面向推荐系统的结构感知增量学习与个性化模仿权重

Yuening Wang[1], **Yingxue Zhang**[1], **Antonios Valkanas**[2*], **Ruiming Tang**[1], **Chen Ma**[3], **Jianye Hao**[1, 4], **Mark Coates**[2]

[1] 华为诺亚方舟实验室, [2] 麦吉尔大学, [3] 香港城市大学, [4] 天津大学, yuening.wang@huawei.com, yingxue.zhang@huawei.com, antonios.valkanas@mail.mcgill.ca, tangruiming@huawei.com, chenma@cityu.edu.hk, haojianye@huawei.com, mark.coates@mcgill.ca

arXiv:2305.01204v1 [cs.IR] 2 May 2023

摘要

推荐系统如今处理海量数据，在提升用户体验方面发挥着重要作用。图神经网络（GNN）因能够对丰富的关系信息进行建模，已成为效果最出色的推荐系统模型之一。不断增长的数据量可能导致图神经网络的训练成本高得难以承受。为解决这一问题，先前的研究提出在新的数据块到达时，以增量方式训练图神经网络模型。研究人员探索了特征和结构知识蒸馏技术，使图神经网络模型能够快速进行增量训练，同时缓解灾难性遗忘问题。然而，对所有用户保留相同数量的历史信息并非最优，因为这没有考虑到每位用户偏好变化的动态性。对于兴趣发生显著转变的用户，保留过多旧知识可能会过度限制模型，使其无法快速适应用户的新兴趣。相比之下，对于偏好稳定的用户，尽可能多地保留其长期偏好，则能大幅提升模型性能。在这项工作中，我们提出了一种新颖的训练策略，能够自适应地为每位用户学习个性化的模仿权重，从而平衡近期数据的贡献与从先前时间段蒸馏的知识量。我们在五个不同的数据集上，对三种基于结构蒸馏的先进推荐系统进行比较，展示了学习模仿权重的有效性。结果表明，该方法的性能始终优于具有竞争力的增量学习技术。

引言

在线服务的增长使推荐系统成为向用户提供个性化推荐的重要组成部分。提供高度相关的推荐能够改善用户体验，并增加服务提供商的收入。由于深度学习模型能够以端到端方式构建高质量的用户和物品表示，因此它们在推荐系统设计的各个方面都日益普及（Covington、Adams 和 Sargin，2016年；Guo 等，2017年；Cheng 等，2016年）。近期的一种趋势是将推荐问题表述为一个由于图能够建模丰富的关系信息，因此在图上的学习任务具有重要意义。推荐系统中的许多数据都可以自然地用图结构表示（van den Berg、Kipf 和 Welling 2017年；Wang 等，2019b、2021年；Sun 等，2019）。例如，我们可以使用用户-物品二部交互图、物品相似度图、从社交网络互动中构建的用户-用户图，以及额外的知识图谱来改进表示学习过程。基于图神经网络（GNN）的推荐系统已成为最有效的模型之一，因为消息传递范式能够充分建模数据中的关系信息。然而，在大规模图上训练 GNN 的成本可能高得难以承受（Ying 等，2018年；Zou 等，2019；Chiang 等，2019；Zeng 等，2020年；Qiu 等，2020年；Xu 等，2020），这使得在大规模推荐系统中部署采用 GNN 骨干网络的模型极具挑战性，尤其是在线系统需要满足严格的时间限制。

解决计算问题的一种方法，是随着新数据块的到来，增量式地训练深度学习模型（Kirkpatrick 等，2017年；Shmelkov、Schmid 和 Alahari，2017年；Castro 等，2018年；Rebuffi 等，2017年；Mallya 和 Lazebnik，2018年；Xu 和 Zhu，2018年；Qiu 等，2020年）。然而，直接使用增量块中的数据微调模型，可能会导致灾难性遗忘（Kirkpatrick 等，2017年；Shmelkov、Schmid 和 Alahari，2017年）。鉴于知识蒸馏方法在效率和性能方面的优势，这类方法更适用于采用图神经网络骨干架构的模型（Castro 等，2018年；Kirkpatrick 等，2017年；Xu 等，2020年；Wang、Zhang 和 Coates，2021）。得益于知识蒸馏范式，历史数据中的关键信息得以保留，并传递给仅使用新到数据训练的学生模型。其实现方式是对教师模型与学生模型表示之间的距离进行正则化。研究者已探索特征知识蒸馏和结构知识蒸馏技术；这些技术使图神经网络模型能够更好地保留先前训练数据中的节点特征和结构信息，同时享受快速增量训练过程带来的优势（Yang 等，2020年；Xu 等，2020年；Wang、Zhang 和 Coates，2021）。

然而，对所有用户一视同仁地保留相同数量的历史信息

*在华为诺亚方舟实验室实习期间完成。版权所有 © 2023年，美国人工智能促进协会 (www.aaai.org)。保留所有权利。

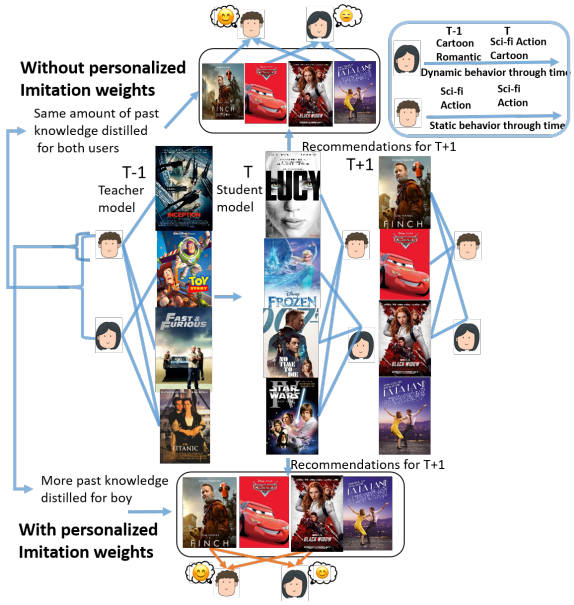

图1：该框架设计动机示意图：男孩在时间 $t-1$ 到 t 之间始终保持着相同的偏好，而女孩的兴趣变化则更加动态。通过个性化增量学习方案捕捉用户之间的兴趣偏移差异，将大有裨益。

这可能并非最优方案，因为它未能考虑每位用户的偏好变化动态。对于兴趣发生显著变化的用户，通过知识蒸馏过程保留过多过去的旧知识，可能会妨碍模型快速适应用户最新的兴趣。相反，对于偏好更为稳定的用户，尽可能保留其长期偏好可以显著提升模型性能。我们在图1中通过一个示例对此进行了说明。

因此，如何设计一种增量学习训练方案，对用户个性化偏好变化的动态进行建模，从而确定应保留多少过去的知识，是一个引人关注且重要的研究问题。我们通过关注知识蒸馏目标函数中的一个重要超参数来解决这一问题，该参数称为模仿权重，用于平衡新数据和蒸馏对总体损失的贡献。

在这项工作中，基于上述直觉，我们提出了一种新颖的端到端训练策略，为每位用户自适应地学习个性化模仿权重，以更好地平衡近期数据的贡献与从先前时间窗口中提炼的知识量。具体而言，我们首先将每位用户的偏好建模为一个分布，该分布表示其与物品簇中心之间的距离；这些簇是通过一种应用于用户-物品二部图的深度结构聚类方法获得的。簇学习过程被整合进整体训练流程中。随后，我们为每位用户构建一个状态向量，用以编码与该用户相关联的两个偏好分布之间的距离，这两个分布分别由连续的训练块推导而来。将此状态向量输入由神经网络参数化的权重生成器，以生成针对该用户的模仿权重。该个性化模仿权重决定了从教师（历史）模型继承多少与该用户相关的信息。我们提出的方法不局限于特定的骨干架构或增量学习流程，并且可以轻松集成到多种现有的最先进方法中。

总而言之，本文的主要贡献如下：

1. 我们证明，明确评估连续训练块之间的用户兴趣偏移，并利用这一信号学习用户专属的模仿权重，是一个重要的建模因素。它会显著影响基于知识蒸馏的增量学习技术的性能。据我们所知，这是首个明确对用户偏好变化进行建模的增量学习训练方案。

2. 我们提出了一种新颖的训练策略，可自适应地为每位用户学习个性化模仿权重，以平衡近期数据的贡献与从先前时期提炼出的知识量。

3. 我们通过在五个不同数据集上进行全面比较，展示了学习模仿权重的有效性。我们表现最佳的模型比最先进方法提升了 2.30%。我们将提出的训练流程与三种近期用于推荐系统的最先进增量学习技术相结合，并证明，与非自适应方法相比，该方法始终能够带来提升。

