中山大學
SUN YAT-SEN UNIVERSITY

机器学习期末报告

基于 CPR 的复现及改进

姓 名 Lunacht、fourteenice

学 号 21424532、21452279

学 院 人工智能学院

专 业 人工智能

September 25, 2024

Contents

1 背景介绍

推荐系统是电子商务、流媒体和社交网络等平台的关键技术，它们主要依赖于用户与物品之间的历史交互数据，如点击和购买记录来学习用户偏好。常用的损失函数包括二元交叉熵（BCE）[1] 和贝叶斯个性化排序（BPR）[2]，但这些方法往往会因数据中的热门物品偏差而影响推荐质量，导致热门物品过度推荐，而冷门物品被忽视的现象。

为了解决偏差问题，逆向倾向评分（IPS）方法 [3] 通过重新加权数据样本的倾向评分来实现无偏学习。然而，IPS 方法在实际应用中存在倾向评分难以准确估计和高方差的问题。尽管现在很多无偏算法被提出，但是很多算法仍然无法真正解决**偏差问题**且**训练耗时非常长**。

2 建模方法

论文的创新点主要在于提出了一种新的交叉成对损失函数 CPR[4]，以实现无偏性；此外，论文设计了一种动态采样算法，以加快训练速度并提高其性能。除此之外，我们根据论文的损失函数进行改进，使其更加无偏。[1]

2.1 交叉成对损失函数

交叉成对损失函数（CPR Loss）旨在通过比较正向对和负向对的预测得分之和，提高模型在无偏学习环境下的性能。公式如下：

$$L_{CPR} = -\sum_{(u_1,u_2,i_1,i_2)\in D_2} \ln \sigma \left(\frac{1}{2}(s_{u_1,i_1} + s_{u_2,i_2} - s_{u_1,i_2} - s_{u_2,i_1}) \right) \tag{1}$$

其中 σ 是 sigmoid 函数。D_2 表示包含训练数据集中正负交互对的集合。$s_{u,i}$ 表示用户 u 和物品 i 之间的预测得分。(u_1, u_2, i_1, i_2) 是一个训练样本，包含两个正向用户-物品对 (u_1, i_1) 和 (u_2, i_2) 以及两个负向用户-物品对 (u_1, i_2) 和 (u_2, i_1)。

之后论文提出采用多用户-样本组成交互数样本的方法来计算损失函数。图 1展示了不同交互数样本的组成，在一个 CPR 样本中，每个用户和各自的正对物品组成正对样本，和其他的物品组成负对样本。

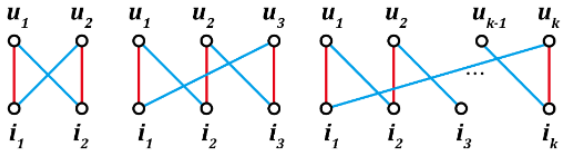


Figure 1: CPR 样本的组成 (红线表示正对, 蓝线表示负对)。图从左到右分别表示 D_2、D_3 和 D_k 中样品的组成。

[1] 本研究的相关代码可通过访问https://github.com/Lunacht/ML_3-12获取。

根据以上定义，论文将损失函数其扩展到 k（$k \geq 2$）个交互，同时保持其无偏性：

$$L_{CPR} = -\sum_{k} \sum_{(u_1,...,u_k,i_1,...,i_k) \in D_k} \ln \sigma \left(\frac{1}{k} \left(\hat{s}_{u_1,i_1} + \cdots + \hat{s}_{u_k,i_k} - \hat{s}_{u_1,i_2} - \cdots - \hat{s}_{u_k,i_1} \right) \right) \quad (2)$$

其中 D_k 为训练数据集中正负交互对的集合，定义为 $D_k = \{(u_1, ..., u_k, i_1, ..., i_k) \mid Y_{u_1,i_1} = 1, Y_{u_2,i_2} = 1, ..., Y_{u_k,i_k} = 1, Y_{u_1,i_2} = 0, Y_{u_2,i3} = 0, ..., Y_{u_k,i_1} = 0\}$。

