MPDA 的复现与改进

摘要

在端云协同推荐系统中，云端推荐系统基于全局数据训练，但用户本地数据分布不同，缺乏个性化推荐能力。纯端侧推荐因数据稀缺容易过拟合，影响性能。为解决这一问题，原作者提出了端云协同推荐框架 MPDA [5]。MPDA [5] 通过云端全局池为用户检索相似数据缓解数据稀缺和过拟合问题。系统采用用户特征向量进行相似用户匹配，实验表明随机匹配效果优于基于交互物品记录的匹配。然而，交互记录不足以充分表示用户特征，因此我提出了两种新的特征表示方法：1）使用用户交互物品的平均嵌入表示特征；2）基于云端召回物品对相似度计算特征。这些改进提升了推荐性能，并为端云协同推荐系统的特征表示研究提供了新思路。

关键词：推荐系统；端云推荐；数据增强

1 引言

随着推荐系统在各类应用中的普及，端云协同推荐系统逐渐成为研究热点。在传统云端推荐系统中，通过全局数据进行训练，能够提供总体较优的推荐效果，但由于忽略了用户本地数据的差异性，其个性化能力较弱，难以满足用户的个性化需求。另一方面，纯端侧推荐虽然能够依据用户本地数据进行个性化建模，但因数据稀缺和用户交互数据有限，容易导致过拟合，进而影响推荐性能。这种端侧和云端模型各自的局限性，促使研究者探索端云协同推荐框架，以期融合两者的优势。

在现有的端云协同推荐框架中，云端系统通常基于全局数据训练推荐模型，同时通过检索相似用户数据缓解端侧数据稀缺问题。已有研究表明，使用用户特征向量进行相似用户匹配的方式能够提升推荐性能。然而，大多数方法基于用户的交互物品记录进行特征匹配，忽略了用户交互记录可能不足以全面表达其兴趣分布的局限性。此外，实验结果显示，随机匹配甚至可能优于简单的基于交互记录的匹配策略。这表明，设计更有效的用户特征表示方式是端云协同推荐系统中的关键问题之一。为此，我提出了两种新的用户特征表示方法：1. 基于用户交互物品的平均嵌入：通过对用户交互的物品嵌入向量取均值，构建更具代表性的特征表示。2. 基于云端召回物品的相似度计算：利用云端全局数据召回的物品与用户交互历史的相似性，进一步提升用户特征的表达能力。

该研究通过优化用户特征表示方法，为端云协同推荐系统提供了新的思路和技术改进：1. 提升推荐性能：改进后的特征表示能够更全面地反映用户兴趣分布，从而提高相似用户匹配的精度，缓解端侧数据稀缺问题，有效提升推荐结果的准确性和个性化水平。2. 增强端云协作的适配性：通过利用云端召回物品与端侧数据的交互关系，充分挖掘端云协同的潜力，探索

更加合理的协同机制，为实现高效、个性化的推荐提供了理论依据和实践指导。3. 推动特征表示研究：所提出的特征表示方法为端云协同推荐系统在特征建模方面提供了新的思路，具有一定的理论研究价值和实际应用意义，能够为后续相关研究提供参考和借鉴。

2 相关工作

在本节中，我们将回顾推荐系统、端云协作学习、数据增强的相关工作。我们还会阐明 MPDA [5] 与现有工作的关键差异。

2.1 推荐系统

推荐系统是当今信息服务的重要组成部分，广泛应用于电子商务、社交媒体、内容分发和个性化服务等领域，其核心目标是根据用户的历史行为和兴趣偏好，为用户推荐相关性更高的内容或产品。传统推荐方法主要分为三类：基于协同过滤的方法、基于内容的方法以及基于深度学习的方法。协同过滤利用用户与用户之间或物品与物品之间的相似性进行推荐，具有简单易用和可扩展的特点，但其在数据稀疏和冷启动场景下性能受限。基于内容的方法通过分析用户和物品的特征，建立特征匹配模型，能够提供一定程度的个性化推荐，但容易导致过度个性化问题，即用户仅接收到与其历史兴趣高度相似的推荐内容。