相关工作

增量学习

增量学习是机器学习的一个分支，旨在开发能够利用新数据持续更新的模型。然而，随着新批次数据到来而直接使用这些数据进行训练，会导致灾难性遗忘 问题：模型会遗忘先前学到的信息，并过度偏向新数据（Kirkpatrick 等人，2017年；Shmelkov、Schmid 和 Alahari，2017年；Castro 等人，2018年）。

应对这一问题的方法主要分为两类：（i）基于正则化的知识蒸馏（Castro 等，2018；Kirkpatrick 等，2017；Xu 等，2020；Wang、Zhang 和 Coates，2021）；（ii）经验回放，也称为蓄水池采样（Prabhu、Torr 和 Dokania，2020；Ahrabian 等，2021）。蓄水池方法会采样构建一个数据蓄水池，其中包含最具代表性的历史数据，并在学习新任务时对其进行回放，以缓解遗忘。一些关键的蓄水池采样研究，如 iCarl（Rebuffi 等，2017）和 GDumb（Prabhu、Torr 和 Dokania，2020），着重通过直接优化或贪心启发式方法来优化蓄水池构建。近期关于图推荐系统的研究拓展了 GDumb 启发式方法，并提出在构建蓄水池时对节点进行逆度采样（Ahrabian 等，2021）。

正则化技术通常在损失函数中引入惩罚参数，以防止模型权重彩票娱乐

使其偏离其在历史数据块上的调优值过远，从而防止遗忘（Yang 等，2019；Xu 等，2020；Wang、Zhang 和 Coates，2021）。知识蒸馏是最常见的正则化方法之一。知识蒸馏指的是将知识从大型复杂的教师模型迁移到较小的学生模型，同时避免性能显著下降的过程（Hinton、Vinyals 和 Dean，2015）。在增量学习中，教师模型 在历史数据上训练，而学生模型 在新数据上训练。

尽管增量学习和序列学习都利用历史信息，但序列推荐学习与增量学习在几个重要方面有所不同。首先，增量学习旨在通过知识蒸馏或经验回放继承先前训练模型的知识，从而大幅减少训练样本数量。相比之下，序列推荐系统通过记忆单元（Hidasi and Karatzoglou，2018年）、循环神经网络（RNN）或注意力设计（Kang and McAuley，2018年；Fan et al.，2021年），着重于更好地刻画用户的长期或短期交互序列。前者是在需要增量更新的场景中采用的一种训练策略，后者则是用于处理给定数据序列的特定模型架构。其次，推荐系统中的增量学习不依赖于任何特定类型的骨干架构。序列模型和非序列模型都应与增量训练方法兼容。

图结构数据上的增量学习

图表示学习技术已成为协同过滤和推荐系统的主流工具（Sun 等人，2019；Ying 等人，2018；Wang 等人，2019b；He 等人，2020）。然而，邻域采样过程、消息传递过程或邻接矩阵存储都会带来计算和内存负担，使图神经网络模型难以满足在线系统严格的训练时间限制（Xu 等人，2020）。为使图神经网络模型更好地保留结构信息，已有若干增量推荐系统设计。GraphSAIL（Xu 等人，2020）在节点层面、节点邻域层面和全局图层面采用知识蒸馏。LSP（Yang 等人，2020）最小化前一时间步训练的模型与微调模型所产生的结构相关分布之间的距离。SGCT（Wang、Zhang 和 Coates，2021）提出了一种用于知识蒸馏的对比方法。SGCT 的目标是最大化学生模型与教师模型中相邻节点嵌入对之间互信息的下界。LWC-KD（Wang、Zhang 和 Coates，2021）在 SGCT 的基础上，考虑了中间层嵌入蒸馏，并在用户—用户图和物品—物品图上增加了对比蒸馏。

我们工作的创新之处 以往的工作侧重于为所有用户普适性地蒸馏尽可能多的信息，而不对用户加以区分。然而，这并不总是最优或理想的做法，因为一些用户的兴趣可能会随时间快速变化，推荐模型应能够迅速适应新用户的偏好。我们的工作提出了一种自适应权重机制，用于学习针对每个用户需要蒸馏的知识量。我们通过实验表明，根据每个用户兴趣变化的速度来个性化蒸馏强度，能够使采用当前最先进骨干模型的推荐系统获得显著更好的推荐性能。

方法

本节介绍我们提出的结构感知增量学习框架——结合个性化模仿权重（简称 SAIL-PIW）。该框架通过刻画历史数据与新到达的增量数据之间用户兴趣分布的变化，利用个性化模仿权重。在知识蒸馏损失中使用个性化模仿权重，以平衡近期数据的贡献与从先前时间段蒸馏出的历史知识量。为更好地建模每位用户的偏好，我们学习用户到物品簇中心距离的分布，其中这些簇通过对用户-物品二部图应用结构聚类方法得到。获得用户兴趣分布后，我们使用增量学习块与教师模型之间的兴趣分布差异，作为用户兴趣转移指标。该转移指标刻画最近训练块与新到达的增量块之间用户偏好的变化。随后，将该转移指标输入由多层感知机参数化的权重生成器，以输出用户专属的模仿权重。我们提出的框架可通过反向传播以端到端方式进行训练。

图 2 展示了我们模型的整体架构。在接下来的小节中，我们首先详细介绍所提出方法的各个组成部分。因此，我们将内容细分为以下部分：(i) 所提出的个性化蒸馏损失；(ii) 模仿权重生成器的架构；(iii) 结构感知的物品聚类技术，用于获取聚类中心嵌入；最后，(iv) 用于构建刻画用户兴趣转移的度量的方法。值得注意的是，我们的框架不局限于特定的图神经网络模型、基于图的推荐系统架构或特定的增量学习框架。我们展示了以 GraphSAIL、SGCT 和 LWC-KD 为骨干模型时取得的良好结果。因此，我们提出的解决方案在现实场景中极具吸引力，因为无论现有基于图的系统采用何种基础模型，都可以轻松将其应用于这些系统之上。

个性化蒸馏损失

知识蒸馏通常用于模型压缩：小型模型（学生模型）通过继承大型模型（教师模型）的重要知识，力求达到与大型模型大致相当的性能（Hinton、Vinyals 和 Dean，2015）。通常，在增量学习场景中应用知识蒸馏时，我们使用在

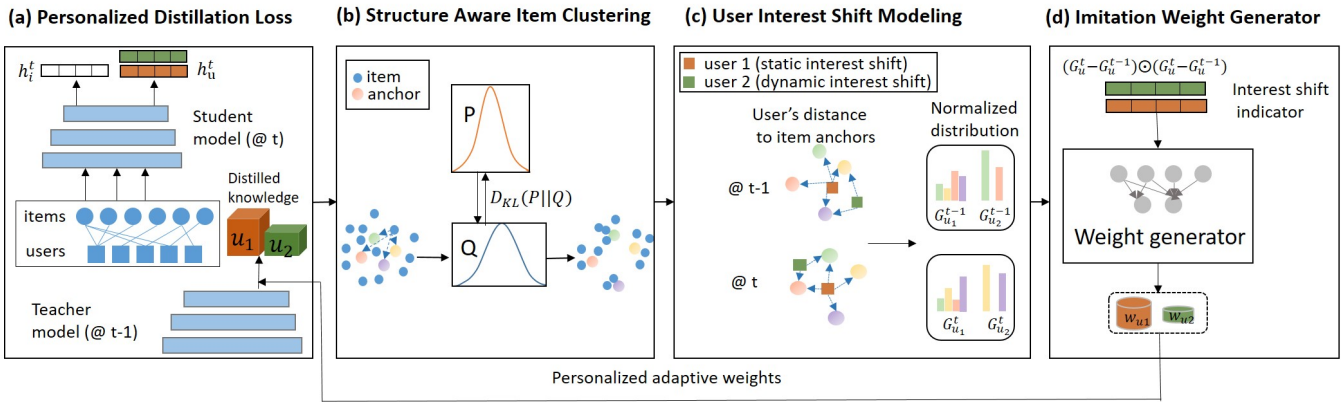

图 2：**我们提出的 SAIL-PIW 模型的整体框架。(a)**。时刻 t 的学生模型以新的用户-物品交互图为输入，学习用户和物品的嵌入，并通过时刻 $t-1$ 教师模型蒸馏的知识进行正则化。对于静态用户 1 和动态用户 2，蒸馏知识的多少由以下步骤中学习到的权重控制。**(b)**。我们学习图神经网络模型输出的物品嵌入与可学习的物品锚点嵌入之间的距离。该方法通过最小化已学习物品分布到簇 P 的分布与目标分布 Q 之间的 KL 散度来进行优化。**(c)**。分别计算用户在时刻 $t-1$ 和时刻 t 对物品锚点的分布。图中展示了两类用户：兴趣偏移较小的用户 1 和兴趣偏移较大的用户 2。**(d)**。用户兴趣偏移通过计算用户到物品簇中心的归一化距离分布之差得到。权重生成器以用户兴趣偏移指标为输入，生成个性化模仿权重，用于控制为每位用户蒸馏的知识，如**(a)**所述。