在论文的实验中，CPR 通常在 $k = 2$ 和 $k = 3$ 时达到最佳性能，当 k 取大于 3 的值时性能降低 [4]。这可能是因为较大的 k 值降低了模型训练的灵活性，甚至导致过拟合。因此在后续实验中，我们的 k 设置为 2,3。

结合以上定义的 L_{CPR} 和正则化损失得到完整的 CPR 损失函数：

$$L = L_{CPR} + \lambda(\|\Theta\|^2) \quad (3)$$

2.2 动态采样策略

样本集 D_k 的构建在 CPR 中起着关键作用。在简单的随机采样中，所有样本选中的概率都是一样的。在训练过程中，一些困难样本 [5] 需要更多迭代才能收敛，但是对加快模型收敛速度的贡献较大，所以期望更多地采样到困难样本。

因此，CPR 设计了一种动态采样策略，赋予困难样本更高的采样概率，并更频繁地训练它们。使用 b、$\beta(\beta \geq 1)$ 和 $\gamma(\gamma > 1)$ 分别表示批量大小、动态采样率和选择率。首先，算法随机选择 $b\beta\gamma$ 个样本，每个样本包含 k 个观测交互。选择率 γ 用于增加初始样本的数量，以确保在下一步丢弃不适当样本后仍能收集到所需数量的样本。接下来，丢弃那些交叉组合不是 k 个负向对的样本，并获得 $b\beta$ 个有效样本。动态采样通过计算以下值来衡量一个样本的难度：

$$O_j = \frac{1}{k} \left(\hat{s}_{u_1,i_1} + \hat{s}_{u_2,i_2} + \cdots + \hat{s}_{u_k,i_k} - \hat{s}_{u_1,i_2} - \hat{s}_{u_2,i3} - \cdots - \hat{s}_{u_k,i_1} \right) \quad (4)$$

O_j 的值越小，样本达到目标排序的难度越大。因此，动态采样选择这些表达式中值最小的样本作为一批样本。

这种动态采样策略有效地选择了较难的训练样本。通过这种方法，CPR 可以在不增加过多计算成本的情况下，加速收敛并提高推荐质量。

3 实验

3.1 数据处理

3.1.1 数据集概要

CPR 适配于用户-物品交互数据集。在大作业提供的三个数据集中，MovieLens 20M[6] 和 Amazon-Book[7] 分别是用户对电影和图书的评分，符合 CPR 的要求；而 Gowalla[8] 是一个包含用户签到的位置信息的数据集，并不适合作为 CPR 的训练数据。

综上，本次实验的四个数据集选定为：MovieLens 20M、Amazon-Book 和原论文使用的数据集 Netflix Prize[9]、Alibaba iFashion[10]。

四个数据集中，iFashion 记录了用户对物品的点击情况，而 Netflix、MovieLens 和 AmazonBook 记录的是用户对物品的打分，对于这三个数据集，需要将评分数据进行二值化处理（将五星评分设为有交互，其余设为无交互）。

经过处理后，初始数据集被转换为了统一格式，每条数据由一个 UserID - ItemID 对组成，表示用户与物品之间存在一次正面交互。

Table 1: 数据集统计信息

数据集	用户数	物品数	交互数	core 数量	稠密度	最大采样率
MovieLens	119,241	10,298	2,874,084	3	0.0023	1/60
AmazonBook	83,355	68,102	1,019,396	5	0.0001	1/12
Netflix	46,420	12,898	2,475,020	3	0.0041	1/60
iFashion	264,251	53,033	1,505,141	3	0.0001	1/12

3.1.2 数据集稠密度

表格中的稠密度 (Density) 指数据集的交互稠密度，计算公式为式 5。

$$\text{Density} = \frac{N_{\text{interactions}}}{N_{\text{users}} \times N_{\text{items}}} \tag{5}$$

其中，$N_{\text{interactions}}$ 是数据集的用户-物品交互的数量，N_{users} 是数据集中用户的数量，N_{items} 是数据集中物品的数量。

对于稠密度低的大型数据集，存在**数据稀疏** [11] 的问题，这会导致训练模型时需要更多的计算资源和时间。CPR 采用 **k-core 过滤** [12] 的预处理方法，将交互次数低于 k 的用户和物品去除，这提升了数据集的稠密度，同时减少了噪音数据。

