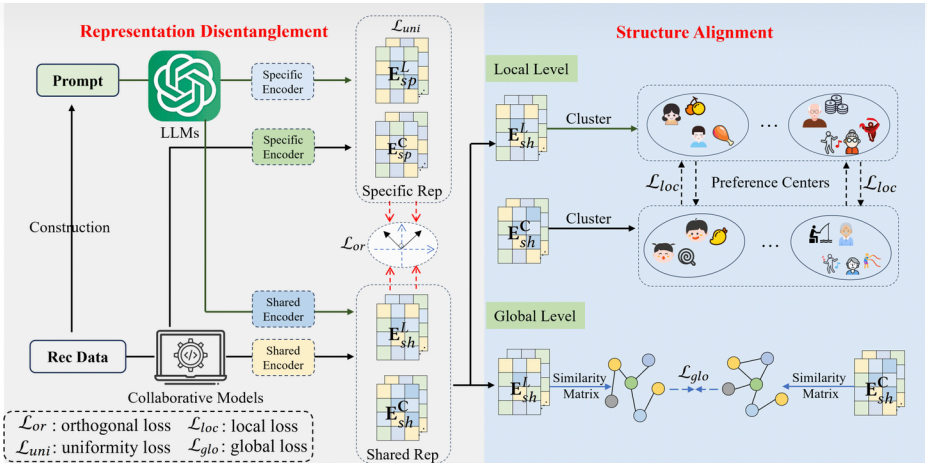
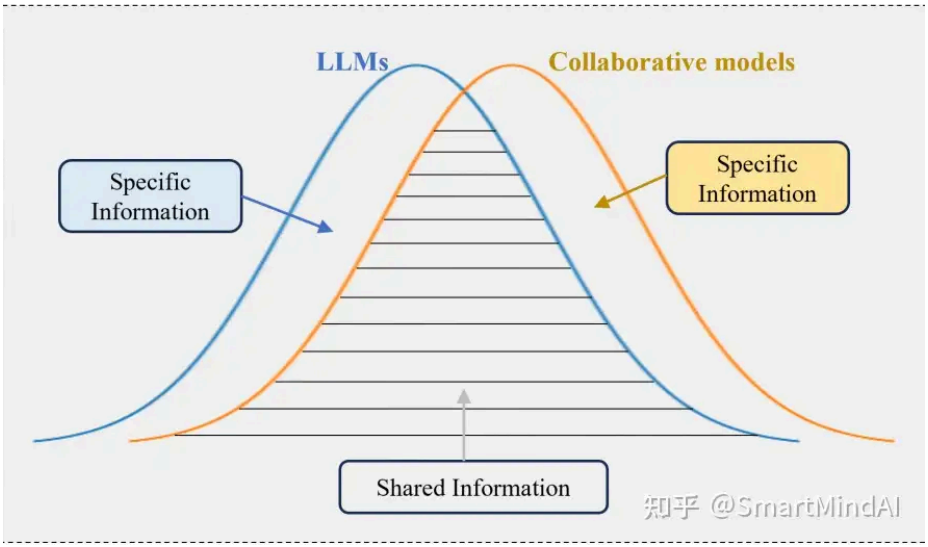


2024百度：DaRec--LLM与推荐系统无缝对齐的新框架

SmartMindAI
专注搜索、广告、推荐、大模型和人工智能最新技术，欢迎关注我
已关注

Introduction

当前已经探索了将大规模语言模型应用于推荐系统[+]的可能性，但面临两个主要限制：首先，大规模语言模型参数庞大，难以满足推荐系统的低延迟要求。其次，这些模型在预测时忽略了协作信号，仅考虑语义。因此，近期研究尝试通过语义对齐方法，将大规模语言模型的潜在表示与协作模型对齐，以提升推荐性能。然而，由于协作模型使用的交互数据与训练大规模语言模型的自然语言不同，存在显著的语义差距。



一些方法通过对比学习对齐协作模型和大规模语言模型的表示，减少差距。然而，直接在潜在空间中对齐表示可能忽略每种模态的特定信息。我们在定理中研究了理论上的表示差距，证明当差距为零时，即完全对齐协作模型和大规模语言模型的表示时，下游推荐任务的性能会受到影响。简单地将零差距的表示映射到相同的潜在空间会引入无关噪声，降低推荐性能。