历史数据上训练的教师模型和在增量数据块上训练的学生模型。训练学生模型时，会在损失中引入蒸馏项（见公式（1）），以保留教师模型所获得的知识。增量学习模型的训练目标函数可表示为：

$$\mathcal{L}_S = \mathcal{L}_{new}(\boldsymbol{y}_S, \tilde{\boldsymbol{y}}_S) + \lambda \mathcal{L}_{KD}(\tilde{\boldsymbol{y}}_T, \tilde{\boldsymbol{y}}_S), \qquad (1)$$

其中，$\mathcal{L}_{new}$ 表示学生模型在真实标签 $\boldsymbol{y}_S$ 与预测值 $\tilde{\boldsymbol{y}}_S$之间的损失函数，而 $\mathcal{L}_{KD}$ 表示教师模型与学生模型之间的知识蒸馏损失。这里，λ 表示一个标量，用于控制训练过程中知识蒸馏损失所占的比重（Hinton、Vinyals 和 Dean, 2015）。

在推荐系统中，$\mathcal{L}_{new}$ 通常是贝叶斯个性化排序（BPR）（Rendle 等，2009）损失。在一种名为 GraphSAIL（Xu 等，2020）的近期基于图神经网络的知识蒸馏方法中，$\mathcal{L}_{KD}$ 由自嵌入蒸馏、全局结构蒸馏和局部结构蒸馏组成。尽管以往的工作已取得令人鼓舞的性能并实现了高效的训练过程（Xu 等，2020；Wang、Zhang 和 Coates，2021；Qiu 等，2020），但它们使用单一的模仿权重 λ 来保留以往知识，并将其应用于所有用户。然而，对所有用户一概完整保留历史信息并非最优，因为这种做法未能考虑每位用户偏好变化的动态性。实际上，我们观察到用户偏好具有动态性，而且不同用户在过去的时间块与增量数据块到来之间，兴趣变化程度可能各不相同。对于兴趣相较于以往时间段发生显著变化的用户，我们不希望教师模型过度约束学生模型对增量数据的学习。因此，基于这一思路，我们旨在针对不同用户区分不同的蒸馏级别，因此我们提出为每位用户自适应地学习一个模仿权重w_u，并将其应用于知识蒸馏目标函数。我们将在以下小节详细说明学习w_u的过程。

我们的个性化知识蒸馏损失为：

$$\mathcal{L}^U_{\text{KD - PIW}} = \sum_{u \in \mathcal{U}} \boldsymbol{w}_u \mathcal{L}^u_{KD}(\tilde{\boldsymbol{y}}_{u,T}, \tilde{\boldsymbol{y}}_{u,S})$$

其中，$\boldsymbol{w}_u$ 是为每位用户学习得到的个性化模仿权重，用于确定从教师模型中保留的知识量。$\mathcal{L}^u_{KD}$ 表示用户 u 的知识蒸馏损失。

整体的增量学习训练目标是：

$$\mathcal{L}_S = \mathcal{L}_{new}(\boldsymbol{y}_S, \tilde{\boldsymbol{y}}_S) + \lambda \left(\mathcal{L}^U_{\text{KD - PIW}} + \mathcal{L}^I_{\text{KD}}\right) \qquad (2)$$

其中，$\mathcal{L}^I_{\text{KD}} = \sum_{i \in \mathcal{I}} \mathcal{L}^i_{KD}(\tilde{\boldsymbol{y}}_{i,T}, \tilde{\boldsymbol{y}}_{i,S})$ 是所有物品的知识蒸馏损失，而 $\mathcal{L}^i_{KD}$ 指物品 i 的知识蒸馏损失。由于物品知识通常具有静态性质，我们采用原始知识蒸馏，不进行模仿权重个性化。

模仿权重生成器

直接学习个性化模仿权重 $\boldsymbol{w}_u$ 会因用户数量庞大而引入大量可学习参数。因此，我们提出使用由神经网络参数化的可学习函数来学习个性化模仿权重。对于每个训练块，我们为每位用户关联一个状态向量 $\boldsymbol{s}_u \in \mathbb{R}^M$，该向量表示用户在连续时间块之间的兴趣变化。其中 M 是物品簇的总数。下文将说明如何构建、初始化和更新该状态向量。目前，假设我们已有这样的向量 $\boldsymbol{s}_u$，

我们将其作为权重生成网络 $f(\boldsymbol{s}_u)$ 的输入，以学习每位用户用于蒸馏的模仿权重。在我们对该框架的实现中， $f(\boldsymbol{s}_u)$由以下方程定义：

$$\boldsymbol{z_u} = \text{relu}(\boldsymbol{W}_1 \cdot \boldsymbol{s}_u + b_1), \tag{3}$$

$$w_u = \text{softplus}(\boldsymbol{W}_2 \cdot \boldsymbol{z}_u + b_2), \tag{4}$$

其中， $\boldsymbol{W}_1 \in \mathbb{R}^{M\times l}$、 $\boldsymbol{W}_2 \in \mathbb{R}^{l}$、 $\boldsymbol{b}_1 \in \mathbb{R}^{l}$和 $\boldsymbol{b}_2 \in \mathbb{R}$为可学习参数， l 为隐藏层大小。我们注意到，采用更先进的网络作为权重生成器或许能进一步提升性能，但这并非本文的重点。因此，我们采用一种简单的形式，使用由两层多层感知机（MLP）参数化的权重生成器。在最后一层中，我们采用 Softplus（Zheng 等人，2015）激活函数，以生成严格非负的模仿权重。此外，我们对每个训练小批次中的所有用户模仿权重进行归一化，以更好地增强训练过程的稳定性。

结构感知物品聚类

为推导兴趣转移状态向量，首先需要对物品进行聚类。受深度结构聚类方法（Wang 等人，2019a；Bo 等人，2020年）的启发，该方法在聚类过程中充分考虑了节点属性和高阶结构信息，我们采用类似的目标函数，在给定底层用户-物品二部图的情况下学习物品中心簇。

我们衡量物品嵌入与物品簇中心嵌入之间的相似度。令$q_{i,m}^t \in \mathbb{R}^1$ 表示在时间点 t ，来自 GNN 骨干编码器 $\boldsymbol{h}_i^t \in \mathbb{R}^d$ 最后几层的物品嵌入与物品簇中心嵌入（物品锚点）$\boldsymbol{\mu}_m^t \in \mathbb{R}^d$之间的相似度。我们使用 Student t 分布来衡量这一距离，从而以计算上便捷的方式处理尺度不同的簇（Van der Maaten 和 Hinton，2008）。这可以视为每个物品的软聚类分配分布。当前时间块 t 中物品 i 属于物品簇 m 的分布质量计算如下：

$$q_{i,m}^t = \frac{(1+||\boldsymbol{h}_i^t - \boldsymbol{\mu}_m^t||^2/\nu)^{-\frac{\nu+1}{2}}}{\sum_{m'\in M}(1+||\boldsymbol{h}_i^t - \boldsymbol{\mu}_{m'}^t||^2/\nu)^{-\frac{\nu+1}{2}}}, \tag{5}$$

其中， M 表示物品簇的总数。

我们采用的深度结构聚类模型通过以下自监督学习损失进行训练：

$$\mathcal{L}_{soft}^{kl} = D_{KL}(P||Q)^t = \sum_i \sum_m p_{i,m}^t log\frac{p_{i,m}^t}{q_{i,m}^t}, \tag{6}$$

$$p_{i,m}^t = \frac{(q_{i,m}^t)^2/f_m^t}{\sum_{m'\in M}(q_{i,m'}^t)^2/f_{m'}^t}, \tag{7}$$

其中 $f_m^t = \sum_i q_{i,\ m}^t$, D_{KL} 表示 Kullback–Leibler（KL）散度（Kullback 和 Leibler，1951），而 $p_i^t \in \mathbb{R}^M$是时间点 t 时物品 i 的目标分布，旨在使表示更接近簇中心。定义好簇之后，我们现在可以推导$\mathbf{s}_u$。