3.1.3 数据集采样

去偏模型需要使用有偏的训练集进行训练，然后在无偏的测试集上进行评估，以此检测去偏效果。所以需要对数据集进行采样，提取出无偏的测试集、验证集。

CPR 参照已有研究中的离线评估协议 [13]，使用倾向性评分 [14] 的方法创建模拟无偏数据。具体方法为：定义 $p_s(u,i)$ 为采样用户-物品对 (u,i) 的概率，采样概率的计算公式为：$p_s(u,i) \propto \min\left(\frac{1}{d_i}, a\right)$，其中 a 是上限，也就是表 1中的最大采样率。对于 MovieLens 和 Netflix 数据集，上限 a 设为 $\frac{1}{60}$；对于更稀疏的数据集 iFashion 和 AmazonBook，设为 $\frac{1}{12}$。

采样得到的模拟无偏数据随机分为验证集和测试集，剩下的数据作为训练集。最终得到的训练集、验证集和测试集的比例为 70/10/20。

3.2 数据动态采样

根据论文提出的动态采样方法，可以将实现代码可以分成参数初始化、随机采样、排序提取困难样本三个部分。

- **参数初始化：**对数据采样器 CPRSampler(python 类, 位于 recq/utils/data.py) 的参数进行初始化。通过 args 指定的 k、采样率 sample_rate、采样比率 sample_ratio 计算出采样数量等参数。

- **随机采样：**随机采样部分使用了针对性优化的 Cython 实现。代码通过嵌套实例化类的方式实现了从 python 类到 C++ 类的转换，这减少了 Python 解释器的运行开销，也是 Cython 高效的原因。

  在 CppCPRSampler 类 (C++ 类，位于 recq/cyutils/include/sampler.h) 中定义了随机采样函数 CPRSample1Batch，该函数实现了从用户-物品交互数据中批量采样负对的功能。随机抽取 k 个不重叠的正用户-物品对，如果它们的配套组合都是负对，则接受这些负对，否则丢弃它们并重新采样。

- **排序提取困难样本：**在随机采样得到正负对数据之后，需要根据式 4对这些正负对进行打分和排序。在 cpr_loss 函数 (位于 recq/utils/loss.py) 中实现此排序功能，选择分数最低的 batch_size 个样本投入训练。

3.3 训练

在代码结构中，CPR 类（位于 recq/recommenders/cpr.py）继承自 BPR 类（位于 recq/recommenders/bpr.py)。因此，在训练过程中，CPR 类主要基于 BPR 类的实现进行函数调用。不同之处主要体现在损失函数和数据采样的处理上。以下将详细介绍 CPR 方法的训练过程:

在训练之前，代码根据 args 设定的值，设置各项参数。首先，模型会**对用户和项目的嵌入向量进行随机初始化**，通过优化过程，嵌入向量会不断调整，以更准确地反映用户的偏好和项目的属性。

在每个训练周期，通过 **CPRSampler** 抽取用户与项目的交互样本。模型对每个样本预测用户对物品的评分，并使用优化器根据损失函数的梯度来更新模型参数。为防止模型过拟合，训练过程中还会加入正则化项的计算，以保持模型的泛化能力。此外，代码会根据设定定期在验证集上评估模型，使用各种指标监控模型在未见数据上的表现。如果模型在验证集上的表现不再提升，则采用早停策略终止训练，避免不必要的计算和过拟合。

3.4 损失函数定义

在损失函数的定义中，CPR 使用 $Loss_{CPR}$ 和 $Loss_{REG}$ 两部分来组成损失函数，其中 $Loss_{CPR}$ 由式 3定义，从而实现无偏的损失函数 (在 recq/utils/loss.py 中实现);$Loss_{REG}$ 通过添加参数的 L2 范数来控制模型的复杂度，避免产生过拟合 (在 recq/recommenders/bpr.py 中实现)。

