序列推荐中的双重增强倾向得分估计

Chen Xu Jun Xu*
Xu Chen 高瓴人工智能学院 中国人民大学 高瓴人工智能学院 中国人民大学 高瓴人工智能学院 中国人民大学 xc_chen@ruc.edu.cn
junxu@ruc.edu.cn xu.chen@ruc.edu.cn

Zhenghua Dong 华为诺亚方舟实验室
dongzhenhua@huawei.com

Ji-Rong Wen 高瓴人工智能学院 中国人民大学
jrwen@ruc.edu.cn

arXiv:2303.08722v1 [cs.IR] 15 Mar 2023

摘要

序列推荐系统基于大量隐式用户反馈数据训练模型；当用户系统性地对某些物品曝光不足或过度曝光时，模型可能会受到偏差影响。为解决这一问题，已有研究提出基于逆倾向得分（IPS）的无偏学习方法。逆倾向得分用于估计给定历史信息时观测到某个用户-物品对的概率。在这些方法中，倾向得分估计通常局限于物品视图，即将反馈数据视为与用户交互过的物品序列。然而，也可以从用户视图来考察反馈数据，即将其视为与物品交互过的用户序列。此外，这两种视图能够共同增强倾向得分估计。受这一观察启发，我们提出从用户和物品视图估计倾向得分的方法，称为双重增强倾向得分估计（DEPS）。具体而言，给定一个目标用户-物品对及相应的物品和用户交互序列，DEPS 首先构建时间感知因果图，以表示用户-物品观测概率。根据该图，分别从物品视图和用户视图出发，基于同一组用户反馈数据估计两个互补的倾向得分。最后，我们设计了两个 Transformer 来进行最终的偏好预测。理论分析证明了 DEPS 的无偏性并分析了其方差。在三个公开数据集和一个工业数据集上的实验结果表明，DEPS 的表现显著优于最先进的基线方法。

CCS 概念

• **信息系统** →**推荐系统**。

*Jun Xu 为通讯作者。部分工作是在北京大数据管理与分析方法重点实验室完成的。

允许免费为个人或课堂用途制作本作品全部或部分内容的数字或纸质副本，前提是这些副本不得用于或分发以获取利润或商业利益，并且副本首页须附有本声明及完整引文。对于本作品中由 ACM 以外的其他方拥有版权的组件，必须尊重其版权。注明出处后可以摘录。若要以其他方式复制、转载、发布到服务器或分发至列表，须事先获得特别许可和/或支付费用。请通过 permissions@acm.org 申请许可。
CIKM '22，*2022*年*10*月*17*日至*21*日，美国佐治亚州亚特兰大© 2022 Association for Computing Machinery。ACM ISBN 978-1-4503-9236-5/22/10...$15.00 https://doi.org/10.1145/3511808.3557299

关键词

序列推荐，倾向得分估计

ACM 参考格式：
Chen Xu、Jun Xu、Xu Chen、Zhenghua Dong 和 Ji-Rong Wen。2022。序列推荐中的双重增强倾向得分估计。载于第*31*届*ACM*国际信息与知识管理会议（*CIKM '22*）论文集，*2022*年*10*月*17*日至*21*日，美国佐治亚州亚特兰大。ACM，纽约，纽约州，美国，11页。https://doi.org/10.1145/3511808.3557299

1 引言

序列推荐 [18, 31, 32] 受到业界和学术界越来越多的关注。序列推荐模型的主要优势在于能够显式建模物品之间的时间顺序相关性。为准确捕捉这类信息，近年来，研究者围绕马尔可夫链和循环神经网络开展了大量工作。尽管这些模型取得了显著成功，但由于曝光偏差或选择偏差 [3]，观测到的物品相关性可能会产生偏差。如图1(a)所示，给定一个用户行为序列，观测到的下一个物品是咖啡壶清洁剂。基于观测数据构建模型，可以学习清洁剂与咖啡壶之间的相关性。然而，从用户偏好的角度来看，下一个物品也可能是墨盒。但由于打印机和墨盒未被推荐并记录在数据中，模型没有机会捕捉它们之间的相关性。这种偏差会降低推荐效果，尤其是在测试环境与办公用品关联更密切时。

为了缓解上述问题，以往的模型大多基于逆倾向评分（IPS）技术 [30],：如果某个训练样本更有可能出现在数据集中，那么在优化过程中就应赋予它较低的权重。在这一研究方向中，关键在于根据历史信息 H，准确估计观测到用户-物品对（u，i）的概率，即 $P(u, i|H)$。为此，以往的方法通常将$P(u, i|H)$分解为 $P(i|u, H)P(u|H)$，并着重对 $P(i|u, H)$ 进行参数化（即从物品视角估计 $P(u, i|H)$），以预测给定此前交互过的物品后，用户接下来会与哪个物品交互。其原因之一是，从物品视角估计倾向得分与序列推荐的在线流程十分契合：用户随机进入系统，而系统旨在

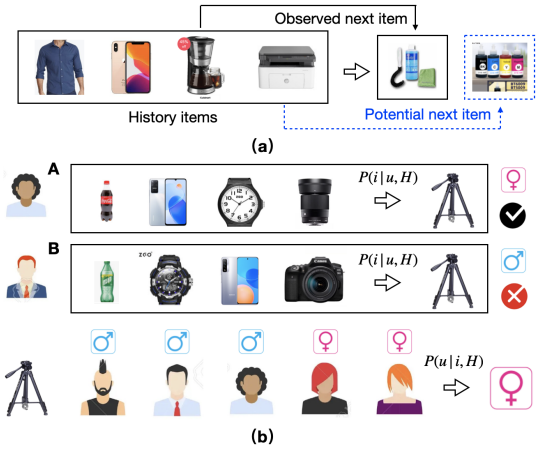


图1：(a) 无偏序列推荐示例。(b) 现有 IPS 方法中用户预测问题互补作用的玩具示例。

立即提供推荐物品。

尽管这些方法效果显著，我们认为，观测到用户-物品对的概率也可以从对偶视角来考量：对于某个物品，根据此前与其交互过的用户，预测下一位交互用户。从原理上说，这相当于以另一种方式将 $P(u, i|H)$ 分解为 $P(u|i, H)P(i|H)$，其中 $P(u|i, H)$ 的目标正是根据物品和历史用户预测用户（即从用户视角估计 $P(u, i|H)$）。直观而言，对于同一个物品，如果两位用户在短时间内与其交互，那么他们在该时间段内应有某些相似之处。因此，历史用户可以为预测下一位用户以及观测用户-物品对提供有用信号 [12]。我们认为，这种以用户为导向的方法能够为以往以物品为导向的模型提供互补信息。

例如，在图1(b)中，从物品预测视角来看，由于历史信息相似，三脚架可以被视为序列 A 和 B 的下一个物品。然而，从用户预测视角来看，我们可以推断序列 A 更有可能被观测到，因为近期女性用户与三脚架的交互更频繁，例如，这可能是由于妇女节促销等原因。这个例子表明，时间用户相关性信号可以很好地弥补传统的物品侧逆倾向评分方法，因此在对序列推荐系统进行去偏时应予以重视。

本文提出构建一种采用双重增强逆倾向评分估计（称为 DEPS）的无偏序列推荐模型。主要挑战体现在三个方面：首先，物品侧和用户侧逆倾向评分都很有用，那么如何从同一组用户反馈数据中估计它们？其次，如何将二者结合起来仍不明确，尤其是我们必须考虑时间顺序信息。最后，如何从理论上确保所提出的目标仍然无偏，也需要我们仔细设计。

为解决这些挑战，我们使用两个门控循环单元来估计倾向得分：一个从物品视图出发，另一个从用户视图出发。此外，为了使我们的模型 DEPS 具备实用性，我们使用两个 Transformer 来生成最终推荐：一个对目标用户的历史交互物品序列进行编码，另一个对与目标物品交互过的用户序列进行编码。编码后序列的嵌入，以及目标物品和目标用户的嵌入，将共同用于预测最终推荐得分。此外，我们设计了一个两阶段学习流程，用于估计倾向得分估计模块和物品推荐模块的参数。

本文的主要贡献总结如下：(1) 我们从两个视角强调了倾向评分估计的重要性，并提出了一种双重增强的逆倾向评分方法，用于对序列推荐模型进行去偏。

(2) 为实现上述设想，我们设计了双重门控循环单元架构，同时考虑用户侧和物品侧的逆倾向评分估计，并从理论上证明了目标函数的无偏性。

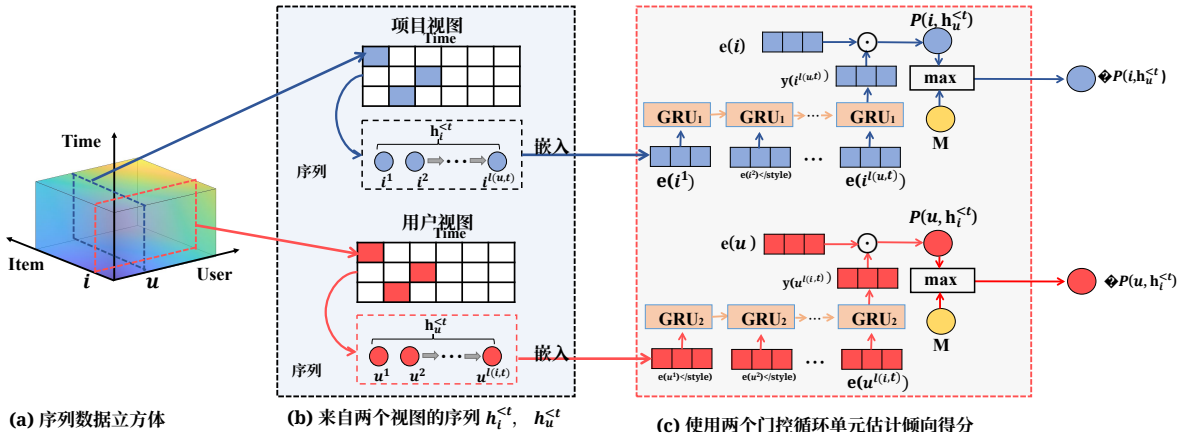
(3) 我们基于三个公开基准数据集和一个工业规模的商业数据集，通过与最先进的方法进行比较，开展了大量实验，以验证模型的有效性。