近年来，深度学习技术的快速发展为推荐系统开辟了新的方向。通过引入序列模型、图神经网络和自监督学习等技术，深度学习推荐系统不仅能够捕捉用户复杂的行为模式，还能从高维、非线性的数据中挖掘潜在的兴趣关系。经典的深度学习推荐模型包括 Factorization Machine (FM) [4]、WideDeep [1]、DIN [6] 和 DeepFM [2]。FM 模型通过因子分解技术有效捕捉特征之间的二阶交互信息，在大规模稀疏数据的推荐任务中表现优异。FM 的优点在于其能够通过低秩矩阵分解来学习特征之间的交互关系，这对于用户行为数据尤其重要。WideDeep 模型结合了传统的广度模型（如线性模型）和深度神经网络，通过这两种模型的优势互补，可以在提供较强泛化能力的同时也能学习到复杂的非线性特征交互。此模型的设计能够处理两类不同类型的信息：宽部分负责捕捉大规模的浅层信息，而深部分则负责深入学习复杂的特征表示。DIN (Deep Interest Network) 则通过引入用户历史点击的序列信息，特别是基于用户兴趣的关注机制，改善了推荐的个性化程度，能够捕捉用户在历史行为中的动态兴趣变化，进而提高推荐的准确性。DeepFM 模型将 FM 和深度神经网络相结合，通过共享相同的输入层，能够同时学习特征的低阶和高阶交互，从而获得更强大的特征表示能力，广泛应用于广告点击率预测等场景。尽管深度学习极大地提升了推荐系统的性能，但其高计算成本和对大规模数据的依赖使得模型在资源受限的设备上难以部署。

为解决传统云端推荐系统个性化不足和端侧推荐系统易过拟合的问题，端云协同推荐系统逐渐成为研究热点。这种框架结合了云端全局信息的广度与端侧本地数据的深度，通过云端训练全局模型和端侧优化个性化推荐模型，既能有效缓解数据稀疏问题，又能提升用户体验。其中，基于用户特征向量的相似性匹配策略被广泛采用，但传统匹配方法多依赖于用户的交互物品记录，可能无法充分表达用户特征，尤其是在交互数据稀缺的情况下。为此，研究者逐渐探索更精细的特征表示方法，例如结合用户历史行为和嵌入向量的混合表示，以及通过云端召回物品对的相似度计算来增强特征表达能力。

总体来看，推荐系统的发展正向多样化、智能化和个性化方向迈进，尤其是端云协同推

荐系统通过整合端云两侧的优势，为解决个性化与数据稀缺问题提供了新的研究方向和应用场景。

2.2 端云推荐

端云协同推荐系统是近年来推荐系统领域的重要研究方向，旨在结合云端的全局数据优势和端侧的个性化能力，为用户提供更加精准和高效的推荐服务。传统的云端推荐系统通过基于全局数据训练模型，能够捕获广泛的用户行为模式，但由于用户的兴趣分布多样化，单一的云端模型在个性化推荐方面存在局限性，难以适应不同用户的特定需求。另一方面，纯端侧推荐系统利用用户本地数据进行个性化建模，但由于数据规模有限，容易出现过拟合问题，从而影响推荐性能。

端云协同推荐系统通过在云端构建全局模型，并结合端侧本地数据进行优化，实现了全局数据和本地个性化的有机结合。具体而言，云端利用全局数据训练一个通用的推荐模型，同时为端侧提供一个用户特征池或相似用户池，以增强本地数据的表达能力；端侧则利用本地数据对模型进行微调，以生成更加个性化的推荐结果。一些研究工作通过基于用户特征的相似性匹配，从云端池中为用户检索相似数据，从而缓解数据稀缺问题。然而，传统的基于交互物品记录的匹配策略可能不足以充分表达用户特征，尤其在交互数据较少的情况下，这一问题更加明显。