3.4.1 $Loss_{CPR}$

$Loss_{CPR}$ 主要通过以下几个步骤实现：

1. **嵌入查找:** 在矩阵分解模型中，索引集合中的每个物品 j 都与一个嵌入向量 v_j 关联，该嵌入向量可以从预训练的物品嵌入矩阵 I 中检索得到。

2. **得分计算:** 使用点积计算用户嵌入 u_i 与物品嵌入 v_j 之间的相似度得分。对于正样本（用户实际交互过的物品），得分计算如下：

$$s_{ij}^{+} = u_i \cdot v_j$$

   对于负样本（用户未交互过的物品），系统采用负采样技术洗牌或随机选择物品嵌入，得分计算如下：

$$s_{ij}^{-} = u_i \cdot v_{j'}$$

   其中 j' 表示随机选择或洗牌后未交互物品的索引。

3. **得分聚合:** 得到所有正负样本的得分后，将所有正样本得分聚合到一个向量 S^{+} 中，将所有负样本得分聚合到另一个向量 S^{-} 中，便于后续的损失计算。

4. **损失计算:** 最终通过式 3计算 CPR 损失。如果正样本得分普遍高于负样本得分，则模型表现良好；反之，则需要调整。

3.4.2 $Loss_{REG}$

正则化损失通过计算嵌入向量和网络权重的 L2 损失来防止模型的过拟合。在这个方法中，首先计算了嵌入向量 self.all_embeds_0 的 L2 损失，并乘以正则化系数 args.reg。接着，根据模型的嵌入类型，进一步计算每层权重和偏置的 L2 损失。

3.4.3 评估指标

代码使用 NDCG、Recall、Precision、ARP[15] 作为评估指标，在每隔指定的 epoch 对模型进行评估。由于在期中报告中已经有详细说明，因此这里不再赘述。

4 技术创新点

4.1 损失函数的改进

在 DICE 的启发下，我们对 CPR 的损失函数进行了改进。根据 DICE 提出的因果模型，我们将样本对分为流行和感兴趣两个样本类型，旨在评估物品的流行度与用户的个人兴趣对选择偏好的影响，从而提供更精确的个性化推荐。

在计算过程中，我们首先将每对正负样本进一步分类为流行度和兴趣度，并分别计算 $\hat{s}^{pop}_{u_1,i_1}$ 和 $\hat{s}^{int}_{u_1,i_1}$，之后再对其进行求和作为 $\hat{s}_{u_1,i_1}$ 的取值:

$$\hat{s}_{u_1,i_1} = \hat{s}^{pop}_{u_1,i_1} + \hat{s}^{int}_{u_1,i_1} \tag{6}$$

最后将式 3中的 $\hat{s}_{u_1,i_1}$ 用式 6替换，从而求出最终的 $Loss_{CPR}$。

通过将样本对按照流行度和兴趣度进行分类，并为每一类独立计算损失，我们的模型不仅维持了对用户个性化体验的支持，而且增强了对市场流行趋势的响应能力。这种方法使得推荐系统可以更灵活地适应用户需求和市场变化。我们将改进后的模型称为 CPR_Improved（简称 CPR_Im）。

5 结果分析

5.1 CPR 及改进模型在四个数据集上的表现

在本实验中，我们探索并评估了 CPR 及我们的改进方法 CPR_Im 在 MovieLens 20M、Amazon-Book、iFashion 以及 Netflix 数据集上的效果。此外，我们还记录了模型训练所需的最佳迭代次数（epoch），以衡量模型收敛的速度。评估结果如表 2所示。

Table 2: CPR 与 CPR_Im 在四个数据集的表现。每列中最好的结果会加粗。更高的 Recall、Presicion 和 NDCG 意味着更高的预测精度；更低的 ARP 意味着更少的流行度偏差。所有评估指标后缀 @ 20 表示仅考察每个用户的前 20 个推荐项目的准确度

Method	**MovieLens 20M**					**Amazon-Book**				
	Recall	Precision	NDCG	ARP	epoch	Recall	Precision	NDCG	ARP	epoch
CPR	**0.1944**	**0.0387**	**0.1204**	1850.3	60	0.0865	0.0082	0.0503	101.6	**22**
CPR_Im	0.1580	0.0325	0.1010	**1174.3**	**24**	**0.1206**	**0.0159**	**0.0830**	**42.4**	166
	Netflix					**iFashion**				
	Recall	Precision	NDCG	ARP	epoch	Recall	Precision	NDCG	ARP	epoch
CPR	**0.1491**	**0.0539**	**0.1172**	1981.5	**13**	0.0220	0.0017	0.0099	366.1	**24**
CPR_Im	0.1173	0.0409	0.0918	**1245.7**	39	**0.0231**	**0.0018**	**0.0110**	**152.2**	30