2 相关工作

已有大量研究致力于开发序列推荐模型 [7, 11, 18, 31, 32, 42, 43]。与传统推荐 [24, 36]相比，序列推荐旨在捕捉物品随时间变化的关联。研究者提出了基于马尔可夫链或循环神经网络的模型。例如，GRU4Rec+ [32] 引入了一种循环神经网络，将历史物品序列编码为用户偏好。BERT4Rec [31]提出了一种基于注意力的方法 [20]，用于有效建模用户行为序列。BST [7] 利用 Transformer [20] 从交互序列中捕捉用户偏好。LightSANs [11]将低秩分解的自注意力机制引入物品上下文建模。至于模型训练，S3-Rec [44] 结合了自监督学习，并采用预训练/微调范式。

现代推荐系统必须面对各种偏差，包括选择偏差 [19]，位置偏差 [9, 40]，流行度偏差 [38]，和曝光偏差 [1, 5, 6, 17, 28]。偏差通常会发生在多个方面 [1, 3]。例如，物品曝光既受用户先前行为 [17, 28] 的影响，也受用户背景 [4–6]的影响。Wang 等人 [34, 35]，和 Zhang 等人 [37] 指出，序列场景有所不同，还需要开展更多研究。

一种常见的偏差校正方法是采用逆倾向评分（IPS） [30]。Devooght 等人 [10]，Hu 等人 [14] 将先前经验作为倾向得分，对样本进行均匀重加权。UIR [28] 和 UBPR [27] 提出利用潜在概率模型来估计倾向得分。Agarwal 等人 [2]，Fang 等人 [13] 利用干预采集来学习倾向得分。Joachims 等人 [15]，Qin 等人 [23] 使用 EM 算法学习倾向模型。USR [34] 提出了一种从序列推荐中的物品视角估计倾向得分的网络。 [4, 6] 指出，估计倾向得分时仔细考虑用户视角很有帮助。



(a) 序列数据立方体 (b) 来自两个视图的序列 $h_i^{<t}$, $h_u^{<t}$ (c) 使用两个门控循环单元估计倾向得分

图 2：序列推荐中的倾向得分估计。(a) 将历史用户-物品交互表示为数据立方体，其中物品和用户可以按任意顺序放置在坐标轴上；(b) 分别从物品和用户的视角，构建与目标元组 (u, i, t) 对应的两个交互序列；(c) 使用两个门控循环单元估计倾向得分。

3 问题表述

3.1 序列推荐

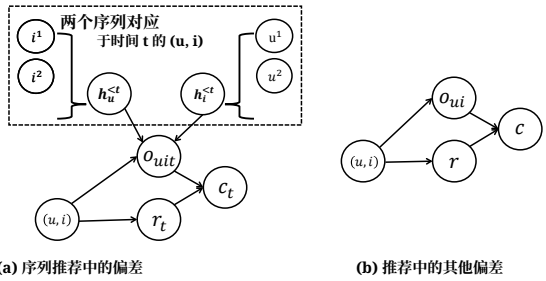
假设某个序列推荐系统管理一组用户-物品历史交互 $D = \{(u, i, c_t)\}$，其中每个元组 (u, i, c_t)记录了在时间戳 t，用户 $u \in U$ 访问系统并与物品 $i \in I$ 交互的情况，且用户反馈为$c_t \in \{0, 1\}$；其中 U 和 I 分别表示系统中的用户集合和物品集合，而 $c_t = 1$ 表示用户 u 点击了物品 i，若在时间 t未点击则为 0。此外，从系统收集的 (u, i) 的上下文信息（例如用户画像和物品属性）通常表示为实值向量（嵌入）$\mathbf{e}(u)$，$\mathbf{e}(i) \in \mathbb{R}^d$，其中 d表示嵌入的维度。

在特定时间 t，给定一个目标用户-物品对 (u, i)，可以从 D 中导出两种交互序列：（1）用户 u 在时间t 之前交互过的物品序列：$\mathbf{h}_u^{<t} = [i^1, i^2, \cdots i^{l(u,t)}],$，其中 $l(u, t)$ 表示用户 u 在时间 t之前交互过的物品数量；（2）在时间 t之前曾与物品 i 交互过的用户序列：$\mathbf{h}_i^{<t} = [u^1, u^2, \cdots u^{l(i,t)}],$，其中 $l(i, t)$ 表示与物品 i交互过的用户数量，时间早于 t。

图 2（a、b）说明，$\mathbf{h}_u^{<t}$和 $\mathbf{h}_i^{<t}$实际上是从 D 导出的数据立方体的双视图。具体而言，将用户、物品和时间视为三个轴，D 可表示为一个稀疏数据立方体：若 u在时间 t点击了 i，则 (u, i, t) 位置的元素为 1；若观察到但未点击，则为 0；若没有交互，则为 NULL。显然，$\mathbf{h}_i^{<t}$和$\mathbf{h}_u^{<t}$是数据立方体的两种视图：（1）项目视图：给定用户 u，其在 t之前的历史交互记录存储在按 u切片的矩阵中。由于 u一次只能与一个物品交互，因此可以移除未点击物品，按时间对剩余物品排序，从而得到列表 $\mathbf{h}_u^{<t}$，其中 $l(u, t)$ 是切片矩阵中非零元素的数量；（2）用户视图：给定一个物品

i及其交互历史，按 i 切片的矩阵也可以聚合成另一个列表$\mathbf{h}_i^{<t}$。

序列推荐的任务是基于数据集 D 中的用户-物品交互和用户反馈，学习一个函数 $\hat{r}_t = f(u, i, \mathbf{h}_u^{<t}, \mathbf{h}_i^{<t})$，用于预测用户 u 对物品 i 在



(a) 序列推荐中的偏差 (b) 推荐中的其他偏差

图3：序列推荐中的偏差因果图。

时间 t的偏好。我们期望预测偏好接近于时间 t 时不可观测的真实用户偏好 $r_t \in \{0, 1\}$，其中 $r_t = 1$ 表示用户 u在时间 t偏好 i，否则为 0。

3.2 序列推荐中的偏差

在序列推荐中，当用户 u 对某些物品的曝光系统性地不足或过度时，就会产生偏差。如图3(a)所示，从因果意义上说，只有当物品 i 与用户 u 相关，且用户 i接触到 u时，用户才会在时间t ($c_t = 1$)点击该物品，其中 $o_{uit} \in \{0,1\}$和 $r_t \in \{0,1\}$分别表示用户 u 是否知晓物品 i，以及该物品是否与用户 i相关，或形式化地表示为 $c_t = r_t \cdot o_{uit}$。进一步假设，两个交互序列$\mathbf{h}_u^{<t}$和$\mathbf{h}_i^{<t}$也会影响 u 是否知晓 i。由于模型根据观察到的点击 c_t来预测用户偏好 r_t，预测不可避免地会受到物品曝光 o_{uit}的偏差影响。这是因为在因果图中观察到点击后，o_{uit}会成为混杂因素（即点击是一个碰撞变量 [22]）。

形式上，用户 u在时间 t点击物品 i的概率，可以分解为 i对用户可见的概率（即，

$o_{uit}=1$）以及（u，i）相关的概率 $r_t=1$[1]：

$$P(c_t=1|u,i,t)=P(r_t=1|u,i,\mathbf{h}_u^{<t},\mathbf{h}_i^{<t})\cdot P(o_{uit}=1). \quad (1)$$

如图1所示，现有研究通常仅从物品视角估计$P(o_{uit}=1)$，而忽略了用户的双重视角。此外，估计还需要考虑时间顺序信息（即$\mathbf{h}_u^{<t}$和$\mathbf{h}_i^{<t}$）。

3.3 推荐的无偏目标函数

理想情况下，推荐（包括序列推荐和非序列推荐场景）的学习目标应基于真实偏好与推荐模型预测分数之间的相关性构建：

$$\mathcal{L}_n^{\text{ideal}}=\sum_{u\in\mathcal{U}}\sum_{i\in\mathcal{I}}\delta(r,\hat{r}(u,i)), \quad (2)$$

其中，$\hat{r}(u,i)$是推荐模型的预测值，r是真实偏好，$\delta(\cdot,\cdot)$是定义在每个用户-物品对上的损失函数。请注意，在非序列推荐中，不考虑时间t和历史信息$\mathbf{h}_u^{<t}$，$\mathbf{h}_i^{<t}$。然而，在现实世界中，无法直接优化 Lideal$_n$，因为真实偏好 r 无法被观测到。

传统推荐系统将观察到的点击 c 视为模型学习的标签，因此不可避免地会受到曝光偏差或自选择偏差的影响，如图3(b)所示。一种实用的解决方案是通过混杂因素 [3]的倾向得分来纠正这些偏差。在非序列场景下，一种典型方法是利用倾向得分 $P(o=1)$为每个观察到的交互加权：

$$\mathcal{L}_n^{\text{unbiased}}=\sum_{u\in\mathcal{U}}\sum_{i\in\mathcal{I}}\left[\mathbb{I}(o_{ui}=1)\frac{\delta(c,\hat{r}(u,i))}{P(o_{ui}=1)}\right], \quad (3)$$

其中 $\mathbb{I}(\cdot)$ 是指示函数，且公式 (3) 是理想目标公式 (2) 的无偏估计，即 $\mathbb{E}_o\left[\text{L}_{\text{无偏}_n}\right]=\text{L}_{\text{理想}_n}$。更多详情请参阅 [3]。