用户兴趣转移建模

本节详细介绍$\mathbf{s}_u$的生成与初始化。我们使用 $\boldsymbol{\mu}_m^t$ 作为物品簇锚点嵌入。我们计算用户到各簇的距离，如下所示：

$$\tilde{\boldsymbol{G}}_u^t = [\boldsymbol{\mu}_1^t \boldsymbol{W}_1 (\boldsymbol{h}_u^t)^T, ..., \boldsymbol{\mu}_M^t \boldsymbol{W}_M (\boldsymbol{h}_u^t)^T], \tag{8}$$

$$\boldsymbol{G}_{u,m}^t = \frac{e^{\tilde{\boldsymbol{G}}_{u,m}^t}}{\sum_{m'=1}^M e^{\tilde{\boldsymbol{G}}_{u,m'}^t}}, \tag{9}$$

其中， $\boldsymbol{W}_m \in \mathbb{R}^{d\times d}$ 是特定于簇的变换矩阵。同样，利用前一个时间块的节点嵌入，我们可以得到 $\boldsymbol{G}_u^{t-1}$。我们假设，用户兴趣转移与用户到各物品簇的距离分布变化密切相关。因此，我们设计状态向量时假设用户的重要性权重与其在两个时间块之间的分布变化有关。状态向量定义如下：

$$\boldsymbol{s}_u = (\boldsymbol{G}_u^{t-1} - \boldsymbol{G}_u^t) \odot (\boldsymbol{G}_u^{t-1} - \boldsymbol{G}_u^t) \tag{10}$$

其中 $\odot$ 表示 Hadamard乘积。

我们的框架与任何基于图的推荐系统增量学习架构兼容，例如 GraphSAIL（Xu 等，2020）、SGCT（Wang、Zhang 和 Coates，2021）和 LWC-KD（Wang、Zhang 和 Coates，2021）。这些都是推荐系统领域标准的最先进增量学习方法。

整体训练框架

在介绍了自适应知识蒸馏损失的详细设计，以及刻画用户兴趣转变行为的状态向量 $\mathbf{s}_u$ 之后，我们现在重点介绍整体训练目标函数。我们的模型采用完全端到端的方式进行训练，将应用于增量块的贝叶斯个性化排序损失 $\mathcal{L}_{\text{BPR}}$ 、针对用户的个性化知识蒸馏损失 $\mathcal{L}_{\text{KD -PIW}}^U$ 、针对物品的蒸馏损失 $\mathcal{L}_{\text{KD}}^I$，以及用于物品聚类的自监督损失 $\mathcal{L}_{soft}^{kl}$ 联合起来，如下所示：

$$\mathcal{L} = \mathcal{L}_{\text{BPR}} + \lambda_1 \mathcal{L}_{soft}^{kl} + \lambda_2 \left(\mathcal{L}_{\text{KD - PIW}}^U + \mathcal{L}_{\text{KD}}^I\right). \tag{11}$$

这里， λ_1 和 λ_2 是用于平衡三个项之间损失贡献的系数。

表 1：数据统计。平均新用户百分比和平均新物品百分比分别指每个增量块中的新用户/新物品占全部用户/物品的平均比例。

数据集 / Stat	Gowalla	Yelp	Taobao2014	淘宝2015	Netflix
# 边	281412	942395	749438	1332602	12402763
用户	5992	40863	8844	92605	63691
平均用户度	46.96	23.06	84.74	14.39	194.73
# 物品	5639	25338	39103	9842	10271
平均物品度	49.90	37.19	19.17	135.40	1207.56
平均新用户百分比	2.67	3.94	1.67	2.67	4.36
平均新物品百分比	0.67	1.72	2.60	0.22	0.72
# 时间跨度（月）		6	1	5	6

表 2：所有基线方法、三种最新知识蒸馏算法与我们提出的个性化自适应权重设计的性能对比（Recall@20）。提升比例是相对于微调性能而言的。

数据集	方法	增量 1	增量 2	增量 3	Avg.	Imp %
Gowalla	微调	0.1412	0.1637	0.2065	0.1705	0.00
	LSP s	0.1512	0.1741	0.2097	0.1783	4.57
	均匀	0.1480	0.1653	0.2051	0.1728	1.34
	逆度	0.1483	0.1680	0.2001	0.1738	1.93
	GraphSAIL	0.1529	0.1823	0.2195	0.1849	8.44
	GraphSAIL-PIW	0.1547	0.1825	0.2253	0.1875	9.97
	SGCT	0.1588	0.1815	0.2207	0.1870	9.68
	SGCT-PIW	0.1599	0.1892	0.2321	0.1937	13.6
	LWC-KD	0.1639	0.1921	0.2368	0.1977	15.9
	LWC-KD-PIW	0.1698	0.1978	0.2425	**0.2033**	**19.3**
Yelp	微调	0.0705	0.0638	0.0640	0.0661	0.00
	LSP 秒	0.0722	0.0661	0.0644	0.0676	2.27
	均匀	0.0718	0.0635	0.0610	0.0654	-1.05
	逆度	0.0727	0.0699	0.0605	0.0677	2.42
	GraphSAIL	0.0674	0.0617	0.0625	0.0639	-3.33
	GraphSAIL-PIW	0.0718	0.0638	0.0615	0.0657	-0.66
	SGCT	0.0740	0.0656	0.0608	0.0668	1.06
	SGCT-PIW	0.0735	0.0655	0.0632	0.0674	1.92
	LWC-KD	0.0739	0.0661	0.0637	0.0679	2.72
	LWC-KD-PIW	0.0760	0.0690	0.0651	**0.0700**	**5.95**
淘宝14	微调	0.0208	0.0112	0.0138	0.0153	0.00
	LSP s	0.0213	0.0106	0.0138	0.0152	-0.65
	均匀	0.0195	0.0127	0.0148	0.0157	2.61
	逆度	0.0228	0.0140	0.0159	0.0175	14.63
	GraphSAIL	0.0222	0.0105	0.0139	0.0155	1.31
	GraphSAIL-PIW	0.0206	0.0103	0.0129	0.0146	-4.58
	SGCT	0.0240	0.0092	0.0148	0.0160	1.74
	SGCT-PIW	0.0227	0.0104	0.0142	0.0158	3.05
	LWC-KD	0.0254	0.0119	0.0156	0.0176	15.3
	LWC-KD-PIW	0.0256	0.0118	0.0161	**0.0178**	**16.3**
淘宝15	微调	0.0933	0.0952	0.0965	0.0950	0.00
	LSP s	0.0993	0.0952	0.0957	0.0968	1.86
	均匀	0.0988	0.0954	0.1004	0.0982	3.37
	逆度	0.0991	0.0977	0.1000	0.0989	4.16
	GraphSAIL	0.0959	0.0959	0.0972	0.0963	1.39
	GraphSAIL-PIW	0.1024	0.0983	0.1018	0.1008	6.14
	SGCT	0.1030	0.0983	0.0984	0.0999	5.16
	SGCT-PIW	0.1040	0.0999	0.1027	0.1022	7.58
	LWC-KD	0.1039	0.1022	0.1029	0.1030	8.42
	LWC-KD-PIW	0.1044	0.1045	0.1052	**0.1047**	**10.2**
Netflix	微调	0.1092	0.1041	0.0977	0.1036	0.00
	LSP 的	0.1173	0.1136	0.1076	0.1128	8.88
	均匀	0.1018	0.1055	0.0800	0.0957	-7.63
	逆度	0.1000	0.1050	0.0820	0.0957	-7.63
	GraphSAIL	0.1163	0.1023	0.0968	0.1051	1.45
	GraphSAIL-PIW	0.1142	0.1028	0.0986	0.1052	1.54
	SGCT	0.1166	0.1161	0.1077	0.1135	9.56
	SGCT-PIW	0.1185	0.1144	0.1098	0.1142	10.23
	LWC-KD	0.1185	0.1170	0.1071	**0.1142**	**10.23**
	LWC-KD-PIW	0.1185	0.1146	0.1087	0.1139	9.97

实验

数据集 我们使用了一组多样化的数据集，这些数据集由真实世界的用户-物品交互构成。如表 1 所示，各数据集的边数以及用户节点和物品节点数相差最多可达两个数量级，展现了我们方法的可扩展性。我们使用的 5 个主流公开数据集包括：Gowalla、Yelp、淘宝14、淘宝15 和 Netflix。

基础模型 我们在增量学习方法中使用 MGCCF（Sun 等人，2019）作为基础模型。它是增量推荐框架中先进的骨干模型，不仅在嵌入学习过程中融合了多个图，还在对二部图执行图卷积时考虑了用户节点和物品节点之间的内在差异。