从表 2可以看到，在四个数据集的性能比较中，CPR 和 CPR_Im 表现出不同的优势。CPR 在 MovieLens 20M 和 Netflix 这两个电影推荐数据集上表现出色，表现出快速收敛和精度高的特性，表明它适合处理稠密的用户-物品交互数据。相比之下，CPR_Im 模型在 Amazon-Book 和 iFashion 这两个稀疏数据集上表现更佳，且在所有数据集上的 ARP 都拿到了最好的分数，显示了其改进模型的优势。

总体来看，**原始的 CPR 模型在稠密数据集上具有快速收敛和高准确性的特点，而 CPR_Im 模型则在减少模型的流行度偏见和优化稀疏数据集的处理方面展现出更强的能力**。

5.2 与其他模型的比较

为了验证 CPR 模型的先进性，我们将其与当前最先进的去偏算法 DICE 以及经典的去偏算法 UBPR 进行比较。鉴于先前研究 LightGCN 作为算法框架表现最佳，我们选用 LightGCN 作为所有模型的基础架构进行实验。

Table 3: UBPR、DICE、CPR 和 CPR_Im 模型在两个数据集上的效果比较

Method	**Backbone**	**MovieLens 20M**				**Amazon-Book**			
		Recall	Precision	NDCG	ARP	Recall	Precision	NDCG	ARP
UBPR		0.1633	0.0310	0.0973	4645.5	0.1055	0.0136	0.0679	112.5
DICE	LightGCN	0.1781	0.0356	0.1083	1983.6	0.0648	0.0082	0.0412	64.7
CPR		**0.1944**	**0.0387**	**0.1204**	1850.3	0.0865	0.0082	0.0503	101.6
CPR_Im		0.1580	0.0325	0.1010	**1174.3**	**0.1206**	**0.0159**	**0.0830**	**42.4**

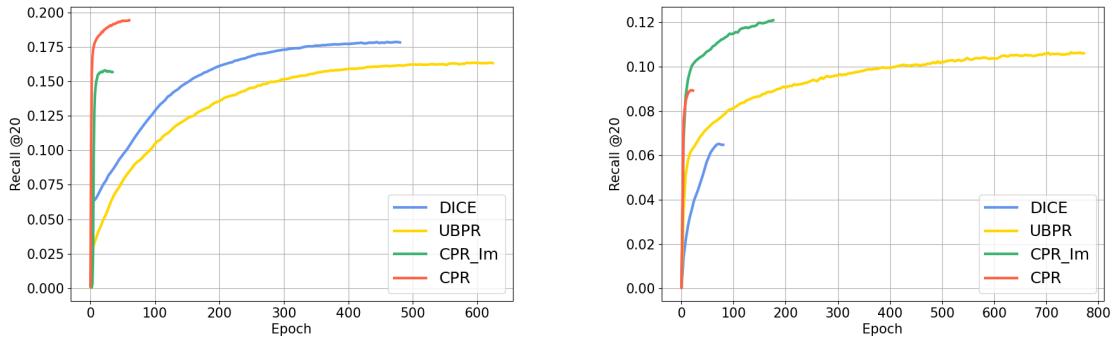
在表 3中，可以看到 CPR 在 MovieLens 20M 数据集上显示出了卓越的性能，它在召回率、精确度和 NDCG 等关键指标上都超越了其他算法。在这个数据集中，虽然 CPR_Im 模型的预测效果不如其他算法，但是其在 ARP 值上达到了最低，显示出强大的去偏能力。

在 Amazon-Book 这一相对稀疏的数据集上，CPR_Im 的表现尤为出色，不仅在召回率和 NDCG 上超越了其他模型，其 ARP 值也是最低的，展现了其在减少推荐系统中的流行度偏见方面的优势。总的来说，**CPR 及其改进版在不同类型数据集上都展现出了自己的优势和适应性，特别是在解决推荐系统中的偏见问题上表现出色**。