在这一框架下，已有无偏推荐模型。将这些方法推广到序列推荐并非易事。本文提出了一种利用用户—物品交互序列估计倾向得分的方法框架，称为 DEPS。

4 我们的方法：DEPS

在本节中，我们为序列推荐提出了一个无偏目标函数。随后，我们开发了一种名为 DEPS 的双重增强逆倾向评分估计模型，用于估计序列推荐目标函数中的倾向得分。最后，我们提出了两种 Transformer，以便在实际应用中调整我们的框架。

4.1 序列推荐的无偏损失

给定用户-物品历史交互数据集 D，我们将序列推荐的理想学习目标定义为：评估用户在每个时间点访问系统时的偏好：

$$\mathcal{L}_s^{\text{ideal}}=\sum_{u\in\mathcal{U}}\sum_{t:(u,i',c_t)\in\mathcal{D}}\sum_{i\in\mathcal{I}}\delta(r_t,\hat{r}_t(u,i,\mathbf{h}_i^{<t},\mathbf{h}_u^{<t})), \quad (4)$$

其中，$\hat{r}(u,i,\mathbf{h}_i^{<t},\mathbf{h}_u^{<t})$ 是序列推荐模型的预测结果，而 r_t 是时间 t 时真实但不可观测的偏好。不同于非序列的无偏推荐模型，序列推荐中的倾向得分与时间相关，如图3(a)所示的因果图所示。实现无偏序列推荐学习目标的一种方法，是基于$\mathbf{h}_i^{<t}$和$\mathbf{h}_u^{<t}$的历史交互序列，估计时间 t 时与 (u,i) 对应的倾向得分 $P(o_{uit}=1)$，如下述定理所示。

定理1（时间感知无偏学习目标）。给定用户-物品交互D $=\{(u,\ i,\ c_t)\}$，我们有

$$\mathbb{E}_o\left[\mathcal{L}_s^{\text{unbiased}}\right]=\mathbb{E}_o\left[\alpha\mathcal{L}_u+(1-\alpha)\mathcal{L}_i\right]=\mathcal{L}_s^{\text{ideal}}, \quad (5)$$

其中，$\alpha\in[0,1]$ 是用于平衡两个目标函数的系数：

$$\mathcal{L}_u=\sum_{(u,i,c_t)\in\mathcal{D}}\left[\frac{\delta(c_t,\hat{r}_t)}{P(i,\mathbf{h}_u^{<t})}\right]=\sum_{u\in\mathcal{U}}\sum_{(i,c_t)\in\mathcal{D}^u}\frac{\delta(c_t,\hat{r}_t)}{P(i,\mathbf{h}_u^{<t})},$$

$$\mathcal{L}_i=\sum_{(u,i,c_t)\in\mathcal{D}}\left[\frac{\delta(c_t,\hat{r}_t)}{P(u,\mathbf{h}_i^{<t})}\right]=\sum_{i\in\mathcal{I}}\sum_{(u,c_t)\in\mathcal{D}^i}\frac{\delta(c_t,\hat{r}_t)}{P(u,\mathbf{h}_i^{<t})},$$

其中，$D^u=\{(i,c_t):(u,i,c_t)\in D\}$，$D^i=\{(u,c_t):(u,i,c_t)\in D\}$，$P(i,h_u^{<t})$ 是 i 和 $h_i^{<t}$ 出现的概率，而 $P(u,h_i^{<t})$ 是 u 和 $h_i^{<t}$ 出现的概率。

定理1的证明见附录A.1。从该定理可以看出，序列推荐的无偏学习目标函数既可以从用户视角 L$_u$ 获得，也可以从物品视角 L$_i$ 获得。此外，很容易知道，两个无偏损失的平均值，即式（5）中定义的 Lunbiased$_s$，仍然是一个无偏目标函数。

考虑到倾向得分在 L$_i$ 和 L$_u$ 中作为分母。为提高估计的稳定性，采用裁剪技术 [29] 对式（8）和式（9）中的估计概率进行处理，得到

$$\widetilde{P}(i,\mathbf{h}_u^{<t})=\max\{P(i,\mathbf{h}_u^{<t}),M\}, \quad (6)$$

$$\widetilde{P}(u,\mathbf{h}_i^{<t})=\max\{P(u,\mathbf{h}_i^{<t}),M\}, \quad (7)$$

其中 $M\in(0,1)$ 为裁剪值。我们证明，使用裁剪后的倾向得分可以对估计方差进行界定：

定理2（估计方差）。设 $L_u^t=\frac{\delta(c_t,\hat{r}_t)}{\widetilde{P}(u,h_i^{<t})}$ 和 $L_i^t=\frac{\delta(c_t,\hat{r}_t)}{\widetilde{P}(i,h_u^{<t})}$ 是关于单个训练样本 $(i,u,c_t)\in D$ 上损失的两个随机变量，则 $\alpha L_u^t+(1-\alpha)L_i^t$ 的估计方差满足：

$$\mathbb{V}\left[\alpha L_u^t+(1-\alpha)L_i^t\right]\le\max\{\mathbb{V}\left[L_u^t\right],\mathbb{V}\left[L_i^t\right]\}\le\left(\frac{1}{M}-1\right)\delta^2(r_t,\hat{r}_t).$$

定理2的证明见附录 A.2。我们得出结论，平均损失不会带来额外的方差。直观而言，裁剪值 M 是在无偏性与方差之间的权衡，并提供了一种控制方差的机制。较大的 M 会带来较低的方差和更大的偏差。我们表明，Saito 等人 [29] 提出的裁剪技术在对偶视角下仍然有效。

[1]我们假设观测概率 $P(o=1)$ 仅取决于用户和物品的历史交互序列。在实际任务中，物品的可观测概率还会受到其他因素的影响，例如排序位置，如图3(b)所示。将这些因素纳入序列推荐是未来的研究方向。

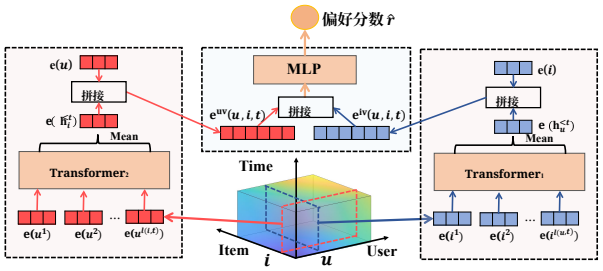


图4：基于来自用户和物品的序列构建的双 Transformer 序列推荐模型。

上述分析为学习无偏序列推荐模型提供了一种优雅且理论上可靠的方法，包含三个步骤：(1) 对每个元组 $(u, i, \mathbf{h}_u^{<t}, \mathbf{h}_i^{<t})$，估计两个倾向得分 $\widetilde{P}(i, \mathbf{h}_u^{<t})$ 和 $\widetilde{P}(u, \mathbf{h}_i^{<t})$；(2) 开发一个序列推荐系统模型；(3) 训练参数，其中包括最小化平均损失 $\alpha \mathrm{L}_i + (1-\alpha)\mathrm{L}_u$。接下来，我们将在第 4.2 节展示使用两个门控循环单元实现步骤 (1) 的方法。第 4.3 节和第 4.4 节分别实现步骤 (2) 和步骤 (3)。

4.2 使用门控循环单元估计倾向得分

在现实世界中，与物品和用户的全集相比，序列 $\mathbf{h}_u^{<t}$ 和 $\mathbf{h}_i^{<t}$ 非常稀疏且较短。本文采用门控循环单元（GRU）神经语言模型 [8] 来估计倾向得分。此外，根据定理2，我们对估计的倾向得分应用裁剪技术。

具体而言，分别使用两个门控循环单元来估计 $P(i, \mathbf{h}_u^{<t})$ 和 $P(u, \mathbf{h}_i^{<t})$，如图2(c)所示。具体来说，给定一个元组 $(u, i, \mathbf{h}_u^{<t}, \mathbf{h}_i^{<t})$，从物品视图得到的倾向得分估计为 M 和 $P(i, \mathbf{h}_u^{<t})$ 中的最大值，其中 $P(i, \mathbf{h}_u^{<t})$ 与 i 的嵌入 $\mathbf{e}(i)$ 及一个门控循环单元的输出的点积成正比，该单元以序列 $\mathbf{h}_u^{<t} = [i^1, i^2, \cdots\ i^{l(u,t)}]$ 为输入。根据公式(6)应用裁剪技术后，我们将从物品视图得到的估计倾向得分写为：

$$\widetilde{P}(i, \mathbf{h}_u^{<t}) = \max\left\{\frac{\exp\left(\mathbf{e}(i)^T \mathbf{y}(i^{l(u,t)})\right)}{\sum_{i' \in \mathcal{I}} \exp\left(\mathbf{e}(i')^T \mathbf{y}(i^{l(u,t)})\right)}, M\right\}, \quad (8)$$

其中，$\mathbf{y}(i^{l(u,t)}) \in \mathbb{R}^d$ 是 GRU 最后一层的输出（即对应于第 $l(u,t)$ 个输入）。GRU 按如下方式扫描 $\mathbf{h}_u^{<t}$ 中的物品：在第 $k(k = 1, \cdots\ l(u,t))$ 层，它将第 k 个物品的嵌入 $\mathbf{e}(i^k)$ 作为输入，并输出 $\mathbf{y}(i^k)$，作为已扫描子序列 $[i^1, i^2, \cdots\ i^k]$ 的表示：

$$\mathbf{y}(i^k), \mathbf{z}^k = \mathbf{GRU}_1(\mathbf{e}(i^k), \mathbf{z}^{k-1}),$$

其中，$\mathbf{z}^k$ 和 $\mathbf{z}^{k-1}$ 分别是第 k 步和第 $(k-1)$ 步的隐藏向量，而 $\mathbf{GRU}_1$ 是从项目视图处理该序列的门控循环单元。