基线方法 为了证明我们模型的优势，我们将我们的算法与多种基线方法进行了比较。

微调：微调仅使用每个时间块的新数据，对使用此前时间块训练的模型进行微调。

LSPs（Yang 等，2020）：LSP 是一种近期提出的先进方法，它在图卷积网络（GCN）模型上应用知识蒸馏。通过最小化表示局部拓扑语义的分布之间的距离，该方法将教师模型的局部结构保留到学生模型中。**均匀**：这是一种朴素的蓄水池回放方法。该方法对部分旧数据进行采样，并将其加入新数据中。

逆度（Ahrabian 等，2021）：逆度是一种先进的蓄水池回放方法。蓄水池基于图结构构建。该方法从历史数据中选取固定数量的用户-物品对；每次交互被选中的概率与其用户的逆度成正比。

最先进的图推荐系统增量学习方法：GraphSail（Xu 等，2020）、SGCT（Wang、Zhang 和 Coates，2021）和 LWC-KD（Wang、Zhang 和 Coates，2021）是最先进的模型，我们通过整合本方法对它们进行了改进。为展示本方法的优势，我们将其与基线模型进行比较。

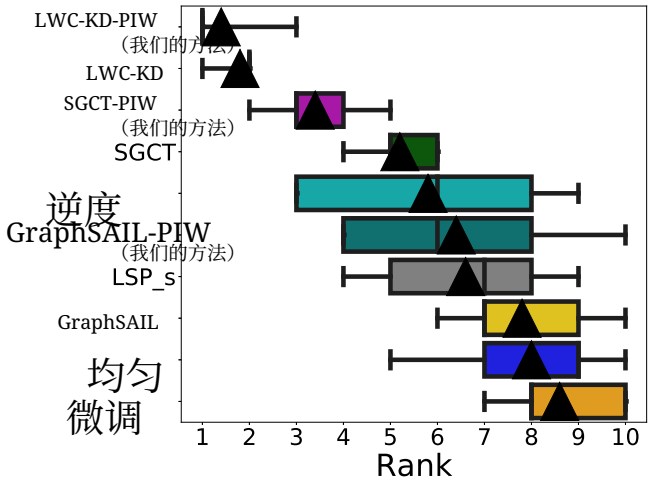

图 3：算法在 5 个数据集上的排名箱线图。排名的中位数和均值分别由垂直线和黑色三角形表示；须线延伸至最小排名和最大排名。

结果与讨论

表 2 汇报了基线方法、三种采用或未采用自适应权重的蒸馏算法，以及标准的储存回放方法的性能。请注意，所有报告的结果均为使用不同随机种子进行三次试验后的平均值。表 2 中所有数据集的结果汇总于图 3。我们注意到，自适应权重框架在所有情况下都提升了方法的性能，因为在引入自适应权重后，LWC-KD、SGCT 和 GraphSAIL 的基线方法的平均排名和中位数排名均有所提升。我们的方法通常在所有数据集上都位列前三，

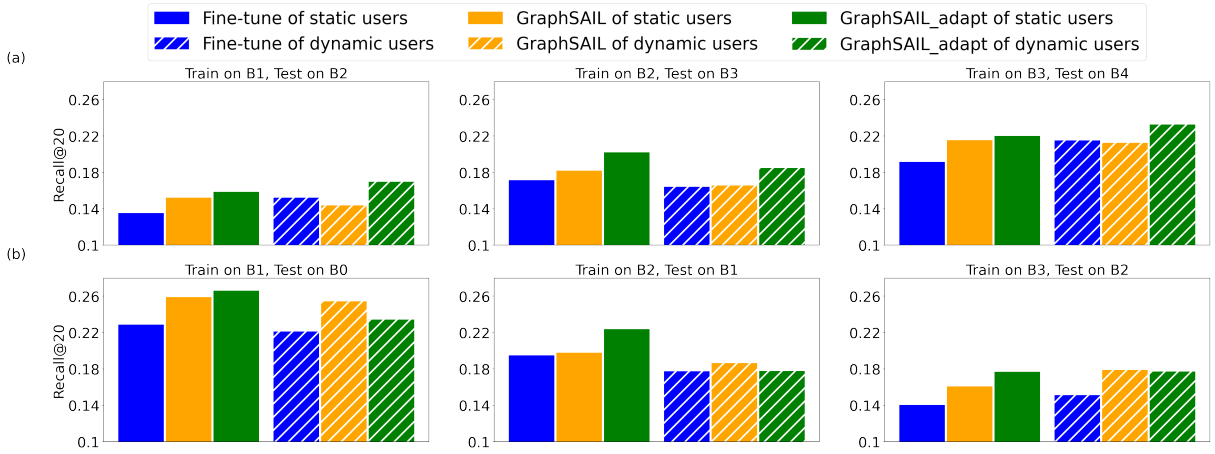

图4：**案例研究**：**(a)** 两组用户的模型性能：静态用户（兴趣变化最小的20%用户）和动态用户（兴趣变化最大的20%用户）。对于这两组用户，GraphSAIL adapt的表现均优于微调和不使用自适应权重的GraphSAIL。为探究自适应权重如何有效控制从教师模型为每个用户蒸馏的知识量，我们在时间块 t 上训练模型，并在**(b)**中的时间块 t-1 上进行测试。对于静态用户组，GraphSAIL adapt优于GraphSAIL；而对于动态用户组，GraphSAIL表现更好。这表明，GraphSAIL adapt会为具有持久偏好的用户保留更多信息，并为偏好最动态的用户遗忘更多信息。因此，不同程度的蒸馏有助于提升学生模型的性能。

并且经常取得最佳性能（见表 2 最后一列）。除了相对排名外，从绝对性能提升来看，在各种数据集上，自适应权重相较于微调带来了两位数百分比的性能提升。尤其是在 Gowalla 等传统增量学习数据集上，自适应权重表现最为出色。在 Netflix 等数据集中，大多数用户的交互数量非常多（超过 100 次）。因此，数据集的基础部分能够很好地反映每位用户的兴趣集合。因此，用户不太可能表现出显著的兴趣变化。

与全批次训练的比较在Gowalla和Taobao2014上，我们使用所有先前的时间块以及时间块 t 进行训练，并在每个增量块 t 上进行测试+1 （即全批次训练）。Gowalla的平均Recall@20为0.1963，Taobao14为0.0191。尽管全批次方法在Taobao2014上取得了更好的性能，但其训练耗时是LWC-KD-PIW的三倍。因此，在面向客户的推荐系统的在线部署环境中，我们的系统能够提供每日更新，而随着新数据不断累积，完全重新训练将很快变得在计算上不可行。

案例研究

我们开展了案例研究，以更仔细地考察三个模型对两类不同用户的表现。这三个模型分别是微调模型、不使用自适应权重的 GraphSAIL，以及使用自适应权重的 GraphSAIL。我们确定了两类用户：(i) 静态用户，其兴趣偏移很小；以及动态用户，其兴趣偏移显著。随后，我们在历史信息上测试这些模型（即来自前一时间块的数据）。在旧数据上进行评估，可以帮助我们了解模型如何保留每类用户的历史信息。我们还通过在下一个时间块上进行测试，检查每类用户的表现。因此，我们可以评估为每类用户保留不同数量的信息会如何影响模型在相关任务上的性能。

我们观察到，对于这两组用户，采用自适应权重的 GraphSAIL 均表现最佳（图4 (a)）。通过对历史数据的评估（图4 (b)），我们发现，与微调相比，GraphSAIL 受到的影响较小，这表明它能够有效缓解遗忘问题。对于静态用户，采用自适应权重的 GraphSAIL 优于未采用自适应权重的 GraphSAIL；而对于动态用户，未采用自适应权重的 GraphSAIL 表现更好。这意味着，采用自适应权重的 GraphSAIL 能够为静态用户保留更多信息，而为动态用户保留的信息较少。因此，我们得出结论：通过自适应方式蒸馏知识，有助于改进对未来用户偏好的建模。

结论

本文提出了一种用于基于图的推荐系统增量学习的新方法。我们的方法通过学习自适应个性化权重，在增量学习过程中调整针对每位用户偏好所蒸馏的知识量。我们在多个数据集上，结合三种不同的增量学习骨干模型对所提出的方法进行了评估；结果表明，该方法始终优于标准的非自适应技术。案例研究进一步支持了我们的观点：采用自适应权重，模型能够为兴趣稳定的用户蒸馏更多信息，并为兴趣变化迅速的用户保留较少信息。这使模型能够更快地适应兴趣不断变化的用户的偏好变化。

参考文献

Ahrabian, K.； Xu, Y.； Zhang, Y.； Wu, J.； Wang, Y.； 和 Coates, M. 2021年。《基于图的推荐系统增量学习的结构感知经验回放》。载于*ACM*国际信息与知识管理会议论文集，2832–2836。

Bo, D.； Wang, X.； Shi, C.； Zhu, M.； Lu, E.； 和 Cui, P. 2020年。结构化深度聚类网络。载于《万维网大会（*WWW*）论文集，1400–1410。

Castro, F. M.; Mar´ın-Jim´enez, M. J.; Guil, N.; Schmid, C.; 和 Alahari, K. 2018年。端到端增量学习。载于欧洲计算机视觉会议（*ECCV*），241–257。

Cheng, H.; Koc, L.; Harmsen, J.; Shaked, T.; Chandra, T.; 等. 2016年. 面向推荐系统的宽深度学习. 收录于*ACM*推荐系统会议论文集 - 推荐系统深度学习研讨会, 7–10.