通过图 2四个模型的 Recall 在训练过程中随 epoch 的变化可以发现在 MovieLens 数据集上，相比于另外两个模型，CPR 和 CPR_Im 的收敛非常迅速，在数十次 epoch 内便已达到较高的召回率; 相比之下，UBPR 收敛的 epoch 数量将近 CPR 的 10 倍;DICE 速度比 UBPR 略快，但依然和 CPR 相距甚远。

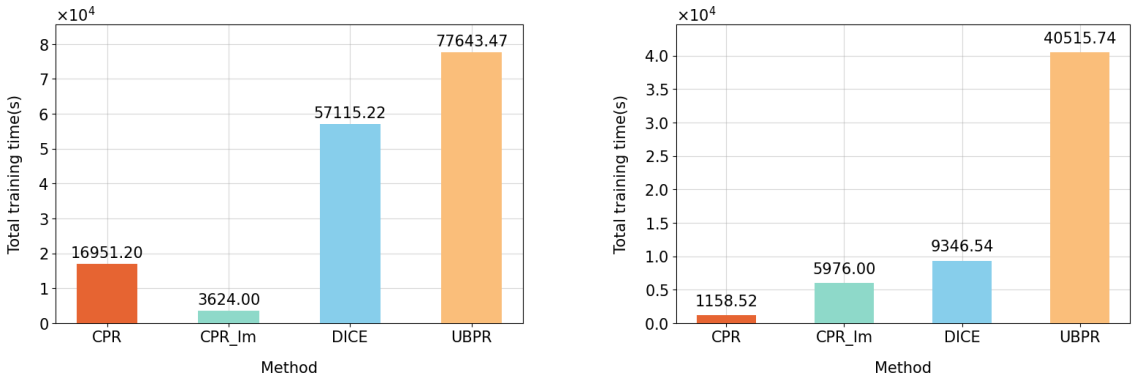
在 AmazonBook 数据集上，CPR 的收敛速度依然是最快的;DICE 的收敛速度接近了 CPR，但在指标的数值上比 CPR 差很多; CPR_Im 的收敛速度比 DICE 略慢，但最终的指标最为优秀；UBPR 的召回率比 CPR 略高，但是 epoch 数量达到了 800，接近

CPR 的 40 倍。



(a) MovieLens (b) AmazonBook

Figure 2: 四个模型召回率曲线的比较



(a) MovieLens (b) AmazonBook

Figure 3: 四个模型训练总时间的比较

通过图 3中训练总时间的比较，可以看到在两个数据集上，CPR 和 CPR_Im 的训练时间都远远低于另外两个模型。在 AmazonBook 数据集上，**CPR 只占 UBPR 训练时间的 2.85%**。

由于**动态采样**，CPR 和 CPR_Im 以**最少的迭代次数**收敛到最佳性能。虽然动态采样需要处理更多的样本来选择困难样本，但困难样本对加快模型收敛速度的帮助非常大，因此可以忽略处理更多样本的时间成本。

6 思考和改进方向

6.1 采样率对模型的影响

在实验中，我们发现不同的采样率和采样比率对模型性能有显著影响。

- **较高**的采样率和采样比率可以**加快**模型召回率的**收敛速度**，但可能导致**过拟合**且较快**达到性能瓶颈**，从而限制了模型在未见数据上的泛化能力。
- **较低**的采样率虽然初期可能导致性能下降，但随着训练的深入，模型的召回率逐渐提升，显示出**更持久**的学习和优化潜力。

这表明**适当调整**采样率和采样比率是优化推荐系统性能的关键，能够根据需求平衡学习速度与最终性能。

6.2 用户-物品对的改进

在论文中提到 k 值的增加会导致推荐系统的灵活性降低。这是因为在 k 值较大时算法仍将用户-物品作为一对一的正对，而忽视了单个用户可能对多个物品持正面评价的现实情况。为了克服这一限制并提升推荐系统的效果，我们在 CPR 的基础上提出以下策略：