同样，给定一个元组 $(u, i, \mathbf{h}_u^{<t}, \mathbf{h}_i^{<t})$，也可以从用户视角估计其倾向得分：取 $P(u, \mathbf{h}_i^{<t})$ 与 M 中的最大值，其中 $P(u, \mathbf{h}_i^{<t})$ 与用户嵌入 $\mathbf{e}(u)$ 和序列 $\mathbf{h}_i^{<t}$ 的表示的点积成正比 $[u^1, u^2, \cdots\ u^{l(i,t)}]$。通过应用公式（7）中的裁剪技术，我们将从用户视角估计的倾向得分写为：

$$\widetilde{P}(u, \mathbf{h}_i^{<t}) = \max\left\{\frac{\exp\left(\mathbf{e}(u)^T \mathbf{y}(u^{l(i,t)})\right)}{\sum_{u' \in \mathcal{U}} \exp\left(\mathbf{e}(u')^T \mathbf{y}(u^{l(i,t)})\right)}, M\right\}, \quad (9)$$

其中，$\mathbf{y}(i^{l(u,t)}) \in \mathbb{R}^d$ 是另一个 GRU 的输出，该 GRU 按如下方式扫描 $\mathbf{h}_i^{<t}$：在第 k 层，它将第 k 个用户的嵌入 $\mathbf{e}(u^k)$ 作为输入，并输出 $\mathbf{y}(u^k)$，作为已扫描子序列 $[u^1, u^2, \cdots\ u^k]$ 的表示：

$$\mathbf{y}(u^k), \mathbf{z}^k = \mathbf{GRU}_2(\mathbf{e}(u^k), \mathbf{z}^{k-1}),$$

其中 $\mathbf{z}^k$ 和 $\mathbf{z}^{k-1}$ 是隐藏向量，而**门控循环单元**$_2$ 是另一个门控循环单元，用于从用户视角处理交互序列。

4.3 骨干网络：基于 Transformer 的推荐器

如图 4 所示，该序列推荐模型由 Transformer 层和预测层组成。Transformer 层包含两个 Transformer [20]。其中一个将序列 $\mathbf{h}_u^{<t}$ 和目标物品 i 转换为表示向量，另一个则将序列 $\mathbf{h}_i^{<t}$ 和目标用户 u 转换为另一个表示向量。预测层将这些向量拼接起来，并使用多层感知机进行预测。

4.3.1 Transformer 层。 输入元组（u, i, t）的整体物品表示可以表示为物品 ID 嵌入与用户历史序列嵌入的拼接：

$$\mathbf{e}^{\mathrm{iv}}(u, i, t) = \left[\mathbf{e}(\mathbf{h}_u^{<t}) \| \mathbf{e}(i)\right], \quad (10)$$

其中，运算符'‖'用于拼接两个向量，$\mathbf{e}(i)$ 是物品的嵌入，$\mathbf{h}_u^{<t} = [i^1, i^2, \cdots\ i^{l(u,\ t)}]$ 是与目标用户 u 相关的用户序列，$\mathbf{e}(\mathbf{h}_u^{<t})$ 是对该序列进行编码的向量，定义为 Transformer 输出向量的均值：

$$\mathbf{e}(\mathbf{h}_u^{<t}) = \mathrm{Mean}\left(\mathbf{Transformer}_1\left([\mathbf{e}(i^1), \cdots, \mathbf{e}(i^{l(u,t)})]\right)\right), \quad (11)$$

其中，“Mean”表示对所有输入向量执行的均值池化操作，而 $\mathbf{Transformer}_1(\cdot)$ 是一种 Transformer [20] 架构。

类似地，输入元组（u, i, t）的整体物品表示可表示为用户 ID 嵌入与物品历史序列嵌入的拼接：

$$\mathbf{e}^{\mathrm{uv}}(u, i, t) = \left[\mathbf{e}(\mathbf{h}_i^{<t}) \| \mathbf{e}(u)\right], \quad (12)$$

$\mathbf{e}(u)$ 是目标用户 i 的嵌入，$\mathbf{h}_i^{<t} = [u^1, u^2, \cdots\ u^{l(i,t)}]$ 是与目标物品 i 交互的物品序列，而 $\mathbf{e}(\mathbf{h}_i^{<t})$ 是另一个 Transformer 输出的均值：

$$\mathbf{e}(\mathbf{h}_i^{<t}) = \mathrm{Mean}\left(\mathbf{Transformer}_2\left([\mathbf{e}(u^1), \cdots, \mathbf{e}(u^{l(i,t)})]\right)\right), \quad (13)$$

其中 $\mathbf{Transformer}_2(\cdot)$ 是另一种 Transformer 架构。

4.3.2 预测层。最后，使用多层感知机（MLP），将 $\mathbf{eiv}(u, i, t)$ 和 $\mathbf{euv}(u, i, t)$ 作为输入，并输出预测偏好 $\hat{r}$：

$$\hat{r} = \sigma(\mathbf{MLP}(\mathbf{e}^{\mathrm{iv}}(u, i, t) \| \mathbf{e}^{\mathrm{uv}}(u, i, t))), \quad (14)$$

其中“σ”表示 sigmoid 函数运算，**多层感知机** 是一个两层全连接神经网络，同时接收 $\mathbf{eiv}(u,\ i,\ t)$ 和

算法1： DEPS 学习算法

输入： 训练集 $\mathrm{D}=\{(u,\ i,\ c_t)\}$，迭代次数 n_p，n_u，n_b，系数 λ p

输出： $\Theta=\{\theta_e,\ \theta_p,\ \theta_t,\ \theta_m\}$

1: $\Theta \leftarrow$ 随机值

2: $\mathcal{H} \leftarrow \{\mathbf{h}_u^{<t}|u\in \mathrm{U}\}\cup\{\mathbf{h}_i^{<t}|i\in \mathrm{I}\}$ {从 D 中提取序列}

3: **对** $n=1,\ \cdots n_p$ **执行**

4: 通过最小化 $\theta_p\ \mathcal{L}\mathrm{AR}_u\ +\mathcal{L}\mathrm{AR}i$进行更新

5: 更新 $\theta_e,\ \theta_t$通过最小化 $\lambda_p(\mathrm{LMLM}_u\ +\mathrm{LMLM}i)$

6: **循环结束**

7: **对于** $n=1,\cdots\ n_u$，**执行**

8: 从 (u,i,t) 中提取序列 $\mathbf{h}_u^{<t},\mathbf{h}_i^{<t}$

9: **对于** $k=1,\cdots\ n_b$，**执行**

10: 通过最小化损失 $\mathrm{LAR}_u\ +\mathrm{LAR}i$ 来更新 θ_p。

11: **结束循环**。

12: 计算偏好分数 $\hat{r}=f(u,i,\mathbf{h}_u^{<t},\mathbf{h}_i^{<t})$（式（14））

13: 计算倾向得分 $\widetilde{P}(i,\ \mathbf{h}_u^{<t})$ $\widetilde{P}(u,\ \mathbf{h}_i^{<t})$（式（6）、（7））

14: 通过最小化损失 $\mathrm{Lunbiased}_s$，更新 $\theta_e,\ \theta_t,\ \theta_m$。

15: **结束循环**

将 $\mathrm{euv}(u,\ i,\ t)$ 作为输入。

4.4 使用估计的倾向得分进行学习

所提出的模型包含一组待学习的参数，记为$\Theta=\{\theta_e,\ \theta_p,\ \theta_t,\ \theta_m\}$，其中，参数 θ_e 表示嵌入模型中的参数，该模型输出用户和物品嵌入；参数 θ_p 表示**门控循环单元**1 和**门控循环单元**2 中用于估计倾向得分的参数；参数 θ_t 表示**Transformer**1和**Transformer**2中的参数；参数 θ_m 表示**多层感知机** 中用于生成最终推荐的参数。

受先预训练再微调范式的启发，我们也设计了一个两阶段学习流程来学习模型参数。第一阶段，以无监督学习的方式训练 $\{\theta_e,\ \theta_p,\ \theta_t\}$ 的参数，从而获得较好的初始化。随后，第二阶段使用上述无偏学习目标学习所有参数。使用 Adam [16] 优化器进行优化。

对于阶段1，我们使用n_p 个周期来优化 Lstage-1。对于阶段2，我们使用 n_u 个周期交替训练，其中每个周期分别使用 n_b 个周期来优化 $\mathrm{LAR}_u\ +\mathrm{LAR}i$，并使用1个周期来优化 $\mathrm{Lunbiased}_s$。整体算法流程见算法1。

4.4.1 第一阶段：无监督学习。在第一阶段，所有用户-系统交互序列的两个视图，即所有 $u\in \mathrm{U}$ 和 $i\in \mathrm{I}$ 对应的$\mathbf{h}_i^{<t}$s 和 $\mathbf{h}_u^{<t}$s，均用作无监督训练实例。受自回归语言模型和掩码语言模型成功经验的启发，我们设计了两种学习任务，分别将这两种语言模型应用于用户序列和物品序列，从而得到由四部分组成的总损失 Lstage-1：

$$\mathcal{L}^{\text{stage-1}}=(\mathcal{L}_u^{\mathrm{AR}}+\mathcal{L}_i^{\mathrm{AR}})+\lambda_p(\mathcal{L}_u^{\mathrm{MLM}}+\mathcal{L}_i^{\mathrm{MLM}}),\qquad(15)$$

其中 $\lambda_p>0$ 为权衡系数，$\mathrm{LAR}u$ 和 $\mathrm{LAR}i$ 分别表示将自回归语言模型应用于 $\mathrm{LAR}u$ 和 L_i^{AR}序列时对应的损失。具体而言，$\mathrm{LAR}u$ 定义为

定义如下：

$$\mathcal{L}_u^{\mathrm{AR}}=\sum_{u\in\mathcal{U}}\mathrm{AR}(\mathbf{h}_u^{<t})=\sum_{u\in\mathcal{U}}\sum_{m=1}^{l(u,t)}-\log P\left(i^m|\left[i^1,\cdots,i^{m-1}\right]\right),$$