Chiang, W.-L.; Liu, X.; Si, S.; Li, Y.; Bengio, S.; 和 Hsieh, C.-J. 2019. Cluster-GCN：一种用于训练深层大型图卷积网络的高效算法。载于 *ACM SIGKDD*知识发现与数据挖掘国际会议论文集，257–266。

Covington, P.; Adams, J.; and Sargin, E. 2016年. 用于 YouTube 推荐的深度神经网络. 载于*ACM*推荐系统会议论文集, 191–198.

Fan, Z.; Liu, Z.; Zhang, J.; Xiong, Y.; Zheng, L.; and Yu, P. S. 2021年. 基于时序图协同变换器的连续时间序列推荐. 载于 *CIKM*.

Guo, H.; Tang, R.; Ye, Y.; Li, Z.; and He, X. 2017年. DeepFM：一种用于点击率预测的基于因子分解机的神经网络. 载于国际人工智能联合会议（*IJCAI*）论文集，1725–1731.

He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; and Wang, M. 2020年. LightGCN：简化并增强用于推荐的图卷积网络. *ACM*信息检索研究与发展国际会议论文集.

Hidasi,B.;andKaratzoglou, A. 2018年. 用于基于会话的推荐的具有 top-k增益的循环神经网络。载于*CIKM*。

Hinton, G.; Vinyals, O.; and Dean, J. 2015. 神经网络中的知识蒸馏。*CoRR*，arXiv。

Kang, W.-C.; and McAuley, J. 2018年. 自注意力序列推荐。载于 *ICDM*。

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho,T.;Grabska-Barwinska,A.; 等. 2017年. 克服神经网络中的灾难性遗忘。美国国家科学院院刊，114(13)：3521–3526。

Kullback, S.; and Leibler, R. A. 1951. 关于信息与充分性。数学统计年刊，22：79–86。

Mallya, A.; 和 Lazebnik, S. 2018年。Packnet：通过迭代剪枝将多个任务添加到单个网络中。载于计算机视觉与模式识别会议（*CVPR*），7765–7773。

Prabhu, A.; Torr, P. H.; 和 Dokania, P. K. 2020年。GDumb：一种简单的方法，对我们在持续学习方面的进展提出质疑。载于欧洲计算机视觉会议（*ECCV*），524–540。Qiu, R.; Yin, H.; Huang, Z.; 和 Tong, C. 2020年。GAG：用于流式会话式推荐的全局属性图神经网络。载于*ACM SIGIR*信息检索研究与发展国际会议，669–678。Rebuffi, S.-A.; Kolesnikov, A.; Sperl, G.; 和 Lampert, C. H. 2017年。iCarl：增量分类器与表征学习。载于*IEEE*计算机视觉与模式识别会议论文集，2001–2010。

Rendle, S.; Freudenthaler, C.; Gantner, Z.; and Schmidt-Thieme, L. 2009. BPR：基于隐式反馈的贝叶斯个性化排序。在人工智能不确定性国际会议论文集（*UAI*），452–461。

Shmelkov, K.; Schmid, C.; and Alahari, K. 2017. 无灾难性遗忘的目标检测器增量学习。在国际计算机视觉会议论文集（*ICCV*），3400– 3409。

Sun, J.; Zhang, Y.; Ma, C.; Coates, M.; Guo, H.;Tang, R.; and He, X. 2019. 多图卷积协同过滤。在电气电子工程师学会国际数据挖掘会议论文集（*ICDM*），1306–1311。

van den Berg, R.; Kipf, T. N.; and Welling, M. 2017. 图卷积矩阵补全。卷 abs/1706.02263。

Van der Maaten, L.; and Hinton, G. 2008. 使用 t-SNE 可视化数据。《机器学习研究杂志》，9: 2579–2605。

Wang, C.; Pan, S.; Hu, R.; Long, G.; Jiang, J.; and Zhang, C. 2019a. 属性图聚类：一种深度注意力嵌入方法。发表于国际人工智能联合会议（*IJCAI*）论文集，3670–3676。

Wang, S.; Hu, L.; Wang, Y.; He, X.; Sheng, Q. Z.; Orgun, M. A.; Cao, L.; Ricci, F.; and Yu, P. S. 2021年. 基于图学习的推荐系统：综述。发表于国际人工智能联合会议（*IJCAI*）论文集，4644–4652。

Wang, X.; He, X.;Wang, M.;Feng, F.; and Chua, T.-S. 2019b. 神经图协同过滤。发表于*ACM*信息检索研究与发展国际会议论文集，165–174。

Wang, Y.; Zhang, Y.; and Coates, M. 2021年. 面向推荐系统增量学习的图结构感知对比知识蒸馏. 收录于*ACM*国际信息与知识管理会议论文集, 3518–3522.

Xu, J.; and Zhu, Z. 2018年. 强化持续学习. 收录于神经信息处理系统大会（*NeurIPS*）论文集, 907–916.

Xu, Y.; Zhang, Y.; Guo, W.; Guo, H.; Tang, R.; and Coates, M. 2020年. GraphSAIL：面向推荐系统的图结构感知增量学习. 收录于*ACM*国际信息与知识管理会议论文集, 2861–2868.

Yang, Y.; Qiu, J.; Song, M.; Tao, D.; 和 Wang, X. 2020年. 从图卷积网络中蒸馏知识 . 收录于 电气电子工程师学会计算机视觉与模式识别会议论文集，7074–7083。

Yang, Y.; Zhou, D.-W.; Zhan, D.-C.; Xiong, H.; 和 Ji
ang, Y. 2019. 面向增量学习的自适应深度模型：兼顾容量可扩展性与可持续性。收录于*ACM*知识发现与数据挖掘会议论文集，74–82。Ying, R.; He, R.; Chen, K.; Eksombatchai, P.; Hamilton, W. L.; 和 Leskovec, J. 2018年. 面向网络规模推荐系统的图卷积神经网络. 收录于 *ACM*知识发现与数据挖掘会议论文集，974–983。Zeng, H.; Zhou, H.; Srivastava, A.; Kannan, R.; 和 Prasanna, V. 2020年. GraphSAINT：基于图采样的归纳学习方法. 收录于 国际学习表征会议（*ICLR*）论文集。Zheng, H.; Yang, Z.; Liu, W.; Liang, J.; 和 Li, Y. 2015. 使用 Softplus 单元改进深度神经网络。*2015*年国际神经网络联合会议（*IJCNN*），1–4。Zou, D.; Hu, Z.; Wang, Y.; Jiang, S.; Sun, Y.; 和 Gu, Q. 2019. 训练深层大型图卷积网络的层依赖重要性采样. 收录于 神经信息处理系统大会（*NeurIPS*）论文集，11247–11256。

arXiv:2305.01204v1 [cs.IR] 2 May 2023

补充材料

Yuening Wang[1], **Yingxue Zhang**[1], **Antonios Valkanas**[2*], **Ruiming Tang**[1], **Chen Ma**[3], **Jianye Hao**[1, 4], **Mark Coates**[2]

[1] 华为诺亚方舟实验室, [2] 麦吉尔大学, [3] 香港城市大学, [4] 天津大学, yuening.wang@huawei.com, yingxue.zhang@huawei.com, antonios.valkanas@mail.mcgill.ca, tangruiming@huawei.com, chenma@cityu.edu.hk, haojianye@huawei.com, mark.coates@mcgill.ca