1. **对物品进行分类：**对用户感兴趣的物品属性进行详细分类，并根据这些分类向用户推荐具有相似特性的物品。这种方法不仅能提供更个性化的推荐，还能增强用户的满意度。
2. **对用户进行分类：**当观察到多个用户偏好相似的物品时，可以利用聚类技术对用户进行分组。每个分组的用户可以与一组特定的物品属性相关联。这样我们能够向他们推荐这些属性相关的交叉类别物品，增加了推荐的相关性和多样性。
3. **利用用户喜好交集进行推荐：**通过分析用户间的喜好交集，可以相互推荐那些他们可能感兴趣但尚未发现的物品。这种相互推荐策略有助于发掘和推荐那些较少被注意但可能受欢迎的物品。

7 结论

本研究通过理论分析与四个不同数据集上的实验，验证了 CPR 模型在缓解推荐系统中的偏见问题上的有效性。CPR 通过创新的损失函数设计，减少了对热门物品的过度推荐，从而提高了推荐结果的均衡性。此外，我们对损失函数进行的优化改进提出了 CPR_Improved 模型，实现了更大的无偏。

此外，CPR 的动态采样策略显著提升了模型训练效率和数据利用率，通过选择信息量更大的样本，加快了模型收敛，增强了在未见数据上的泛化能力。总体来看，CPR 及其改进版本为解决推荐系统中的偏见问题提供了有效方案，展示了在提高推荐质量和公平性方面的实际应用潜力。

8 小组分工

Lunacht：

1. **创新点**的提出及实现
2. 代码的部分复现
3. 报告及 PPT 的部分制作

fourteenice：

1. 代码的部分复现
2. 报告及 PPT 的部分制作
3. **PPT 汇报**

References

[1] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. B. Aradhye, G. Anderson, G. S. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah, "Wide & deep learning for recommender systems," *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, 2016.

[2] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," 2012.

[3] T. Schnabel, A. Swaminathan, A. Singh, N. Chandak, and T. Joachims, "Recommendations as treatments: debiasing learning and evaluation," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, p. 1670–1679, JMLR.org, 2016.

[4] Q. Wan, X. He, X. Wang, J. Wu, W. Guo, and R. Tang, "Cross pairwise ranking for unbiased item recommendation," 2022.

[5] S. Rendle and C. Freudenthaler, "Improving pairwise learning for item recommendation from implicit feedback," in *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM '14, (New York, NY, USA), p. 273–282, Association for Computing Machinery, 2014.

[6] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interact. Intell. Syst.*, vol. 5, dec 2015.

[7] J. Ni, J. Li, and J. McAuley, "Justifying recommendations using distantly-labeled reviews and fine-grained aspects," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (K. Inui, J. Jiang, V. Ng, and X. Wan, eds.), (Hong Kong, China), pp. 188–197, Association for Computational Linguistics, Nov. 2019.

[8] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: user movement in location-based social networks," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, (New York, NY, USA), p. 1082–1090, Association for Computing Machinery, 2011.

[9] J. Bennett and S. Lanning, "The netflix prize," 2007.

[10] W. Chen, P. Huang, J. Xu, X. Guo, C. Guo, F. Sun, C. Li, A. Pfadler, H. Zhao, and B. Zhao, "Pog: Personalized outfit generation for fashion recommendation at

alibaba ifashion," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, (New York, NY, USA), p. 2662–2670, Association for Computing Machinery, 2019.

[11] A. Althbiti, R. Alshamrani, T. Alghamdi, S. Lee, and X. Ma, "Addressing data sparsity in collaborative filtering based recommender systems using clustering and artificial neural network," in *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 0218–0227, 2021.

[12] J. Liu, C. Xu, C. Yin, W. Wu, and Y. Song, "K-core based temporal graph convolutional network for dynamic graphs," *CoRR*, vol. abs/2003.09902, 2020.

[13] Y. Zheng, C. Gao, X. Li, X. He, Y. Li, and D. Jin, "Disentangling user interest and popularity bias for recommendation with causal embedding," *CoRR*, vol. abs/2006.11011, 2020.

[14] T. Ma and S. Yu, "De-selection bias recommendation algorithm based on propensity score estimation," *Applied Sciences*, vol. 13, no. 14, 2023.

[15] H. Yin, B. Cui, J. Li, J. Yao, and C. Chen, "Challenging the long tail recommendation," *Proc. VLDB Endow.*, vol. 5, p. 896–907, may 2012.