其中，i^m是序列$\mathbf{h}_u^{<t}$中的第 m个物品，$\left[i^1,\cdots\ i^{m-1}\right]$是$\mathbf{h}_u^{<t}$的长度为（$m-1$）的前缀，概率 $P(\cdot|\cdot)$根据公式（8）计算。类似地，L_i^{AR}定义为：

$$\mathcal{L}_i^{\mathrm{AR}}=\sum_{i\in\mathcal{I}}\mathrm{AR}(\mathbf{h}_i^{<t})=\sum_{i\in\mathcal{I}}\sum_{m=1}^{l(i,t)}-\log P\left(u^m|\left[u^1,\cdots,u^{m-1}\right]\right),$$

概率 $P(\cdot|\cdot)$ 根据公式（9）计算。

对于 $\mathrm{LMLM}u$ 和 $\mathrm{LMLM}i$ ，我们遵循 BERT4Rec [31]，中的做法，分别将掩码语言模型应用于序列$\mathbf{h}_u^{<t}$和$\mathbf{h}_i^{<t}$，得到：

$$\mathcal{L}_u^{\mathrm{MLM}}=\sum_{u\in\mathcal{U}}\mathrm{MLM}\left(\mathbf{h}_u^{<t}\right);\mathcal{L}_i^{\mathrm{MLM}}=\sum_{i\in\mathcal{I}}\mathrm{MLM}\left(\mathbf{h}_i^{<t}\right),$$

其中，$\mathrm{MLM}(\cdot)$ 用于计算输入序列上的掩码语言模型损失。MLM 任务将使我们的第二阶段训练更加稳定。

4.4.2 第二阶段：无偏学习。在第二阶段训练中，给定 $\mathrm{D}=\{(u,\ i,\ c_t)\}$，使用估计的倾向得分对原始有偏损失 $\delta(c,\ \hat{r})$ 进行重新加权，从而得到无偏损失 Lstage-2：

$$\mathcal{L}^{\text{stage-2}}=\mathcal{L}_s^{\text{unbiased}},\qquad(16)$$

其中，无偏目标函数 $\mathrm{Lunbiased}_s$ 基于定理1中的公式（5）构建，方法是将公式（6）和公式（7）中的估计倾向得分代入公式（5）。

请注意，为避免交替训练时倾向得分估计的方差过大，第二阶段在实际中也会使用 L_u^{AR}和 L_i^{AR}。本文所有实验中，原始损失 $\delta(c,\ \hat{r})$ 均设为二元交叉熵：

$$\delta(c,\hat{r})=c\cdot\log(\hat{r})+(1-c)\log(1-\hat{r}),$$

其中，$\hat{r}$ 由式（14）预测。

5 实验

我们进行了实验，以验证 DEPS 的有效性。[2]

5.1 实验设置

实验在四个大规模公开序列推荐基准数据集上进行：

MIND[3]：一个大规模新闻推荐数据集。为避免出现极度稀疏的情况，剔除了交互物品少于 5 个的用户，以及被交互用户少于 5 个的物品。

亚马逊美妆/Amazon-Digital-Music：亚马逊商品数据集[4]中的两个子集（美妆和数字音乐领域）。同样，我们移除了交互物品/用户少于 5 个的用户/物品。我们将用户在亚马逊数据集中给出的 4–5 星评分视为正反馈（标记为 1），其他评分视为负反馈（标记为 0）。

[2]源代码已共享于 https://github.com/XuChen0427/Dually-Enhanced-Propensity-Score-Estimation-in-Sequential-Recommendation。[3] https://msnews.github.io/ [4]http://jmcauley.ucsd.edu/data/亚马逊/

表 1：数据集统计信息。

数据集	#用户	#物品	#交互	稀疏度
MIND	13863	2464	59228	99.82%
亚马逊美妆	24411	32371	94641	99.98%
Amazon-Digital-Music	4424	5365	32314	99.86%
华为	1997	17490	245564	99.29%

华为数据集：为验证我们的方法在生产数据上的有效性，我们从华为音乐服务系统收集了 1 个月的流量日志，抽样后约有 245K 次交互。

表 1 列出了四个数据集的统计信息。遵循 [25, 31], 去偏推荐模型需要基于无偏测试集进行评估 [29]中的做法。按照 [26],的做法，我们使用按交互时间排序的前 50% 交互进行训练，并对其余 50% 的数据重新采样，用于评估和测试。具体而言，假设物品 i被点击了 m_i次，我们使用逆概率 $m_i/\max_{j\in\mathcal{I}} m_j$进行采样。随后，我们分别使用排序数据中的 20% 和 30% 作为验证集和测试集。

我们选择以下具有代表性的序列推荐模型作为基线方法：**STAMP** [18]，用于建模用户的长期和短期偏好；**GRU4Rec +** [32] 是 GRU4Rec 的改进版本，采用了数据增强并考虑输入中的变化；**BERT4Rec** [31] 采用注意力模块对用户行为进行建模，并以无监督方式训练；**FPMC** [25] 通过结合矩阵分解与一阶马尔可夫链来捕捉用户偏好；**DIN** [43] 应用注意力模块，从用户的历史行为中自适应地学习用户兴趣；**BST** [7] 应用 Transformer 架构，根据用户的历史行为以及用户和物品的辅助信息，自适应地学习用户兴趣；**LightSANs** [11] 是一种基于低秩分解 SANs 的推荐模型。我们还选择以下无偏推荐模型作为基线方法：**UIR** [29] 是一种使用启发式方法估计倾向得分的无偏推荐模型；**CPR** [33] 是一种针对曝光偏差的成对去偏方法；**UBPR** [27]是一种针对非负成对损失的逆倾向评分方法。**DICE** [41]：一种专注于用户群体的去偏模型。**USR** [34]：一种旨在减轻潜在混杂因素所致偏差的序列去偏模型。

为评估 DEPS 和基线方法的性能，我们采用了两类指标：推荐准确率 [25, 31] 在 NDCG@K 和 HR@K 方面。遵循 [31, 43]，中表示序列中的用户和物品的做法（即，用户表示为$\mathbf{h}_i^{<t}$，物品表示为 $\mathbf{h}_u^{<t}$），在原始嵌入中加入了序列位置嵌入。

对于所有模型的超参数，学习率在 $[1e-3, 1e-4]$ 中进行调优，倾向得分估计的截断系数 M 在 $[0.01, 0.2]$ 中进行调优。第一阶段 λ_p 的权衡系数设为 0.5。两个视图的权衡系数 α 在 $[0.4, 0.6]$ 中进行调优。神经网络的隐藏维度 d 在 $\{64, 128, 256\}$ 中进行调优，丢弃率在 $\{0.1, 0.2, 0.3, 0.4, 0.5\}$ 中进行调优，Transformer 层数在 $\{2, 3, 4\}$ 中进行调优。

所有基线方法和实验均使用 RecBole 推荐工具 [39] 和 PyTorch [21] 开发并完成。所有模型均在单张 NVIDIA GeForce RTX 3090 上训练，批次大小在 {1024，2048，4096} 中调优。

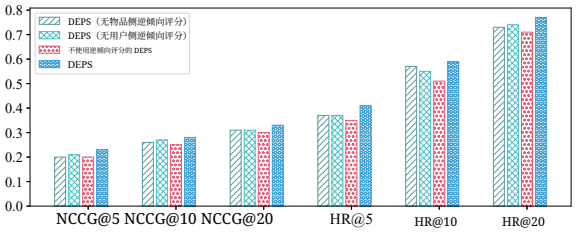


图 5：基于 Amazon 数字音乐的实证分析：DEPS 在 NDCG@K 和 HR@K 上的变化表现

5.2 实验结果

表 2 报告了 DEPS 与基线方法在全部四个数据集上的实验结果，采用 NDCG@K 和 HR@K 衡量推荐准确率。“$*$”表示相较于最佳基线方法的提升具有统计显著性（t 检验和 p 值 < 0.05）。下划线标示表现最佳的方法。

从报告的结果可以看出，除华为商业数据上的 NDCG@5 外，DEPS 在 NDCG 和 HR 方面均显著优于几乎所有基线方法，这验证了 DEPS 在提升序列推荐准确率方面的有效性。此外，DEPS 显著优于无偏模型，证明了从物品和用户序列推荐的角度估计倾向得分的重要性。

5.3 实验分析

我们基于 Amazon-Digital-Music 测试数据开展了更多实验，以分析 DEPS。

5.3.1 消融研究。 为了进一步说明从用户视图和物品视图出发，利用两种类型的序列估计倾向得分的重要性，我们还研究了在训练第二阶段优化 $\mathrm{L}^{unbiased}$ 时这些方法的无偏性能。具体而言，我们展示了若干种 DEPS 变体的 NDCG@K 和 HR@K。这些变体包括：不进行倾向得分估计来训练推荐模型（记为“w/o IPS”）；仅利用项目视图的物品序列估计倾向得分（“w/o user-oriented IPS”）；以及仅利用用户视图的用户序列估计倾向得分（“w/o item-oriented IPS”）。从图 5 所示的性能可以看出：(1) “w/o propensity”的表现最差，说明倾向得分对于无偏序列推荐至关重要；(2) “w/o user-oriented IPS”和“w/o item-oriented IPS”的表现明显更好，说明从任一视图估计的倾向得分都有效；(3) 使用双重倾向得分的 DEPS 表现最佳，验证了 DEPS 同时利用两个视图进行倾向得分估计的有效性。

双重变换器依赖若干重要机制来估计倾向得分和学习模型参数，包括利用$\mathbf{h}_u^{<t}$和$\mathbf{h}_u^{<t}$的交互序列进行估计，以及使用第一阶段无监督学习来初始化参数。基于 Amazon-Digital-Music

表 2：DEPS 与基线方法在 MIND、Beauty、Music 和华为数据集上的性能比较。'*'表示相较于最佳基线方法（下划线标出的数值）的提升具有统计显著性（t 检验，且 p-值 < 0.05）。