因此，有效地对齐这两种模态成为一个关键问题。我们首先对理论上的表示差距进行了研究，证明了当差距为零时，即完全对齐协作模型和大规模语言模型[+]的表示时，下游推荐任务需要为性能付出代价。简单地将具有零差距的表示映射到相同的潜在空间会引入无关的噪声，导致推荐任务的性能降低。

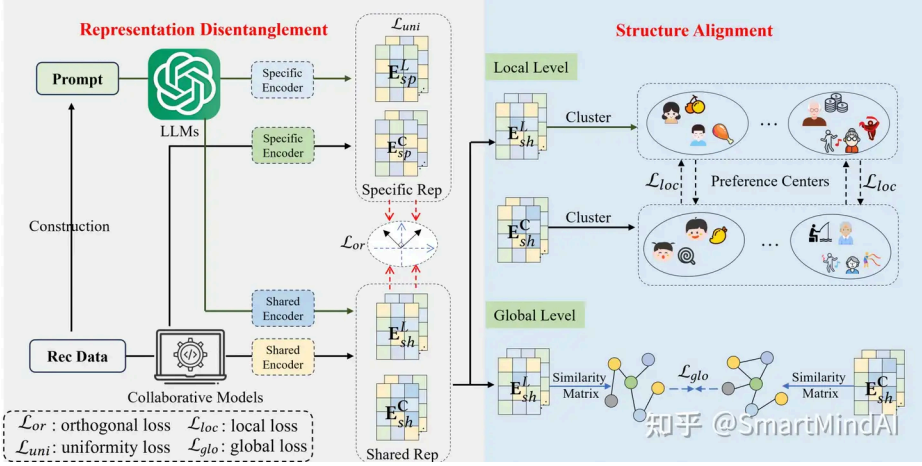
受我们的理论发现启发，我们提出了一种名为DaRec的可插拔分解对齐框架，用于将LLMs和协作模型的语义知识对齐。具体来说，我们将表示分解为共享组件和特定组件，以减少特定信息的负面影响。然后，我们设计了均匀性和正交损失，以保持表示中的有用信息。最后，我们提出了一种局

知乎

Preliminary

本文提出策略，以使协作模型与LLM的语义表示相一致。令 $f_{\mathbf{C}}(\cdot)$ 和 $f_{\mathbf{L}}(\cdot)$ 分别表示协作模型与LLM在潜在空间中的表示。此外 $\mathbf{D}$ 和 $\mathbf{D'}$ 是协作模型与LLM的两种输入类型，即评论数据与提示。我们用 Y 表示推荐任务的目标变量。h 表示预测函数[+]。LLM与协作模型的表示分别表示为 $\mathbf{E}^{\mathbf{L}}$ 与 $\mathbf{E}^{\mathbf{C}}$。此外，我们定义两个表示之间的互信息为： $I(\mathbf{E}^{\mathbf{C}};\mathbf{E}^{\mathbf{L}})$，并使用 $H(Y|\mathbf{E}^{\mathbf{C}},\mathbf{E}^{\mathbf{L}})$ 表示两个表示的条件熵。$\ell_{CE}(\cdot)$是交叉熵损失[+]。

Methodology



Motivation

虽然已经探索了LLM和CM之间的各种对齐策略，但仍然存在一个开放的问题，即在潜在空间中精确对齐语义表示是否对齐最优。直观的想法是使合作模型和LLM的语义表示之间的差距尽可能小。然而，不清楚这种对齐如何影响下游推荐任务。为了解决这个问题，我们提供了一个图示。由于数据组织、训练方法和语义特征的差异，LLM和合作模型的特征之间自然存在差距。受到这个想法的启发，我们假设在潜在空间中直接减少差距并不总是能导致更好的下游推荐任务性能。然而，理解如何减少差距是有指导意义的。为此，我们首先给出信息差异的定义：

$$\Delta p = |I(\mathbf{D};Y) - I(\mathbf{D'};Y)$$

用于描述两种模型输入对于目标标签 Y 的差异。它与编码网络 $f_{\mathbf{C}(\cdot)}$ 和 $f_{\mathbf{L}(\cdot)}$ 无关。因此，在训练过程中 Δp 是一个常数。接下来，我们将提供一个定理。它表明，如果我们尝试找到允许零差距的表示，信息差异将作为推荐任务错误的下界。因此，信息差异是精确对齐由合作模型和LLM提取的不同表示的代价。这个定理如下所示：