1 补充动机图

我们认为，对所有用户保留相同数量的历史信息可能并非最优，因为这未能考虑到每位用户潜在偏好变化的动态性。对于兴趣发生显著转变的用户，在知识蒸馏过程中保留过多过去的旧知识，可能会限制模型快速适应用户最新的兴趣。相反，对于偏好较为稳定的用户，尽可能保留其长期偏好可以显著提升模型性能。我们在图1中用一个例子进行说明。我们根据用户交互过的物品类别，提取每位用户在每个时间块上的兴趣分布。我们将用户在时间块 t 与类别 c 中的物品进行交互的概率定义为 $\frac{e^{N_c^t}}{\sum_{c'=1}^{C} e^{N_{c'}^t}}$，其中 N_c^t 表示用户在时间块 t 中与类别 c 的物品发生交互的数量，C 表示物品类别的总数。数据集中的类别（物品标签）是预先定义的。直方图展示了在 Taobao2014 数据集中，用户在基础块和第一个增量块之间兴趣分布的 L2 差异大小。虽然大多数用户在不同时间块中与相似类别的物品保持交互，表现出较小的分布变化（图的左侧），但确实有一部分用户在连续时间块之间的兴趣分布发生了较大变化（图的中间和右侧）。因此，在执行基于知识蒸馏的增量学习时，考虑每位用户偏好的潜在变化动态至关重要。

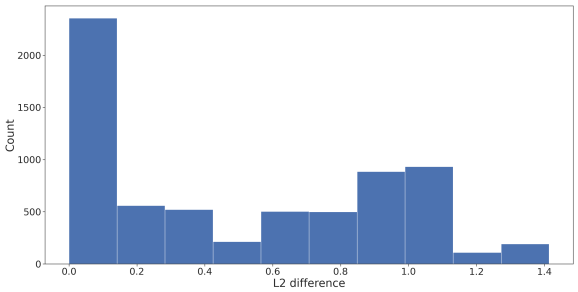

图1：Taobao2014 中两个连续时间块之间用户兴趣转移的示意图。

2 数据集详情

我们采用了基线方法的实验设置和评估方案（Ahrabian 等，2021；Xu 等，2020；Wang、Zhang 和 Coates，2021）。

• **Gowalla**[1]：这是一个真实世界的数据集，收集自用户在基于位置的社交网络网站上签到并分享位置的信息。我们过滤掉了交互少于 20 次的用户和物品节点。

• **Yelp**[2]：这是一个包含 Yelp 商家综述和用户数据的数据集。由于数据量庞大，我们只保留了最近五年的数据，并过滤掉了交互少于 10 次的用户和物品节点。

• **Taobao2014**[3]：阿里巴巴集团提供的 2014 年真实用户行为数据。我们筛除了交互次数少于 10 次的用户节点和物品节点。

•**Taobao2015**[4]：该真实世界数据集于 2015 年在天猫/淘宝和支付宝应用上积累。我们筛除了交互次数少于 10 次的用户节点和物品节点。

• **Netflix-Prize**[5]：这是 NetflixPrize 竞赛使用的官方用户电影评分数据集。我们仅保留评分为 5 的交互，并筛除了交互次数少于 100 次的用户节点和物品节点。

对于所有数据集，我们按时间顺序将数据划分为一个基础块和多个增量块。基础块包含 60% 的数据，并随机划分为训练集、验证集和测试集。其余 40% 的数据均匀划分为四个增量块。在训练过程中，当数据块 t 用作训练集时，数据块 $t+1$ 的前半部分用作验证集，后半部分用作测试集（图 2）。

*在华为诺亚方舟实验室实习期间完成。版权所有 © 2023年，美国人工智能促进协会（www.aaai.org）。保留所有权利。

[1]https://snap.stanford.edu/data/loc-gowalla.html

[2]https://www.yelp.com/dataset

[3]https://tianchi.aliyun.com/dataset/dataDetail?dataId=46

[4]https://tianchi.aliyun.com/dataset/dataDetail?dataId=53

[5]https://academictorrents.com/details/9b13183dc4d60676b773c9e2cd6de5e5542cee9a

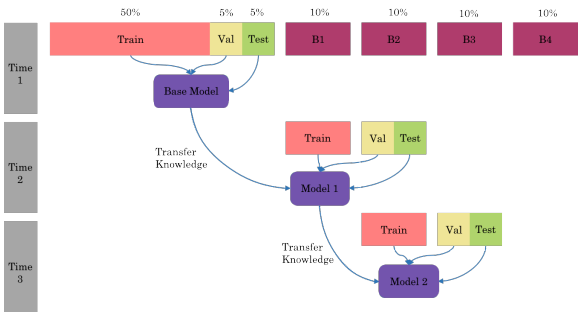

图 2：实验设置的数据划分。

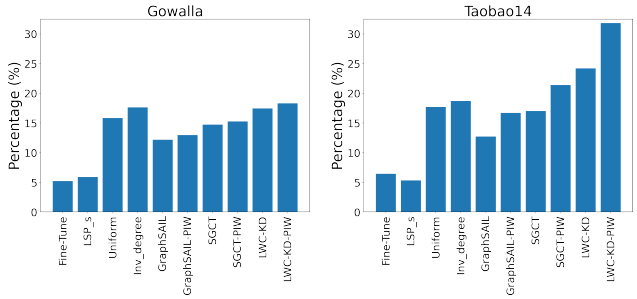

图 3：相较于全批次训练所用时间的百分比。对于 Gowalla 和 Taobao2014 数据集，我们实现了全批次训练版本，并以其训练时间作为参照。图中展示了训练完一个增量块所用时间占全批次训练时间的百分比。

3 超参数设置

训练循环使用 TensorFlow 和 Adam 优化器（Kingma 和 Ba，2015）实现。学习率固定为 0.0005。我们至少训练模型 10 个训练轮次，并采用早停策略，耐心参数设为 2。

骨干图神经网络采用 MGCCF（Sun 等，2019），并使用表 1 所示的超参数进行训练。

表 1：我们的模型在所有基准数据集上的超参数。

超参数	值
最小训练轮次（基础块）	10
最小训练轮次（增量）	3
最大训练轮次（基础块）	N/A
最大训练轮次（增量）	10
早停耐心值	2
批次大小	64
优化器	Adam
学习率（最大值）	5e-4
随机失活	0.2
损失函数	L2、贝叶斯个性化排序
图神经网络层数（R）	2
嵌入维度	128
数据增强	NONE

4 计算需求与经验时间复杂度

对于实验中规模最大的数据集 Netflix，在配备 72 个 Intel Xeon (R) Gold 6140 2.30GHz CPU 和一块 NVIDIA Tesla V100 GPU 的专用服务器上，训练代表数据集 60% 的基础块约需 40 小时。每个增量块占数据集的 10%，训练约需 7 小时。如果在每个时间步都进行全批次重新训练，训练全部 4 个数据块约需 $40+47+54+61 = 202$ 小时。另一方面，采用增量学习方法，训练耗时为 $40 + 7 + 7 + 7 = 61$ 小时（减少了 70%！）。这一结果表明，我们的增量学习设计在节省时间和能源方面具有显著潜力。

在模型准确率方面，对于淘宝 2014，全批次训练的 Recall@20 为 0.0191，而表现最佳的增量学习方法 LWC-KD-PIW 的 Recall@20 为 0.0178。对于 Gowalla，全批次训练的 Recall@20 为 0.1963，而 LWC-KD-PIW 的 Recall@20 为 0.2033，高于全批次训练范式。因此，我们可以得出结论：即使从模型准确率的角度来看，也不能保证累积更多历史数据进行训练就一定能带来性能优势。

在面向客户的推荐系统在线部署环境中，增量训练方法能够每日更新模型，而全批次重新训练则会面临收敛缓慢的问题，且无法使用最新的数据流训练模型，从而导致性能下降（Xu 等，2020年）。我们比较了 Taobao2014 和 Gowalla 上采用增量训练方法完成一个增量块 t 的平均训练时间与全批次训练方案的训练时间（使用截至 t 的所有时间块进行训练，并在 t+1 上测试）。如图 3 所示，尽管 LWC-KD-PIW 在所有增量学习方法中耗时最长，但其训练时间最多仅为全批次训练方案的 30%，这表明增量学习有望显著节省训练时间。

5 种具体实现

本节介绍三种具体实现，将我们的自适应模仿学习策略与三种近期的推荐系统增量学习技术结合应用。

GraphSAIL（Xu 等，2020） GraphSAIL（GS）对每个节点的局部邻域、相对于其他节点簇的全局位置，以及每个节点自身的嵌入进行蒸馏。蒸馏损失项 $\mathcal{L}_{GS-adapt}$（Xu 等，2020）的公式 12）修改为：

$$\mathcal{L}_{GS-adapt} = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}} w_u\Big(\mathcal{L}_{self}(u) + \mathcal{L}_{local}(u) + \mathcal{L}_{global}(u)\Big) + \mathcal{L}^I_{KD}, \tag{1}$$

其中 $\mathcal{U}$ 表示用户节点集合，GraphSAIL 的自身、局部和全局损失项定义见 GraphSAIL论文（Xu等人（2020））中的公式 3、8 和 9。这里，w_u 是由权重生成网络学习得到的自适应权重。