		序列推荐基线方法							无偏推荐基线方法					我们的方法	
数据集	指标	STAMP	DIN	BERT4Rec	FPMC	GRU4Rec+	BST	LightSANs	UIR	CPR	UBPR	DICE	USR	**DEPS**	提升
MIND	NDCG@5	0.0471	0.1149	0.0900	0.0670	0.0865	0.0865	0.1148	0.0594	0.0582	0.0588	0.0612	0.0658	**0.1197***	4.2%
	NDCG@10	0.0669	0.1548	0.1277	0.1006	0.1306	0.1233	0.1650	0.0823	0.0847	0.0863	0.0861	0.0955	**0.1728***	4.7%
	NDCG@20	0.0997	0.1948	0.1817	0.1400	0.1819	0.1753	0.2159	0.1233	0.1204	0.1237	0.1235	0.1339	**0.2249***	4.2%
	HR@5	0.0861	0.2090	0.1671	0.1229	0.1504	0.1607	0.2024	0.1141	0.1037	0.1048	0.1201	0.1207	**0.2200***	8.7%
	HR@10	0.1519	0.3379	0.2922	0.2359	0.2918	0.2825	0.3692	0.1909	0.1898	0.1942	0.2030	0,2194	**0.3961***	7.3%
	HR@20	0.2863	0.5019	0.5145	0.3999	0.5052	0.4948	0.5760	0.3577	0.3379	0.3511	0.3571	0.3807	**0.6078***	5.5%
亚马逊-Beauty	NDCG@5	0.0985	0.1139	0.1008	0.1225	0.1050	0.1156	0.1312	0.1188	0.1172	0.0918	0.1238	0.1046	**0.1362***	3.8%
	NDCG@10	0.1330	0.1407	0.1333	0.1563	0.1438	0.1548	0.1661	0.1603	0.1465	0.1214	0.1645	0.1429	**0.1830***	10.2%
	NDCG@20	0.1780	0.1699	0.1722	0.1962	0.1919	0.1983	0.2078	0.2109	0.1823	0.1580	0.2127	0.1903	**0.2302***	8.2%
	HR@5	0.1945	0.2160	0.1964	0.2292	0.2132	0.2134	0.2430	0.2338	0.2119	0.1780	0.2365	0.2064	**0.2557***	8.1%
	HR@10	0.3349	0.3174	0.3166	0.3530	0.3555	0.3679	0.3696	0.3836	0.3176	0.2879	0.3834	0.3435	**0.4175***	8.8%
	HR@20	0.5364	0.4434	0.4864	0.5233	0.5656	0.5524	0.5424	0.5980	0.4756	0.4465	0.5890	0.5470	**0.6130***	2.5%
亚马逊-数字-Music	NDCG@5	0.0727	0.1220	0.1419	0.1355	0.0651	0.1556	0.1940	0.0691	0.1664	0.1353	0.1170	0.0921	**0.2256***	16.3%
	NDCG@10	0.0961	0.1513	0.1771	0.1799	0.0961	0.1976	0.2192	0.0912	0.2121	0.1643	0.1432	0.1222	**0.2766***	30.4%
	NDCG@20	0.1305	0.1890	0.2231	0.2153	0.1385	0.2454	0.2586	0.1317	0.2558	0.2060	0.1789	0.1643	**0.3273***	26.6%
	HR@5	0.1373	0.2745	0.2606	0.2941	0.1709	0.3249	0.3613	0.1541	0.3473	0.2521	0.2185	0.2017	**0.4093***	13.3%
	HR@10	0.2437	0.3950	0.4249	0.4391	0.3025	0.4902	0.4737	0.2437	0.5014	0.3754	0.3361	0.3305	**0.5852***	16.7%
	HR@20	0.4034	0.5518	0.5949	0.6091	0.4874	0.6779	0.6162	0.4314	0.6415	0.5546	0.4902	0.5098	**0.7757***	14.4%
华为	NDCG@5	0.0919	0.1081	0.1247	0.1079	0.1090	0.1016	0.1335	0.0554	0.1414	0.1033	0.0670	0.0922	0.1400	**-1.0%**
	NDCG@10	0.1024	0.1195	0.1352	0.1193	0.1163	0.1125	0.1323	0.0634	0.1489	0.1192	0.0767	0.1042	**0.1503***	1.0%
	NDCG@20	0.1202	0.1421	0.1561	0.1416	0.1357	0.1344	0.1454	0.0793	0.1711	0.1409	0.0926	0.1234	**0.1755***	2.5%
	HR@5	0.3569	0.3874	0.4192	0.3954	0.3941	0.3594	0.4576	0.2129	0.4726	0.3880	0.2453	0.3435	**0.4765***	0.9%
	HR@10	0.5540	0.6138	0.6242	0.6174	0.5833	0.5625	0.6315	0.3569	0.6730	0.6132	0.4155	0.5552	**0.6852***	1.8%
	HR@20	0.7404	0.8011	0.8090	0.8188	0.7890	0.7785	0.7889	0.5625	0.8603	0.8115	0.6107	0.7529	**0.8713***	1.3%

表3：在 Amazon-Digital-Music 测试集上的消融研究。

指标	NDCG@K			HR@K		
K	5	10	20	5	10	20
不含$\mathbf{h}_u^{<t}$	0.2161	0.2720	0.3192	0.3599	0.5467	0.7115
不含$\mathbf{h}_i^{<t}$	0.2034	0.2541	0.3035	0.3819	0.5383	0.7253
不含阶段 1	0.1953	0.2423	0.2869	0.3489	0.5082	0.6869
DEPS	**0.2256**	**0.2766**	**0.3273**	**0.4093**	**0.5852**	**0.7747**

表 4：DEPS 与非序贯逆倾向评分方法在 Amazon-Digital-Music 测试集上的性能比较。

指标	NDCG@K			HR@K		
K	5	10	20	5	10	20
物品-Pro	0.1984	0.2385	0.1928	0.3819	0.5220	0.7170
用户-Pro	0.2044	0.2494	0.2983	0.3681	0.5275	0.7088
物品-用户-Pro	0.2133	0.2665	0.3156	0.3736	0.5495	0.7273
DEPS	**0.2256**	**0.2766**	**0.3273**	**0.4093**	**0.5852**	**0.7747**

数据集，我们进行了消融研究，通过移除表3所示的这些组件来测试不同 DEPS 变体的性能。这些 DEPS 变体包括：在估计倾向得分和进行推荐时不使用$\mathbf{h}_u^{<t}$（记为"w/o $\mathbf{h}_u^{<t}$"）、不使用$\mathbf{h}_i^{<t}$（记为"w/o $\mathbf{h}_i^{<t}$"），以及跳过第一阶段调优进行训练（记为"w/o stage-1"）。

根据表3报告的结果，我们发现，与原始 DEPS 相比，所有 DEPS 变体的表现均有所下降，这表明这些机制十分重要。具体而言，我们发现，当从模型中移除$\mathbf{h}_u^{<t}$或$\mathbf{h}_i^{<t}$的任一序列时，表现都会大幅下降，这验证了在序列推荐中同时从两个视角估计倾向得分的重要性。结果还表明，无监督学习的第一阶段确实提升了推荐准确率。

*5.3.2 序贯IPS估计方法的影响。*本节中，我们研究序贯IPS估计方法相较于非序贯IPS估计方法的影响。在我们的模型中，用户侧和物品侧IPS通过门控循环单元进行估计。我们将其与非序贯IPS估计方法（基于频率的倾向得分）进行了比较。用户侧IPS的计算方式为 $p_{u,*} = m_u / \max_{u' \in \mathcal{I}} m_{u'}$，物品侧IPS的计算方式为 $p_{*,i} = m_i / \max_{i' \in \mathcal{I}} m_{i'}$，其中 m_u和 m_i分别表示用户 u和物品 i的交互次数。

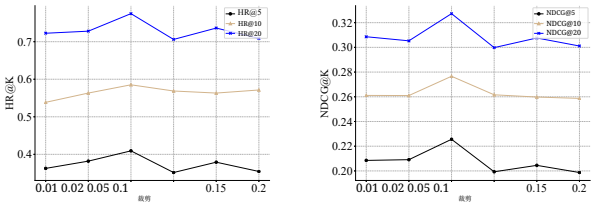
我们研究了在训练第二阶段优化无偏损失函数 Lunbiased 时，它们的无偏性能。具体而言，我们展示了若干 DEPS 变体的 NDCG@K 和 HR@K，包括使用单个倾向得分$p_{*,i}$学习推荐模型（记为"物品-Pro"）、使用单个倾向得分 $p_{u,*}$学习推荐模型（记为"用户-Pro"），以及使用双重倾向得分 $p_{*,i}$，$p_{u,*}$学习推荐模型（即在 DEPS 中分别将估计的倾向得分 $P(i, \mathbf{h}_u^{<t})$ $P(u, \mathbf{h}_i^{<t})$ 替换为 $p_{*,i}$，$p_{u,*}$，并记为"物品-用户-Pro"）。

从表 4 所示的性能可以看出：（1）"DEPS"的表现大幅优于"物品-用户-Pro"，这表明按序估计倾向得分十分重要；（2）"物品-Pro"和"用户-Pro"的表现均不如"物品-用户-Pro"，这表明从两个视角（用户或物品）估计的倾向得分都有效且互为补充。

*5.3.3 DEPS 作为模型无关框架。*尽管 DEPS 设计了一个基于 Transformer 的模型来进行推荐，但也可以将底层模型（第4.3节所示的 Transformer 和多层感知机）替换为其他序列推荐模型，从而将其用作模型无关框架。在实验中，我们将其替换为 GRU4Rec+ [32] 和 FPMC [25],

表5：将底层推荐模型替换为 GRU4Rec+或 FPMC 后，DEPS 不同变体之间的性能比较。实验在 Amazon-Digital-Music 上进行，“$*$”表示相较于底层模型的提升具有统计显著性（t检验，p-值 < 0.05）。