端云协同推荐系统通过将云端的全局视野与端侧的个性化能力相结合，不仅有效解决了传统推荐系统的局限性，还为个性化推荐技术的发展提供了新的研究方向和实践场景。

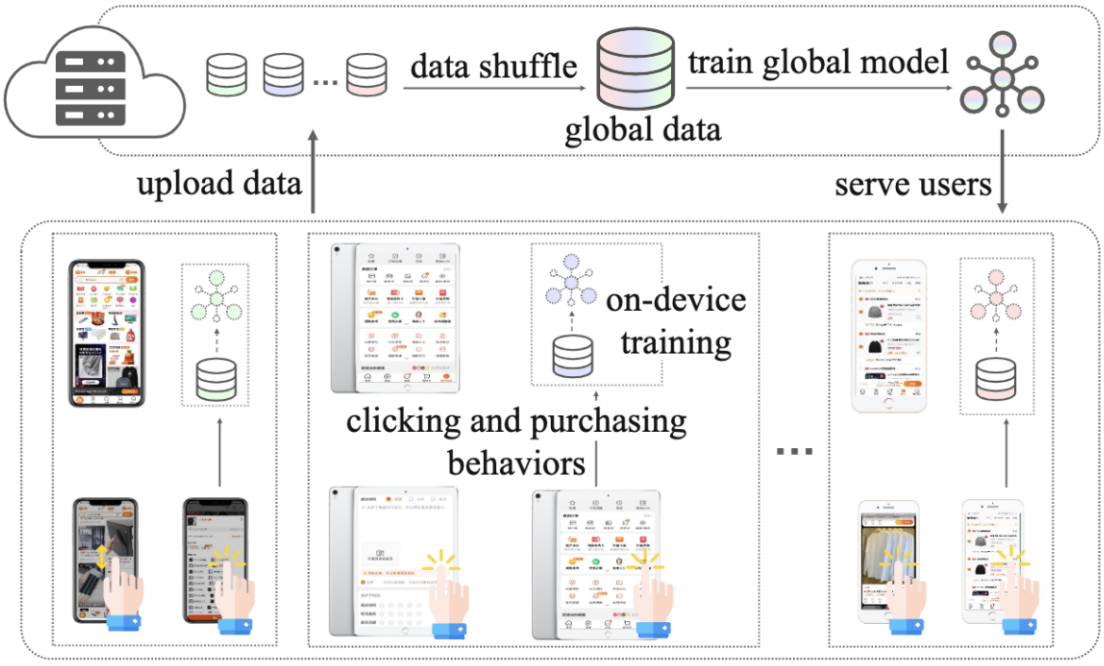

图 1. 端云推荐示意图

2.3 数据增强

端侧数据增强技术是推荐系统中针对数据稀缺和个性化需求的重要研究方向，尤其在端云协同推荐系统中发挥了关键作用。传统推荐系统依赖云端的大规模用户交互数据进行模型

训练，而端侧环境通常面临数据稀疏、分布不均的问题，导致推荐性能受限。为解决这些挑战，端侧数据增强技术通过生成、扩展或优化本地数据，提升模型的泛化能力和个性化表现。

一种常见的端侧数据增强方法是基于数据生成模型（如生成对抗网络 GAN 或变分自编码器 VAE），通过学习用户行为模式来生成与真实交互数据相似的合成数据。这些合成数据不仅可以丰富用户的历史行为序列，还能弥补冷启动用户数据不足的问题。此外，数据扰动技术也被广泛应用，即通过对用户历史行为或特征添加噪声、随机采样或特定变换，生成多样化的数据实例，从而增强模型的鲁棒性。

在端云协同推荐系统中，数据增强技术通常结合云端全局信息进行优化。例如，云端可以基于全局用户行为构建一个共享的特征或样本池，端侧通过从该池中检索相似用户或物品的交互数据，来补充本地数据的不足。此外，用户特征表示的优化也在数据增强中扮演重要角色。一些方法利用用户交互物品的平均嵌入或基于召回物品的相似性构建用户特征，这种增强特征的方法不仅改善了数据表示的丰富性，还提高了个性化推荐的效果。