定理 1. 对于协作模型的编码网络$f_{\mathbf{C}(\cdot)}$和LLM的编码网络$f_{\mathbf{L}(\cdot)}$，如果表示： $\mathbf{E}^{\mathbf{C}} = f_{\mathbf{C}}(\mathbf{D})$ 和 $\mathbf{E}^{\mathbf{L}} = f_{\mathbf{L}}(\mathbf{D'})$

在潜在空间中完全对齐，即 $\mathbf{E}^{\mathbf{C}} = \mathbf{E}^{\mathbf{L}}$

我们有： $\inf_h \mathbb{E}_p[\mathcal{L}_{ce}(h(\mathbf{E}^{\mathbf{C}},\mathbf{E}^{\mathbf{L}}),Y)] - \inf_{h'} \mathbb{E}_p[\mathcal{L}_{ce}(h'(\mathbf{E}^{\mathbf{C}},\mathbf{E}^{\mathbf{L}}),Y)] \geq \Delta_p.$

理论表明，如果协作模型与LLM之间的信息差距显著，那么精确对齐表示所得到的最优推荐误差至少比从输入数据中获得的误差多Δp。此外，由于LLM和协作模型具有不同的语义场景和训练过程，每种模型都有特定的信息。进行精确对齐将引入协作模型和LLM各自特有的信息，这种特定信息可能会相互影响，导致下游推荐任务的性能降低。因此，在本文中，我们首先将协作模型和LLM的初始表示分解为特定的表示和共享的表示。然后，我们设计了一种局部和全局层面的结构对齐策略，以进行更宽松的对齐。

Representation Disentanglement

，导致下游推荐任务的性能可能不理想。受到这一直觉的启发，我们设计了一种表示分解方法，们将其分解为两个部分，即特定表示和共享表示：

$$\mathbf{E}_{sp}^{\mathbf{C}} = f_{sp}^{\mathbf{C}}(\mathbf{E}^{\mathbf{C}}), \mathbf{E}_{sh}^{\mathbf{C}} = f_{sh}^{\mathbf{C}}(\mathbf{E}^{\mathbf{C}}),$$
$$\mathbf{E}_{sp}^{L} = f_{sp}^{\mathbf{L}}(\mathbf{E}^{L}), \mathbf{E}_{sh}^{L} = f_{sh}^{\mathbf{L}}(\mathbf{E}^{L}),$$

在其中，针对特定表示的编码网络和针对共享表示的编码网络分别表示为$f_{sh}(\cdot)$和$f_{sp}(\cdot)$。我们采用多层感知器+作为$f_{sh}(\cdot)$和$f_{sp}(\cdot)$的主干网络。 为了确保特定表示和共享表示能够获得独特且互补的信息，我们旨在通过最小化以下方程对特定表示和共享表示施加正交约束：

$$\mathcal{L}_{or} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{S}(\mathbf{E}_{sp_i}^{L}, \mathbf{E}_{sh_i}^{L}))^2 + \frac{1}{N}\sum_{i=1}^{N}(\mathbf{S}(\mathbf{E}_{sp_i}^{C}, \mathbf{E}_{sh_i}^{C}))^2,$$

其中$\mathbf{S}(\cdot,\cdot)$是余弦相似度+N也就是用户和项目的数量之和，即$N = N_U + N_I$。

为了防止特定表示成为模型的无关信息，我们设计了一种方法来约束协作模型和LLMs的特定表示。这里，我们引入了均匀性损失到特定表示，该损失最大化两个变量的高斯势能。均匀性损失的计算公式为：

$$\mathcal{L}_{uni} = \log \mathop{\mathbb{E}}_{x,y\sim\mathbf{E}_{sp}^{C}} e^{-2||G(x)-G(y)||^2} + \log \mathop{\mathbb{E}}_{x,y\sim\mathbf{E}_{sp}^{L}} e^{-2||G(x)-G(y)||^2},$$

Structure Alignment

受到其他领域对齐方法的启发，本文尝试从结构视角设计对齐策略。

Global Structure Alignment.