指标	NDCG@K			HR@K		
TopK	5	10	20	5	10	20
GRU4Rec+	0.0651	0.0961	0.1385	0.1709	0.3025	0.4874
DEPS(GRU4Rec+)	**0.0861***	**0.1219***	**0.1608***	**0.1905***	**0.3272***	**0.5182***
FPMC	0.1355	0.1799	0.2153	0.2941	0.4391	0.6091
DEPS(FPMC)	**0.1396***	**0.1858***	**0.2189***	**0.3025***	**0.4566***	**0.6426***



(a) HR@K 曲线 (b) NDCG@K 曲线

图6：DEPS 关于裁剪值的 NDCG 和命中率曲线 M

由此得到两个新模型，分别记为“DEPS (GRU4Rec+)”和“DEPS (FPMC)”。请注意，GRU4Rec 和 FPCM 的序列推荐模型无法在 MLM 任务上训练。因此，DEPS (GRU4Rec+) 和 DEPS (FPMC) 第一阶段训练中的损失函数退化为 LAR_u + LAR_i。根据表5中的结果，我们发现，DEPS (GRU4Rec+) 和 DEPS (FPMC) 相较于其底层模型 GRU4Rec+ 和 FPMC 分别取得了提升。结果表明，DEPS 估计的倾向得分具有通用性，可以以与模型无关的方式用于改进其他序列推荐模型。

5.3.4 裁剪值 M 的影响。根据定理2，裁剪值 M 平衡了 DEPS 中的无偏性和方差。在本实验中，我们研究了当裁剪值 M 设为 $[0.05.0.2]$ 中的不同值时，NDCG@K 和 HR@K 如何变化。从图6所示的曲线可以看出，性能在 $M \in [0.01, 0.05]$ 时有所提升，随后在 $[0.05, 0.2]$ 之间下降。结果验证了理论分析：M 过小（例如，$M = 0.01$）会导致方差估计较大，而过大（例如，$M = 0.2$）则会导致偏差较大。在实际应用中，平衡无偏性和方差至关重要。

6 结论

本文提出了一种新颖的逆倾向评分估计方法，称为双重增强倾向得分估计（DEPS），用于纠正序列推荐中的曝光偏差或选择偏差。DEPS从物品和用户两个视角估计倾向得分，并具有多项优势：理论可靠、模型无关以及端到端学习。在四个真实数据集上的大量实验结果表明，DEPS能够显著优于无偏测试设置下最先进的基线方法。

致谢

本研究获中国国家重点研发计划（2019YFE0198200）、国家自然科学基金（61872338、62102420、61832017）以及北京市优秀青年科学家计划（编号：BJJWZYJH012019100020098）资助。

A 定理证明

A.1 定理1的证明

证明。设 $q_{\mathbf{h}_u^{<t},i} \in \{0,1\}$，其中 $q_{\mathbf{h}_u^{<t},i} = 1$ 表示物品 i出现在历史交互句 $\mathbf{h}_u^{<t}$中，否则 $q_{\mathbf{h}_u^{<t},i} = 0$。根据定义，在序列推荐中，$P(q_{\mathbf{h}_u^{<t},i} = 1) = P(i, \mathbf{h}_u^{<t})$。将历史信息 $\mathbf{h}_u^{<t}, \mathbf{h}_i^{<t}$简记为 H。因此，

$$\begin{aligned}\mathbb{E}_o\left[\mathcal{L}_u\right] &= \mathbb{E}_o\left[\sum_{u\in\mathcal{U}}\sum_{(i,c_t)\in\mathcal{D}^u}\left[\frac{\delta(c_t, \hat{r}_t(u, i^t, H))}{P(i^t, \mathbf{h}_u^{<t})}\right]\right] \\ &= \sum_{u\in\mathcal{U}}\sum_{t:(u,i',c_t)\in\mathcal{D}}\sum_{i\in\mathcal{I}}\mathbb{E}_q\left[q_{\mathbf{h}_u^{<t},i}\cdot\frac{\delta(r_t, \hat{r}_t(u,i,H))}{P(i, \mathbf{h}_u^{<t})}\right] \\ &= \sum_{u\in\mathcal{U}}\sum_{t:(u,i',c_t)\in\mathcal{D}}\sum_{i\in\mathcal{I}}\delta(r_t, \hat{r}_t(u,i,H)) = \mathcal{L}_s^{\text{ideal}}\end{aligned}$$

同样，令 $q_{\mathbf{h}_i^{<t},u} \in \{0,1\}$，其中 $q_{\mathbf{h}_i^{<t},u} = 1$ 表示用户 u 出现在历史交互句 $\mathbf{h}_i^{<t}$中。我们有$P(q_{\mathbf{h}_i^{<t},u} = 1) = P(u, \mathbf{h}_i^{<t})$，并且

$$\begin{aligned}\mathbb{E}_o\left[\mathcal{L}_i\right] &= \mathbb{E}_o\left[\sum_{i\in\mathcal{I}}\sum_{(i,c_t)\in\mathcal{D}^i}\left[\frac{\delta(c_t, \hat{r}_t(u^{l(i,t+1)}, i, H))}{P(u^{l(i,t+1)}, \mathbf{h}_i^{<t})}\right]\right] \\ &= \sum_{i\in\mathcal{I}}\sum_{t:(u,i',c_t)\in\mathcal{D}}\sum_{u\in\mathcal{U}}\mathbb{E}_q\left[q_{\mathbf{h}_i^{<t},u}\frac{\delta(r, \hat{r}(u,i,H))}{P(u, \mathbf{h}_i^{<t})}\right] \\ &= \sum_{u\in\mathcal{U}}\sum_{t:(u,i',c_t)\in\mathcal{D}}\sum_{i\in\mathcal{I}}\delta(r_t, \hat{r}_t(u,i,H)) = \mathcal{L}_s^{\text{ideal}}.\end{aligned}$$

Therefore, $\mathcal{L}_s^{\text{ideal}} = \mathbb{E}\left[\mathcal{L}_s^{\text{unbiased}}\right] = \mathbb{E}\left[\alpha\mathcal{L}_u + (1-\alpha)\mathcal{L}_i\right]$ □

A.2 定理2的证明

证明 沿用定理1中定义的记号，我们可以写作 $L_u^t = \frac{\delta(r_t, \hat{r}_t)\, p(u, \mathbf{h}_i^{<t}) q_{\mathbf{h}_i^{<t},u}}{\widetilde{P}(u, \mathbf{h}_i^{<t})}$，根据公式（7）所示的裁剪操作，我们有 $\widetilde{P}(u, \mathbf{h}_i^{<t}) \geq M$。因此，

$$\begin{aligned}\mathbb{V}\left[L_u^t\right] &= \mathbb{E}\left[L_u^t\right]^2 - \left(\mathbb{E}\left[L_u^t\right]\right)^2 = \frac{\delta^2(r_t, \hat{r}_t)}{\widetilde{P}(u, \mathbf{h}_i^{<t})} - \delta^2(r_t, \hat{r}_t) \\ &= \left(\frac{1}{\widetilde{P}(u, \mathbf{h}_i^{<t})} - 1\right)\delta^2(r_t, \hat{r}_t) \leq \left(\frac{1}{M} - 1\right)\delta^2(r_t, \hat{r}_t).\end{aligned}$$

同样，我们可以将目标函数的方差 L_i^t界定为：

$$\mathbb{V}\left[L_i^t\right] = \left(\frac{1}{\widetilde{P}(i, \mathbf{h}_u^{<t})} - 1\right)\delta^2(r_t, \hat{r}_t) \leq \left(\frac{1}{M} - 1\right)\delta^2(r_t, \hat{r}_t).$$

应用柯西-施瓦茨不等式，我们可以将平均损失 Lunbiased $_s = \alpha L_i^t + (1-\alpha)L_u^t$ 的方差界定为：

$$\begin{aligned}\mathbb{V}\left[\mathcal{L}_s^{\text{unbiased}}\right] &= \alpha^2\mathbb{V}\left[L_i^t\right] + (1-\alpha)^2\mathbb{V}\left[L_u^t\right] + 2\alpha(1-\alpha)\text{Cov}(L_i^t, L_u^t) \\ &\leq \alpha^2\mathbb{V}\left[L_i^t\right] + (1-\alpha)^2\mathbb{V}\left[L_u^t\right] + 2\alpha(1-\alpha)\sqrt{\mathbb{V}\left[L_i^t\right]\mathbb{V}\left[L_u^t\right]} \\ &\leq \max\left\{\mathbb{V}\left[L_i^t\right], \mathbb{V}\left[L_u^t\right]\right\} \leq (\frac{1}{M}-1)\delta^2(r,\hat{r}).\end{aligned}$$