端侧数据增强技术的优势在于能够充分利用本地计算资源，在保护用户隐私的同时提升推荐系统的性能。这种方法尤其适用于个性化推荐需求强烈且数据分布不均的场景。未来，端侧数据增强技术的发展可能会进一步结合联邦学习、多模态数据融合等前沿技术，为推荐系统的创新与优化提供更大的潜力。

3 本文方法

3.1 本文方法概述

增强数据的选择原则是从云端选取一个用户子集，使端侧模型在该子集上训练后性能指标提升最大。然而，遍历所有可能的用户子集以测试其效果显然不切实际。因此，在 MPDA [5] 中将选择增强数据的过程分为两步：云端的相似用户匹配和端侧的增量训练。在效率和效果之间取得了良好的平衡。算法流程如图 2 所示。

Algorithm 2: Efficient Version of MPDA by Approximation

Input: All the users' data with the empirical distributions $\hat{\mathcal{D}}_0, \hat{\mathcal{D}}_1, \cdots, \hat{\mathcal{D}}_n$; the initial model h.

foreach *user* $i \in \{0, 1, \cdots, n\}$ *in parallel* **do**
  The cloud matches k other users $\{s_1, \cdots, s_k\}$ for user i and delivers these matched users' data to user i;
  foreach *candidate user* $s \in \{s_1, \cdots, s_k\}$ **do**
    User i trains h on user s's data for one epoch and gets $\tilde{h}$;
    User i evaluates h and $\tilde{h}$ on the local data and keeps the better model $h \leftarrow \min_{h' \in \{h, \tilde{h}\}} L_{\hat{\mathcal{D}}_i}(h')$;
  end
  User i trains h on the local data for one epoch and returns the final model h^+;
end

图 2. MPDA 算法流程

MPDA [5] 的官方代码中实现了两种相似用户匹配算法：一种是随机匹配相似用户，一种是使用用户交互物品记录做 KNN 匹配，即交互物品交集越多的用户相似度越高。从我跑出来的结果来看，随机匹配算法比使用用户交互物品记录的效果好。分析表明，使用交互物品记录作为用户特征向量的表示方法存在局限性，因此改进的重点在于优化用户特征向量的表示方式，以用于后续的相似用户匹配。为此，我使用了两种新的思路来表示用户特征向量：1. 使用用户交互物品的平均嵌入表示用户特征向量。2. 使用云端召回物品对的相似度表示用户特征向量。整体框架流程图如图 3 所示。

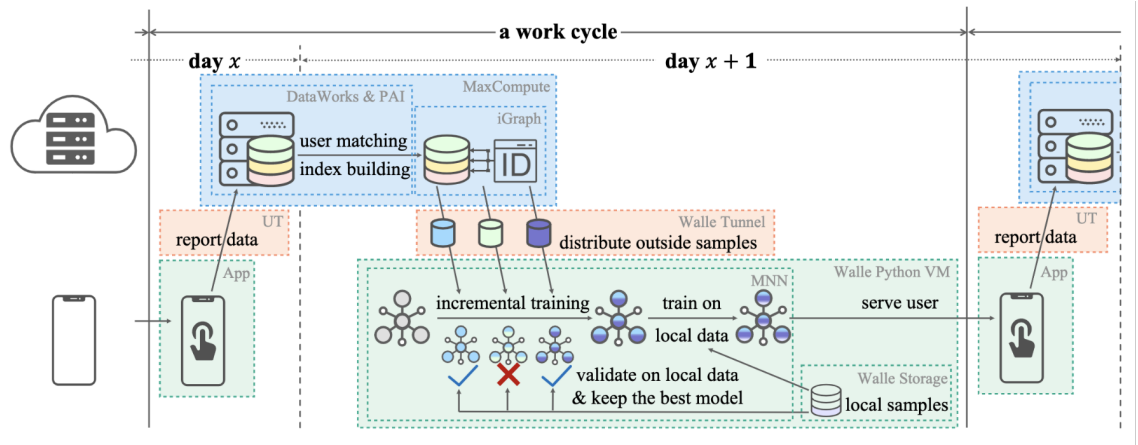

图 3. MPDA 流程图

3.2 用户交互物品平均嵌入

首先，在云端根据全局数据训练一个特征提取器。随后将这个特征提取器分发给端侧，端侧使用这个特征提取器输出自己所有交互物品的特征嵌入，随后对所有交互物品的嵌入取平均来表示用户特征向量。用户上传这个特征向量给云端做相似用户匹配。且实验结果可知对比随机匹配的效果好。方法示意图如下所示。