基于共享表示的协作模型和LLMs的基础，我们设计了一个全局结构对齐策略。具体而言，我们首先计算了共享表示的相似性矩阵，其可以表示为：

$$\mathbf{S}_{C}^{G} = \mathbf{E}_{sh}^{C}(\mathbf{E}_{sh}^{C})^{\top},$$
$$\mathbf{S}_{L}^{G} = \mathbf{E}_{sh}^{L}(\mathbf{E}_{sh}^{L})^{\top},$$

我们使用矩阵乘法+计算公式获得共享表示的结构，该结构涵盖了所有成对实例，包含用户表示和项目表示的拼接，可以视为用户偏好的成对实例。

接下来，我们可以通过以下方式对齐协作模型的结构与LLMs的共享表示：

$$\mathcal{L}_{glo} = ||\mathbf{S}_{C}^{G} - \mathbf{S}_{L}^{G}||_{F}^{2}$$

Local Structure Alignment.

为了全面确保协作模型和LLMs的表示结构的一致性，与从所有共享表示的对对关系出发进行的全局结构一致性不同，局部结构是从大尺度的角度进行的。具体来说，我们尝试使用喜好来展示一致性。因此，我们首先在协作模型和具有共享表示的LLMs中收集用户的喜好。在本工作中，我们对共享表示进行聚类分析+，如下所示：

$$\mathbf{C}_{C} = f_{C}(\mathbf{E}_{sh}^{C}),$$
$$\mathbf{C}_{L} = f_{C}(\mathbf{E}_{sh}^{L}),$$

其中$f_{C}(\cdot)$是聚类函数，例如K-Means+。 $\mathbf{C}_{C} \in \mathbb{R}^{K\times d}$ 和 $\mathbf{C}_{L} \in \mathbb{R}^{K\times d}$ 分别表示协作模型和LLMs共享表示的聚类中心K表示偏好中心的数量。

通过等式我们可以在协作模型和具有不同语义场景的LLMs中获得用户的偏好。与全局结构对齐相比，聚类操作可以缩小用户和项目的数量规模。用户的偏好应该与协作模型和LLMs保持一致。然而，如何正确对齐不同的偏好中心是一个挑战，因为没有明确的目标信息。

的欧几里得距离*和第j个表示在第二个偏好集群中的欧几里得距离。对于所有偏好集群：

$$dis(\mathbf{C}_C^i, \mathbf{C}_L^j) = ||\mathbf{C}_C^i - \mathbf{C}_L^j||_2,$$

其中i和j分别等于1、2、$\ldots$、K。然后，我们将dis从小到大排序，并对$\mathbf{C}_C$和$\mathbf{C}_L$进行微调，可以表示为：

$$\begin{aligned}\text{ind} &= \text{Sort}(dis(\mathbf{C}_C^i, \mathbf{C}_L^j)),\\ \mathbf{C}_C &= \mathbf{C}_C[\text{ind}], \mathbf{C}_L = \mathbf{C}_L[\text{ind}],\end{aligned}$$

其中Sort是按升序排列的排序函数。ind表示排序偏好集群的索引。通过此操作，最相似的对中心可以调整到正确的位置。然后，我们标记排序中心，并从$\mathbf{C}$中选择未标记的向量重新计算相应的dis，直到所有偏好中心都排序。这样，协作模型和LLMs中的偏好中心大致对应。为了执行我们的局部对齐，我们计算协作模型和LLMs中不同偏好中心之间的余弦相似矩阵：

$$\mathbf{S}_{ij}^{\mathbf{C}} = \frac{(\mathbf{C}_C^i) \cdot (\mathbf{C}_L^j)}{||\mathbf{C}_C^i||_2 ||\mathbf{C}_L^j||_2}.$$

然后，我们最小化以下函数，以在局部级别对齐不同的偏好中心：

$$\mathcal{L}_{loc} = \frac{1}{K}\sum_{i=1}^{K}(\mathbf{S}_{ii}^{\mathbf{C}} - 1)^2 + \frac{1}{K^2 - K}\sum_{i=1}^{K}\sum_{i\neq j}(\mathbf{S}_{ij}^{\mathbf{C}})^2,$$