□

参考文献

[1] Himan Abdollahpouri 和 Masoud Mansoury。2020。推荐中的多侧曝光偏差。*CoRR* abs/2006.15772（2020）。[2] Aman Agarwal、Ivan Zaitsev、Xuanhui Wang、Cheng Li、Marc Najork 和 Thorsten Joachims。2019。无需侵入式干预的位置偏差估计。载于第十二届 *ACM* 国际网络搜索与数据挖掘会议论文集。474–482。[3] Jiawei Chen、Hande Dong、Xiang Wang、Fuli Feng、Meng Wang 和 Xiangnan He。2020。推荐系统中的偏差与去偏：综述与未来方向。*CoRR* abs/2010.03240（2020）。[4] Jiawei Chen、Yan Feng、Martin Ester、Sheng Zhou、Chun Chen 和 Can Wang。2018。利用社会知识影响和消费影响建模用户曝光以进行推荐。载于第*27*届 *ACM* 国际信息与知识管理会议论文集。953–962。[5] Jiawei Chen、Can Wang、Sheng Zhou、Qihao Shi、Jingbang Chen、Yan Feng 和 Chun Chen。2020。针对隐式反馈推荐的快速自适应加权矩阵分解。载于*AAAI*人工智能会议论文集，第34卷。3470–3477。[6] Jiawei Chen、Can Wang、Sheng Zhou、Qihao Shi、Yan Feng 和 Chun Chen。2019。Samwalker：采用信息性采样策略的社交推荐。载于万维网大会。228–239。[7] Qiwei Chen、Huan Zhao、Wei Li、Pipei Huang 和 Wenwu Ou。2019。用于阿里巴巴电子商务推荐的行为序列变换器。载于首届高维稀疏数据深度学习实践国际研讨会论文集。1–4。[8] Kyunghyun Cho、Bart van Merrienboer、Çaglar Gülçehre、Dzmitry Bahdanau、Fethi Bougares、Holger Schwenk 和 Yoshua Bengio。2014。用于统计机器翻译的 RNN编码器-解码器短语表示学习。载于*2014*年自然语言处理实证方法会议（*EMNLP*）论文集。ACL，1724–1734。[9] Andrew Collins、Dominika Tkaczyk、Akiko Aizawa 和 Joeran Beel。2018。数字图书馆推荐系统中位置偏差的研究。*arXiv* 预印本 *arXiv:1802.06565*（2018）。[10] Robin Devooght、Nicolas Kourtellis 和 Amin Mantrach。2015。对未知值设置先验的动态矩阵分解。载于第*21*届 *ACM SIGKDD*会议论文集。189–198。[11] Xinyan Fan、Zheng Liu、Jianxun Lian、Wayne Xin Zhao、Xing Xie 和 Ji-Rong Wen。2021。更轻更优：用于下一物品推荐的低秩分解自注意力网络。载于第*44*届 *ACM SIGIR*信息检索研究与发展会议论文集。1733–1737。[12] Ziwei Fan、Zhiwei Liu、Jiawei Zhang、Yun Xiong、Lei Zheng 和 Philip S Yu。2021。利用时序图协同变换器进行连续时间序列推荐。载于第*30*届 *ACM CIKM*会议论文集。433–442。[13] Zhichong Fang、Aman Agarwal 和 Thorsten Joachims。2019。通过干预采集估计依赖上下文的检查偏差。载于第*42*届 *ACM SIGIR*信息检索研究与发展会议论文集。825–834。[14] Yifan Hu、Yehuda Koren 和 Chris Volinsky。2008。针对隐式反馈数据集的协同过滤。载于*2008*年第八届 *IEEE* 国际数据挖掘会议。IEEE，263–272。[15] Thorsten Joachims、Adith Swaminathan 和 Tobias Schnabel。2017。基于有偏反馈的无偏学习排序。载于第十届 *ACM* 国际网络搜索与数据挖掘会议论文集。781–789。[16] Diederik P. Kingma 和 Jimmy Ba。2015。Adam：一种随机优化方法。载于第三届国际学习表征会议。[17] Dugang Liu、Pengxiang Cheng、Zhenhua Dong、Xiuqiang He、Weike Pan 和 Zhong Ming。2020。通过均匀数据进行反事实推荐的通用知识蒸馏框架。载于第*43*届 *ACM SIGIR*会议论文集。831–840。[18] Qiao Liu、Yifu Zeng、Refuoe Mokhosi 和 Haibin Zhang。2018。STAMP：用于基于会话的推荐的短期注意力/记忆优先模型。载于第*24*届 *ACM SIGKDD*论文集。1831–1839。[19] Benjamin Marlin、Richard S Zemel、Sam Roweis 和 Malcolm Slaney。2012。协同过滤与随机缺失假设。*arXiv* 预印本 *arXiv:1206.5267*（2012）。[20] Ankur Parikh、Oscar Täckström、Dipanjan Das 和 Jakob Uszkoreit。2016. 用于自然语言推理的可分解注意力模型。载于《2016年EMNLP会议论文集》。2249–2255。[21] Adam Paszke、Sam Gross、Soumith Chintala、Gregory Chanan、Edward Yang、Zachary DeVito、Zeming Lin、Alban Desmaison、Luca Antiga 和 Adam Lerer。2017。PyTorch 中的自动微分。（2017）。[22] Judea Pearl。2009。统计学中的因果推断：概述。《统计调查》3（2009），96–146。[23] Zhen Qin、Suming J Chen、Donald Metzler、Yongwoo Noh、Jingzheng Qin 和 Xuanhui Wang。2020。用于推荐系统无偏学习的基于属性的倾向性：算法与案例研究。载于《第26届ACM SIGKDD知识发现与数据挖掘国际会议论文集》。2359–2367。[24] Steffen Rendle、Christoph Freudenthaler、Zeno Gantner 和 Lars Schmidt-Thieme。2012。BPR：基于隐式反馈的贝叶斯个性化排序。arXiv预印本 arXiv:1205.2618（2012）。[25] Steffen Rendle、Christoph Freudenthaler 和 Lars Schmidt-Thieme。2010。用于下一篮推荐的个性化马尔可夫链分解。载于《第19届国际万维网会议论文集》。811–820。[26] Yuta Saito。2020。用于去偏非随机缺失显式反馈的非对称三重训练。载于《第43届ACM SIGIR信息检索研究与发展国际会议论文集》。309–318。[27] Yuta Saito。2020。基于有偏隐式反馈的无偏成对学习。载于《2020年ACM SIGIR信息检索理论国际会议论文集》。5–12。[28] Yuta Saito、Suguru Yaginuma、Yuta Nishino、Hayato Sakata 和 Kazuhide Nakata。2020。基于非随机缺失隐式反馈的无偏推荐学习。载于《第13届网络搜索与数据挖掘国际会议论文集》。501–509。[29] Yuta Saito、Suguru Yaginuma、Yuta Nishino、Hayato Sakata 和 Kazuhide Nakata。2020。基于非随机缺失隐式反馈的无偏推荐学习。载于《第13届网络搜索与数据挖掘国际会议论文集》。501–509。[30] Tobias Schnabel、Adith Swaminathan、Ashudeep Singh、Navin Chandak 和 Thorsten Joachims。2016。推荐作为处理：去偏学习与评估。载于国际机器学习大会。PMLR，1670–1679。[31] Fei Sun、Jun Liu、Jian Wu、Changhua Pei、Xiao Lin、Wenwu Ou 和 Peng Jiang。2019。BERT4Rec：采用双向 Transformer 编码器表示的序列推荐。载于《第28届ACM国际信息与知识管理会议论文集》。1441–1450。[32] Yong Kiam Tan、Xinxing Xu 和 Yong Liu。2016。用于基于会话的推荐的改进循环神经网络。载于《第1届推荐系统深度学习研讨会论文集》。17–22。[33] Qi Wan、Xiangnan He、Xiang Wang、Jiancan Wu、Wei Guo 和 Ruiming Tang。2022。用于无偏物品推荐的交叉成对排序。载于《2022年ACM网络会议论文集》。2370–2378。[34] Zhenlei Wang、Shiqi Shen、Zhipeng Wang、Bo Chen、Xu Chen 和 Ji-Rong Wen。2022。基于潜在混杂因素的无偏序列推荐。载于《2022年ACM网络会议论文集》。2195–2204。[35] Zhenlei Wang、Jingsen Zhang、Hongteng Xu、Xu Chen、Yongfeng Zhang、Wayne Xin Zhao 和 Ji-Rong Wen。2021。反事实数据增强的序列推荐。载于《第44届ACM SIGIR信息检索研究与发展国际会议论文集》。347–356。[36] Hong-Jian Xue、Xinyu Dai、Jianbing Zhang、Shujian Huang 和 Jiajun Chen。2017。推荐系统的深度矩阵分解模型。载于国际人工智能联合会议，第17卷。澳大利亚墨尔本，3203–3209。[37] Shengyu Zhang、Dong Yao、Zhou Zhao、Tat-Seng Chua 和 Fei Wu。2021。Causerec：用于序列推荐的反事实用户序列合成。载于《第44届ACM SIGIR信息检索研究与发展国际会议论文集》。367–377。[38] Yang Zhang、Fuli Feng、Xiangnan He、Tianxin Wei、Chonggang Song、Guohui Ling 和 Yongdong Zhang。2021。通过因果干预利用推荐中的流行度偏差。arXiv预印本 arXiv:2105.06067（2021）。[39] Wayne Xin Zhao、Shanlei Mu、Yupeng Hou、Zihan Lin、Yushuo Chen、Xingyu Pan、Kaiyuan Li、Yujie Lu、Hui Wang、Changxin Tian、Yingqian Min、Zhichao Feng、Xinyan Fan、Xu Chen、Pengfei Wang、Wendi Ji、Yaliang Li、Xiaoling Wang 和 Ji-Rong Wen。2021。RecBole：面向推荐算法的统一、全面且高效的框架。arXiv: 2011 01731 [cs.IR]。[40] Yu Zheng、Chen Gao、Xiang Li、Xiangnan He、Yong Li 和 Depeng Jin。2020。利用因果嵌入解耦推荐中的用户兴趣与流行度偏差。arXiv预印本 arXiv:2006.11011（2020）。[41] Yu Zheng、Chen Gao、Xiang Li、Xiangnan He、Yong Li 和 Depeng Jin。2021。利用因果嵌入解耦推荐中的用户兴趣与从众性。载于《2021年网络会议论文集》。2980–2991。[42] Guorui Zhou、Na Mou、Ying Fan、Qi Pi、Weijie Bian、Chang Zhou、Xiaoqiang Zhu 和 Kun Gai。2019。用于点击率预测的深度兴趣演化网络。载于《AAAI人工智能会议论文集》，第33卷。5941–5948。

[43] Guorui Zhou、Xiaoqiang Zhu、Chenru Song、Ying Fan、Han Zhu、Xiao Ma、Yanghui Yan、Junqi Jin、Han Li 和 Kun Gai。2018。用于点击率预测的深度兴趣网络。载于 第*24*届*ACM SIGKDD*知识发现与数据挖掘国际会议论文集。1059–1068。[44] Kun Zhou、Hui Wang、Wayne Xin Zhao、Yutao Zhu、Sirui Wang、Fuzheng Zhang、Zhongyuan Wang 和 Ji-Rong Wen。2020。S3-rec：利用互信息最大化进行序列推荐的自监督学习。载于第*29*届*ACM*信息与知识管理国际会议论文集。1893–1902。