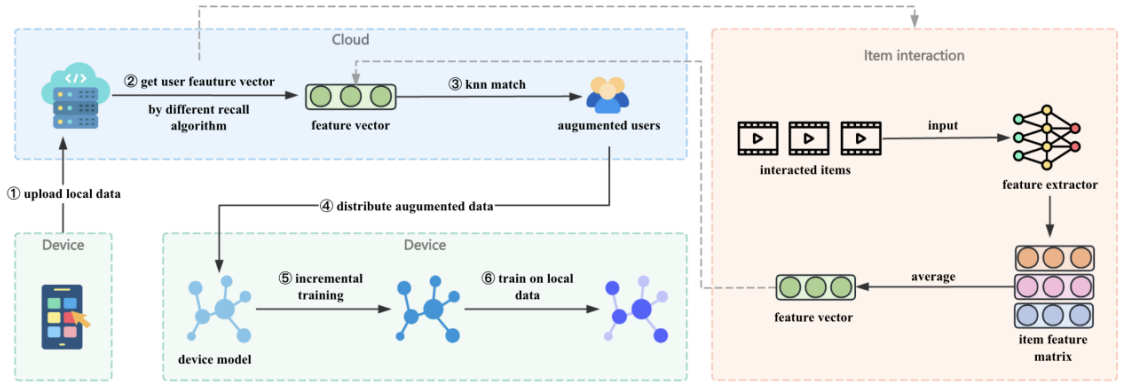

图 4. 用户交互物品平均嵌入方法示意图

3.3 云端召回物品对相似度

根据理论基础可知：如果两个用户对某些物品具有相似的兴趣或偏好，那么他们对这些物品对的嵌入相似度也应接近。因此，我们在云端首先根据物品流行度选取 50 个物品对，将这些物品对以及预训练好的全局模型分发给每个用户。用户在本地数据上微调全局模型后，遍历每个物品对，输出每个物品对中两个物品的嵌入并求一个余弦相似度。之后得到一个 50 纬度的相似度向量来表示用户特征向量。用户上传这个特征向量给云端做相似用户匹配。实验结果可知对比随机匹配的效果好。方法示意图如下所示：