在其中K是聚类偏好数量。通过最小化等式相同的偏好中心被强制相互一致，而不同的中心则被鼓励彼此远离。

Optimization and Complexity

在本文中，我们提出了一种即插即用的框架用于这个工作，以更好地对协作模型和LLM的语义表示进行对齐。我们提出了一种即插即用的框架，通过以下函数联合优化：

$$\mathcal{L} = \mathcal{L}_{base} + \lambda(\mathcal{L}_{or} + \mathcal{L}_{uni} + \mathcal{L}_{glo} + \mathcal{L}_{loc}),$$

其中 $\mathcal{L}_{base}$ 是基线的损失函数*，例如分类损失。λ表示损失函数中的权衡参数。我们用N和d分别表示样本数量和表示的维度。对于$\mathcal{L}_{or}$中的正交操作，时间复杂度为$\mathcal{O}(Nd)$。此外$\mathcal{L}_{glo}$中相似性操作的时间复杂度为N^2d。此外，一致性损失$\mathcal{L}_{uni}$的时间复杂度为N^2d。由于偏好中心$\mathbf{C}$的维度为实数空间$\mathbb{R}^{K\times d}$ $\mathcal{L}_{loc}$的时间复杂度为K^2d。提出的损失函数的整体时间复杂度可近似为$N^2d + Nd + K^2d$。此外，提出的损失函数的空间复杂度*为$N^2 + N + K^2$。实践中，我们随机采样$\hat{N}$个实例进行近似，以减少计算和空间复杂度。综上所述，考虑到$K << \hat{N}$，我们提出的损失函数的时间和空间复杂度分别为$\hat{N}^2d + \hat{N}d$和$\hat{N}^2 + \hat{N}$。

Theoretical analysis

我们从理论视角探讨了我们提出的分离对齐框架的合理性。为了方便，我们给出以下符号。令$\widehat{\mathbf{E}}$表示我们方法的组合共享和特定表示，而$\widetilde{\mathbf{E}}$表示之前未分离方法提取的表示。我们有：

定理 2. 对于推荐任务$\mathbf{R}$，表示$\widehat{\mathbf{E}}$包含比$\widetilde{\mathbf{E}}$更多的相关信息和更少的无关信息，可以表示为：

$$I(\widehat{\mathbf{E}}^{\mathbf{D}}, \mathbf{R}) \geq I(\widetilde{\mathbf{E}}^{\mathbf{D}}, \mathbf{R}),\ H(\widehat{\mathbf{E}}^{\mathbf{D}}|\mathbf{R}) \quad \leq H(\widetilde{\mathbf{E}}^{\mathbf{D}}|\mathbf{R}),$$

其中 $I(\mathbf{E}^{\mathbf{D}}, \mathbf{R})$ 表示表示与推荐任务之间的互信息， $H(\mathbf{E}^{\mathbf{D}}|\mathbf{R})$ 表示在推荐任务条件下表示的熵。

