scenario Sharing Recommendation with Tree-Guided Expert Networks. In *WSDM*. 1038–1046.

[20] Judea Pearl. 2009. *Causality*. Cambridge university press.

[21] Pengjie Ren, Zhumin Chen, Jing Li, Zhaochun Ren, Jun Ma, and Maarten De Rijke. 2019. Repeatnet: A repeat aware neural recommendation machine for session-based recommendation. In *AAAI*. 4806–4813.

[22] Steffen Rendle, Walid Krichene, Li Zhang, and John Anderson. 2020. Neural collaborative filtering vs. matrix factorization revisited. In *RecSys*. 240–248.

[23] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " Why should i trust you?" Explaining the predictions of any classifier. In *SIGKDD*. 1135–1144.

[24] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as treatments: Debiasing learning and evaluation. In *international conference on machine learning*. PMLR, 1670–1679.

[25] Qijie Shen, Wanjie Tao, Jing Zhang, Hong Wen, Zulong Chen, and Quan Lu. 2021. SAR-Net: A Scenario-Aware Ranking Network for Personalized Fair Recommendation in Hundreds of Travel Scenarios. In *CIKM*. 4094–4103.

[26] Qijie Shen, Hong Wen, Wanjie Tao, Jing Zhang, Fuyu Lv, Zulong Chen, and Zhao Li. 2022. Deep Interest Highlight Network for Click-Through Rate Prediction in Trigger-Induced Recommendation. In *WWW*. 422–430.

[27] Qijie Shen, Hong Wen, Jing Zhang, and Qi Rao. 2022. Hierarchically Fusing Long and Short-Term User Interests for Click-Through Rate Prediction in Product Search. In *CIKM*. 1767–1776.

[28] Xiang-Rong Sheng, Liqin Zhao, Guorui Zhou, Xinyao Ding, Binding Dai, Qiang Luo, Siran Yang, Jingshan Lv, Chi Zhang, Hongbo Deng, et al. 2021. One Model to Serve All: Star Topology Adaptive Recommender for Multi-Domain CTR Prediction. In *CIKM*. 4104–4113.

[29] Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. *Advances in neural information processing systems* 30 (2017).

[30] Juntao Tan, Shuyuan Xu, Yingqiang Ge, Yunqi Li, Xu Chen, and Yongfeng Zhang. 2021. Counterfactual explainable recommendation. In *CIKM*. 1784–1793.

[31] Hongyan Tang, Junning Liu, Ming Zhao, and Xudong Gong. 2020. Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In *RecSys*. 269–278.

[32] Georgios Theocharous, Philip S. Thomas, and Mohammad Ghavamzadeh. 2015. Personalized Ad Recommendation Systems for Life-Time Value Optimization with Guarantees. In *IJCAI*.

[33] Chenyang Wang. 2021. Towards Dynamic User Intention in Sequential Recommendation. In *WSDM*. 1121–1122.

[34] Chenyang Wang, Min Zhang, Weizhi Ma, Yiqun Liu, and Shaoping Ma. 2019. Modeling Item-Specific Temporal Dynamics of Repeat Consumption for Recommender Systems. In *WWW*. 1977–1987.

[35] Ruoxi Wang, Rakesh Shivanna, Derek Cheng, Sagar Jain, Dong Lin, Lichan Hong, and Ed Chi. 2021. DCN V2: Improved Deep & Cross Network and Practical Lessons for Web-scale Learning to Rank Systems. In *WWW*. 1785–1797.

[36] Tianxin Wei, Fuli Feng, Jiawei Chen, Ziwei Wu, Jinfeng Yi, and Xiangnan He. 2021. Model-agnostic counterfactual reasoning for eliminating popularity bias in recommender system. In *SIGKDD*. 1791–1800.

[37] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. 2019. MMGCN: Multi-modal graph convolution network for personalized recommendation of micro-video. In *MM*. 1437–1445.

[38] Kai Xiao, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. 2020. Noise or signal: The role of image backgrounds in object recognition. *arXiv preprint arXiv:2006.09994* (2020).

[39] Ling Yan, Wu-jun Li, Gui-Rong Xue, and Dingyi Han. 2014. Coupled group lasso for web-scale ctr prediction in display advertising. In *ICML*. 802–810.

[40] Bo Yang, Xiao Fu, and Nicholas D Sidiropoulos. 2016. Learning from hidden traits: Joint factor analysis and latent clustering. *IEEE Transactions on Signal Processing* 65, 1 (2016), 256–269.

[41] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *CSUR* 52, 1 (2019), 1–38.

[42] Yang Zhang, Fuli Feng, Xiangnan He, Tianxin Wei, Chonggang Song, Guohui Ling, and Yongdong Zhang. 2021. Causal intervention for leveraging popularity bias in recommendation. In *SIGIR*. 11–20.

[43] Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Yong Li, and Depeng Jin. 2021. Disentangling user interest and conformity for recommendation with causal embedding. In *WWW*. 2980–2991.

[44] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2018. Atrank: An attention-based user behavior modeling framework for recommendation. In *AAAI*, Vol. 32.

[45] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *SIGKDD*. 1059–1068.

[46] Han Zhu, Junqi Jin, Chang Tan, Fei Pan, Yifan Zeng, Han Li, and Kun Gai. 2017. Optimized cost per click in taobao display advertising. In *SIGKDD*. 2191–2200.