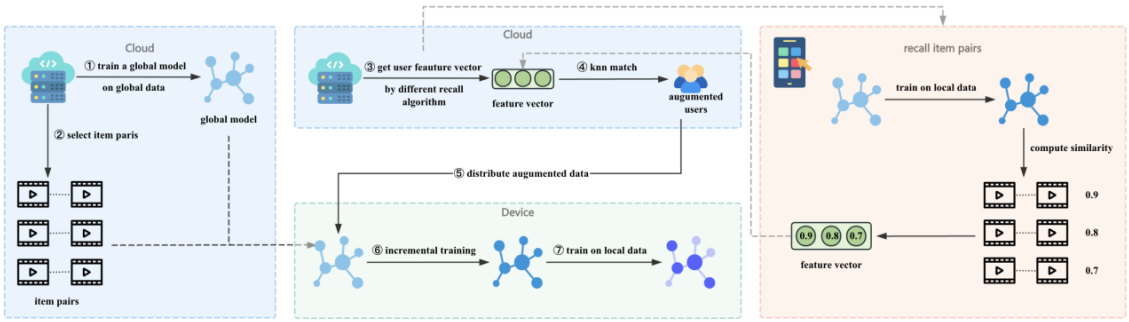

图 5. 云端召回物品对相似度方法示意图

4 复现细节

4.1 论文源代码

我们首先下载并使用了官方提供的源代码，对其中的一些问题进行了修正，以确保项目能够正常运行。在修复完成后，我们成功复现了论文中表格中的实验结果。实验数据显示，我

们基于官方源代码运行所得的结果与论文中报告的数据高度一致，这充分验证了原文工作的可复现性。复现结果如下图所示。

Dataset	Model	召回算法	k	Cloud	Local	Local+	MPDA-	MPDA
MovieLens	LR	随机	50	0.603355759	0.603289584	0.60332231	0.6031517	0.603169202
MovieLens	Wide & Depp	随机	50	0.628104515	0.632570351	0.633093062	0.63139523	0.634901454
MovieLens	DeepFM	随机	50	0.638464027	0.641007838	0.641240009	0.640625229	0.641983215
MovieLens	PNN	随机	50	0.646265901	0.649125267	0.64958286	0.647648318	0.649638931
MovieLens	DIN	随机	50	0.660489912	0.662496904	0.662578101	0.66181534	0.662710483

图 6. 官方代码复现结果

4.2 与已有开源代码对比

由于官方源代码基于低版本的 TensorFlow 框架实现，并且代码之间的耦合度较高，这种设计限制了后续功能扩展和优化的灵活性。因此，我们基于 PyTorch 框架对原文进行了完整的重写与复现，以充分发挥 PyTorch 在开发效率和扩展性方面的优势。在实现过程中，我们尽量保持项目的目录结构与官方代码一致，以便于对比和验证。同时，参考并借鉴了官方源码中的一些方法和实现细节，以确保核心算法逻辑与原始设计一致。与此同时，我们针对代码中的高耦合部分进行了优化与重构，显著降低了模块间的依赖程度，从而提升了代码的可读性和可维护性。

通过这些改进，我们不仅成功复现了原文的核心实验结果，还为后续的功能扩展和算法研究打下了更加稳固的基础。

5 实验结果分析

实验结果表明，我们提出的两种方法均显著优于原论文中的随机召回方法。在使用用户交互物品平均嵌入作为特征表示的策略下，模型的推荐性能得到了有效提升；同时，基于云端召回物品对相似度计算的特征表示进一步提高了模型对用户兴趣的刻画能力，实验结果均表现出较好的效果。此外，为进一步优化用户交互物品平均嵌入方法的表现，我们引入了多种随机掩码策略，对用户交互物品序列进行去燥处理。这些策略有效缓解了噪声数据对用户特征表示的干扰，从而增强了模型对用户真实兴趣的捕捉能力。实验分析显示，这些改进进一步提升了推荐效果，验证了我们方法的鲁棒性和实用价值。

Dataset	Model	recall algorithm	recall_num	Cloud	Local	Local+	MPDA-	MPDA
MovieLens	NCF	random	100	0.613967506	0.614197483	0.613560645	0.617964998	0.615823049
MovieLens	NCF	item_interaction	100	0.613967506	0.614197483	0.613519473	0.622882911	0.619525598
MovieLens	NCF	recall_item_pairs	100	0.613967506	0.614197483	0.624121063	0.635449435	0.628703699
MovieLens	NCF	item_interaction_with_random_mask_10%	100	0.613967506	0.614197483	0.614382133	0.622773544	0.61942046
MovieLens	NCF	item_interaction_with_random_mask_20%	100	0.613967506	0.614197483	0.613853481	0.622605895	0.619306921
MovieLens	NCF	item_interaction_with_single_mask	100	0.613967506	0.614197483	0.613964777	0.622993987	0.61958743
MovieLens	NCF	item_interaction_with_double_mask	100	0.613967506	0.614197483	0.614315695	0.622727436	0.619245414
MovieLens	NCF	item_interaction_with_triple_mask	100	0.613967506	0.614197483	0.613951752	0.622851082	0.619469465
MovieLens	NCF	item_interaction_with_hypernet	100	0.613967506	0.614197483	0.613963053	0.622316111	0.619088942