Experiment

知乎

	Improvement	0.18%	0.78%	1.29%	0.18%	0.74%	0.95%	3.18%	4.09%	5.33%	1.27%	2.14%	3.20%	1.67%	1.93%	1.22%	1.01%	1.12%	0.90%
LightGCN	Baseline	0.057	0.0915	0.1411	0.0574	0.0694	0.0856	0.0421	0.0706	0.1157	0.0491	0.058	0.0733	0.0518	0.0852	0.1348	0.0575	0.0687	0.0855
	RLMRec-Con	0.0608	0.0969	0.1483	0.0606	0.0734	0.0903	0.0445	0.0754	0.123	0.0518	0.0614	0.0776	0.0548	0.0895	0.01421	0.0608	0.0724	0.0902
	RLMRec-Gen	0.0596	0.0948	0.1446	0.0605	0.0724	0.0887	0.0435	0.0734	0.1209	0.0505	0.06	0.0761	0.055	0.0907	0.1433	0.0607	0.0729	0.0907
	Ours	**0.0628**†	**0.0976**†	**0.1495**†	**0.0621**†	**0.0742**†	**0.091**†	**0.0461**†	**0.0759**†	**0.1246**†	**0.0537**†	**0.0625**†	**0.0789**†	**0.0558**†	**0.0917**†	**0.1456**†	**0.0609**†	**0.073**†	**0.0914**†
	Improvement	3.29%	0.72%	0.81%	2.48%	1.09%	0.78%	3.60%	0.66%	1.30%	3.67%	1.79%	1.68%	1.45%	1.10%	1.61%	0.33%	0.14%	0.77%
SGL	Baseline	0.0637	0.0994	0.1473	0.0632	0.0756	0.0913	0.0432	0.0722	0.1197	0.0501	0.0592	0.0753	0.0565	0.0919	0.1444	0.0618	0.0738	0.0917
	RLMRec-Con	0.0655	0.1017	0.1528	0.0652	0.0778	0.0945	0.0452	0.0763	0.1248	0.053	0.0626	0.079	0.0589	0.0956	0.1489	0.0645	0.0768	0.095
	RLMRec-Gen	0.0644	0.1015	0.1537	0.0648	0.0777	0.0947	0.0467	0.0771	0.1263	0.0537	0.0631	0.0798	0.0574	0.094	0.1476	0.0629	0.0752	0.0934
	Ours	**0.0667**†	**0.102**†	**0.1536**†	**0.0662**†	**0.0785**†	**0.0952**†	**0.0471**†	**0.0785**†	**0.1284**†	**0.0545**†	**0.064**†	**0.081**†	**0.0599**†	**0.0968**†	**0.15**†	**0.0655**†	**0.0778**†	**0.0958**†
	Improvement	1.83%	0.29%	0.52%	1.53%	0.90%	0.74%	1.06%	1.82%	1.66%	1.49%	1.43%	1.50%	1.70%	1.26%	0.74%	1.55%	1.30%	0.84%
SimGCL	Baseline	0.0618	0.0992	0.1512	0.0619	0.0749	0.0919	0.0467	0.0772	0.1254	0.0546	0.0638	0.0801	0.0564	0.0918	0.1436	0.0618	0.0738	0.0915
	RLMRec-Con	0.0633	0.1011	0.1552	0.0633	0.0765	0.0942	0.047	0.0784	0.1292	0.0546	0.0642	0.0814	0.0582	0.0945	0.1482	0.0638	0.076	0.0942
	RLMRec-Gen	0.0617	0.0991	0.1524	0.0622	0.0752	0.0925	0.0464	0.0767	0.1267	0.0541	0.0634	0.0803	0.0572	0.0929	0.1456	0.0627	0.0747	0.0926
	Ours	**0.0648**†	**0.103**†	**0.1563**†	**0.0651**†	**0.0781**†	**0.0954**†	**0.0479**†	**0.0804**†	**0.1317**†	**0.0553**†	**0.0656**†	**0.0831**†	**0.0588**†	**0.095**†	**0.1497**†	**0.0642**†	**0.0762**†	**0.0947**†
	Improvement	2.37%	1.88%	0.71%	2.84%	2.09%	1.27%	1.91%	2.55%	1.93%	1.28%	2.18%	2.09%	1.03%	0.53%	1.01%	0.63%	0.26%	0.53%
DCCF	Baseline	0.0662	0.1019	0.1517	0.0658	0.078	0.0943	0.0468	0.0778	0.1249	0.0543	0.064	0.08	0.0561	0.0915	0.1437	0.0618	0.0736	0.0914
	RLMRec-Con	0.0665	0.104	0.1563	0.0668	0.0798	0.0968	0.0486	0.0813	0.1321	0.0561	0.0663	0.0836	0.0572	0.0929	0.1459	0.0627	0.0747	0.0927
	RLMRec-Gen	0.0666	0.1046	0.1559	0.067	0.0801	0.0969	0.0475	0.0785	0.1281	0.0549	0.0646	0.0815	0.057	0.0918	0.143	0.0625	0.0741	0.0915
	Ours	**0.0677**†	0.1045	**0.1582**†	**0.0674**†	**0.0807**†	**0.0981**†	**0.0495**†	**0.0826**†	**0.1352**†	**0.0569**†	**0.0673**†	**0.0850**†	**0.0586**†	**0.0938**†	**0.1479**†	**0.0638**†	**0.0751**†	**0.0937**†
	Improvement	1.65%	-0.10%	1.48%	0.60%	0.75%	1.24%	1.85%	1.60%	2.35%	1.43%	1.51%	1.67%	2.45%	0.97%	1.37%	1.75%	0.54%	1.08%
AutoCF	Baseline	0.0689	0.1055	0.1536	0.0705	0.0828	0.0984	0.0469	0.0789	0.128	0.0547	0.0647	0.0813	0.0519	0.0853	0.1358	0.0572	0.0684	0.0855
	RLMRec-Con	0.0695	0.1083	0.1586	0.0704	0.0837	0.1001	0.0488	0.0814	0.1319	0.0562	0.0663	0.0835	0.05	0.0870	0.1372	0.0578	0.0679	0.0872
	RLMRec-Gen	0.0693	0.1069	0.1581	0.0701	0.083	0.0996	0.0493	0.0828	0.133	0.0572	0.0677	0.0848	0.051	0.0868	0.1318	0.058	0.0681	0.0886
	Ours	**0.0714**†	**0.1102**†	**0.159**†	**0.0725**†	**0.0856**†	**0.1016**†	**0.0512**†	**0.0841**†	**0.1344**†	**0.059**†	**0.0691**†	**0.0861**†	**0.0554**†	**0.0900**†	**0.1422**†	**0.0604**†	**0.0719**†	**0.0895**†
	Improvement	2.73%	1.75%	0.25%	2.98%	2.27%	1.50%	3.85%	1.57%	1.05%	3.15%	2.07%	1.53%	2.59%	1.35%	0.85%	1.85%	1.27%	1.02%