图 7. 实验结果示意

此外，我们的复现结果中使用的 NCF [3] 模型的框架如下图所示。

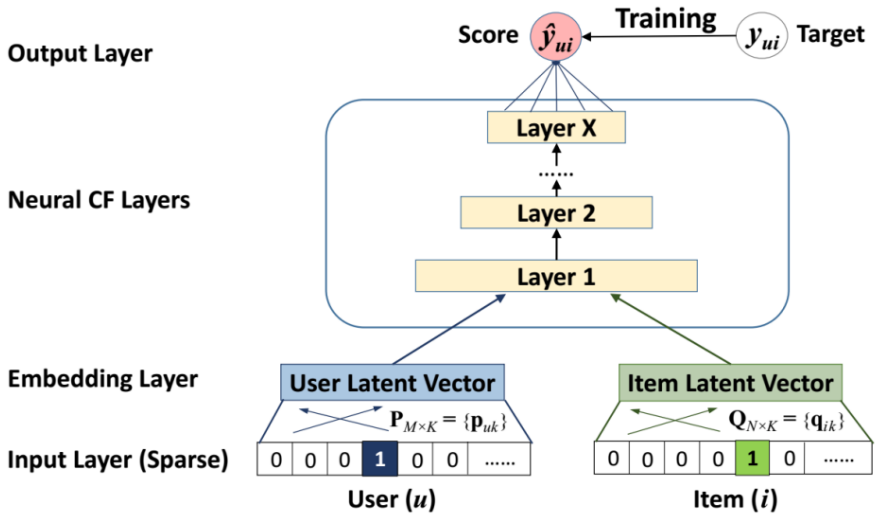

Figure 2: Neural collaborative filtering framework

图 8. 实验结果示意

6 总结与展望

在上述使用交互物品平均嵌入的方法中，我们尝试随机掩码掉部分交互物品，发现模型性能有所提升。这表明用户的交互物品中可能存在噪声数据，影响了用户特征嵌入的准确性。因此，可以对交互物品进行去噪处理以获得更精确的用户特征表示。为此，我们提出预训练一个掩码网络，在计算用户特征向量时，先利用该掩码网络对交互物品进行去噪处理，从而提高特征表示的质量。

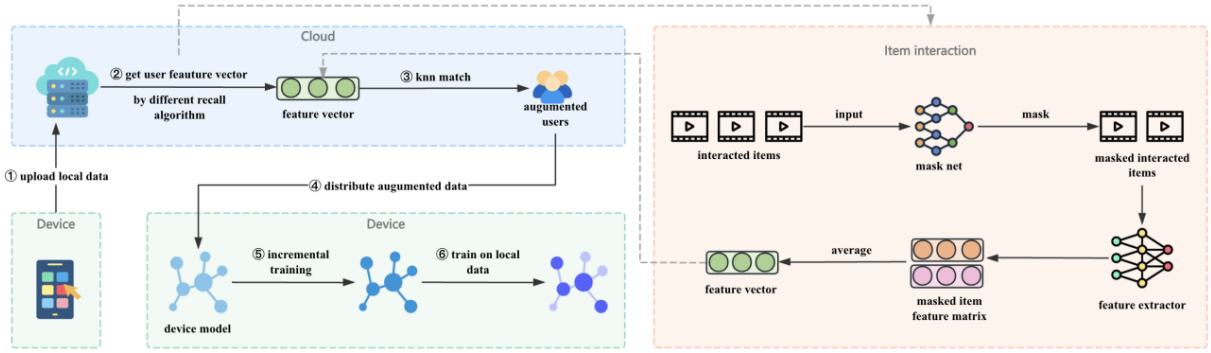

图 9. 掩码网络示意图

参考文献

[1] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. Wide & Deep Learning for Recommender Systems. 2016.

[2] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv:1703.04247*, 2017.

[3] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural Collaborative Filtering. 2017.

[4] Steffen Rendle. Factorization Machines. 2010.

[5] Yikai Yan, Chaoyue Niu, Renjie Gu, Fan Wu, Shaojie Tang, Lifeng Hua, Chengfei Lyu, and Guihai Chen. On-Device Learning for Model Personalization with Large-Scale Cloud-Coordinated Domain Adaption. 2022.

[6] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. Deep Interest Network for Click-Through Rate Prediction. 2018.