为了展示我们提出的DaRec的有效性和优越性，在本节中，我们使用九个最先进的基线算法，在三个数据集上进行实验，共计六个指标。比较的算法大致可以分为两类，即传统的协同过滤✦方法（GCCF，LightGCN，SGL，SimGCL，DCCF，AutoCF），以及LLMs增强的推荐方法（RLMRec-Con，RLMRec-Gene，KAR）。其中，RLMRec-Con和RLMRec-Gene表示RLMRec中的两种方法。

在设计这个插件式的解耦框架时，我们旨在通过更好地对齐协同模型和LLMs，实现更好的协同推荐效果。实验结果表明，DaRec在六个评估指标上均优于其他八个基线算法。具体而言，DaRec在准确率、召回率、NDCG等关键指标上均取得了显著优势，特别是在处理长尾项目和稀疏用户行为方面，DaRec展现出更强的泛化能力和鲁棒性✦。

通过对比实验，我们可以发现，DaRec不仅在整体性能上超越了其他推荐算法✦，而且在处理特定场景（如冷启动问题和新颖项目推荐）时，其表现更加出色。这得益于我们设计的解耦框架，它能够有效整合协同过滤的精确性和基于内容的推荐的多样性，从而实现更全面、更个性化的推荐。

- 与传统的协同过滤算法✦（如GCCF、LightGCN、SGL、SimGCL、DCCF、AutoCF）相比，我们提出的DaRec在推荐性能方面表现出色。我们分析原因在于，通过LLMs提升表示，赋予表示更多的语义信息，从而增强了表示的语义内容，进而提高了推荐性能。
- 语言模型增强的推荐方法（RLMRec以及KAR）在推荐性能上相比我们提出的算法，达到了次优的推荐性能。我们推测，通过我们的分解对齐策略，我们可以更好地对协作模型和LLMs进行对齐。
- 我们的提出的DaRec在六个指标下，在三个数据集上都超越了其他推荐方法。以Yelp数据集上的AutoCF结果为例，通过我们的即插即用框架，DaRec改进了AutoCF，使其在R@5、R@10、N@5和N@10的指标上分别超越第二好的推荐方法3.85%、1.57%、3.15%、2.07%。

原文《DaRec: A Disentangled Alignment Framework for Large Language Model and Recommender System》

发布于 2024-09-10 14:17 · IP 属地北京

LLM 推荐系统 百度

理性发言，友善互动

还没有评论，发表第一个评论吧