SGCT（Wang、Zhang 和 Coates，2021年）对比蒸馏目标用于使潜在空间中的正样本对的嵌入更加接近，并将负样本对的嵌入推向距离更远的位置。正样本对和负样本对由用户-物品二部图构建，以捕获重要的结构信息。随后，将自适应权重应用于用户的对比蒸馏项，从而避免在用户兴趣发生较大变化时，将用户的嵌入与先前的正邻居“推”得过于接近，即用户可能已不再对先前的正邻居感兴趣。因此，蒸馏目标项（Wang、Zhang 和 Coates，2021年）中的 $\mathcal{L}_{const}$ 变为：

$$\mathcal{L}_{sgct-adapt} = \mathcal{L}_{KD}^{I} + \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\frac{-1}{|\mathcal{N}_{UI}^{t-1}(u)|} \sum_{i\in\mathcal{N}_{UI}^{t-1}(u)} w_u \log\frac{\exp\left(\boldsymbol{h}_{u,0}^{t}\cdot\boldsymbol{h}_{i,0}^{t-1}/\tau\right)}{\sum_{\hat{i}\in\mathcal{D}_{UI}^{t-1}(u)}\exp\left(\boldsymbol{h}_{u,0}^{t}\cdot\boldsymbol{h}_{\hat{i},0}^{t-1}/\tau\right)} \tag{2}$$

其中，$\boldsymbol{h}_{u,0}^{t}$ 是节点 u 在时间 t 的嵌入，$\mathcal{N}_{UI}^{t-1}(u)$ 是用户节点 u 在时间 $t-1$ 的用户-物品交互图中的邻居集合，其中提供了正样本；$\mathcal{D}_{UI}^{t-1}(u)$ 是从时间 $t-1$ 的用户-物品二部图中生成的用户 u 的正样本和负样本的集合（并集）。τ 是用于调节浓度水平的温度参数。

LWC-KD（Wang、Zhang 和 Coates，2021年）：在*SGCT*的基础上，该方法进一步利用了：(a) 逐层蒸馏，注入中间层级的监督信号；(b) 不仅在用户-物品二部图上，而且在用户-用户图和物品-物品图上使用对比蒸馏损失。自适应权重会根据用户-物品图和用户-用户图上的对比蒸馏项，为用户应用相应权重。因此，模型会根据用户兴趣的变化程度，控制用户嵌入向先前时间块中相似的用户或感兴趣的物品“靠拢”的程度。蒸馏目标（$\mathcal{L}_{lw-const}$，见 Wang、Zhang 和 Coates，2021年）修改如下：

$$\mathcal{L}_{lw-const-adapt} = \mathcal{L}_{KD}^{I} + \frac{1}{K}\sum_{k=0}^{K}\frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}} \left\{ \frac{-1}{|\mathcal{N}_{UI}^{t-1}(u)|}\sum_{i\in\mathcal{N}_{UI}^{t-1}(u)} w_u \log\frac{\exp\left(\boldsymbol{h}_{u,k}^{t}\cdot\boldsymbol{h}_{i,k}^{t-1}/\tau\right)}{\sum_{\hat{i}\in\mathcal{D}_{UI}^{t-1}(u)}\exp\left(\boldsymbol{h}_{u,k}^{t}\cdot\boldsymbol{h}_{\hat{i},k}^{t-1}/\tau\right)} + \frac{-1}{|\mathcal{N}_{UU}^{t-1}(u)|}\sum_{u'\in\mathcal{N}_{UU}^{t-1}(u)} w_u \log\frac{\exp\left(\boldsymbol{h}_{u,k}^{t}\cdot\boldsymbol{h}_{u',k}^{t-1}/\tau\right)}{\sum_{\hat{u}\in\mathcal{D}_{UU}^{t-1}}\exp\left(\boldsymbol{h}_{u,k}^{t}\cdot\boldsymbol{h}_{\hat{u},k}^{t-1}/\tau\right)} \right\} \tag{3}$$

其中，$\boldsymbol{h}_{u,k}^{t}$ 是节点 u 在时间 t 和层 k 的嵌入。$\mathcal{N}_{UU}^{t-1}(u)$ 和 $\mathcal{D}_{UU}^{t-1}(u)$ 分别是正邻居和负邻居——用户 u 在时间 $t-1$ 的用户-用户相似度图中的邻域集合，为相应术语的对比目标提供样本。

6 消融研究

为了考察各个组件——权重生成器、转移矩阵和聚类——的有效性，我们还使用 SGCT 和 LWC-KD 算法，在 Gowalla 和淘宝2015数据集上进行了实验。在每项消融实验中，我们都按照表 2 所示，从最终方法中移除一个组件。因此，这些方法的形式如下：

SAIL-PIW-no-wg：学习权重时不使用权重生成器。因此，计算完式 (5) 后，我们直接使用 $w_u = ||\boldsymbol{G}_u^{t-1} - \boldsymbol{G}_u^{t}||^2$ 计算权重。

$$\tilde{\boldsymbol{G}}_u^{t} = [\boldsymbol{\mu}_1^{t}\boldsymbol{W}_1(\boldsymbol{h}_u^{t})^T, ..., \boldsymbol{\mu}_M^{t}\boldsymbol{W}_M(\boldsymbol{h}_u^{t})^T], \tag{4}$$

$$\boldsymbol{G}_{u,m}^{t} = \frac{e^{\tilde{\boldsymbol{G}}_{u,m}^{t}}}{\sum_{m'=1}^{M} e^{\tilde{\boldsymbol{G}}_{u,m'}^{t}}}, \tag{5}$$

SAIL-PIW-no-cluster：在计算用户对物品簇的兴趣分布时，不对物品进行深度聚类。相反，我们直接根据两个连续时间块的用户-物品二部图，计算用户相对于其相邻物品的分布变化。因此，状态向量设计如下：

$$\boldsymbol{d}_u^{t} = [\boldsymbol{h}_{i_1}^{t}\boldsymbol{W}(\boldsymbol{h}_u^{t})^T, \ldots, \boldsymbol{h}_{i_n}^{t}\boldsymbol{W}(\boldsymbol{h}_u^{t})^T] \text{ where } i_{1,\ldots,n} \in \mathcal{N}_{UI}^{t-1}, \tag{6}$$

$$\boldsymbol{s}_u = (d_u^{t-1} - d_u^{t}) \odot (d_u^{t-1} - d_u^{t}) \tag{7}$$

其中，$\boldsymbol{W} \in \mathcal{R}^{d\times d}$ 是变换矩阵，且两个时间块使用相同的物品集合。我们以与 $\boldsymbol{d}_u^{t}$ 类似的方式得到 $\boldsymbol{d}_u^{t-1}$。

SAIL-PIW-no-trans：此设计不使用变换。因此，式 5 中的步骤变为

$$\tilde{G}_u^{t} = [\boldsymbol{\mu}_1^{t}(\boldsymbol{h}_u^{t})^T, ..., \boldsymbol{\mu}_M^{t}(\boldsymbol{h}_u^{t-1})^T] \tag{8}$$

SAIL-PIW-hard：除了前述设计外，我们还进行了另一项消融研究，使用 K-means 代替深度聚类网络来获取簇的平均嵌入 $\boldsymbol{\mu}_m^{t}$。具体而言，我们对 $\boldsymbol{h}_i^{t-1}$ 应用 K-means，并识别出 M 个簇。然后，我们设置 $\boldsymbol{\mu}_m^{t} = \frac{1}{|C_m^{t}|}\sum_{i\in C_m^{t-1}}(\boldsymbol{h}_i^{t})$，其中 C_m^{t-1} 表示簇 m 的物品集合。

我们从表 3 中观察到，移除任何一个组件都会导致性能下降，未采用端到端训练进行聚类也会如此。因此，我们得出结论：算法设计的每个方面都有助于其取得更优性能。我们还在表 4 中评估了使用 SGCT 和 LWC-KD 时，模型性能随簇数量变化的情况。我们观察到，性能并不严重依赖于 K 的取值。换言之，模型对簇数量的选择具有稳健性。如果选择的簇数量过少或过多，整体性能可能会略有下降。

表 2：消融分析中的方法组件。

方法	权重生成器	转移矩阵	聚类
SAIL-PIW-no-wg	✗	✓	✓
SAIL-PIW-no-cluster	✓	✓	✗
SAIL-PIW-no-trans	✓	✗	✓
SAIL-PIW-hard	✓	✓	✓
SAIL-PIW	✓	✓	✓

表 3：状态向量设计的消融分析。SGCT 和 LWC-KD 蒸馏算法在 Gowalla 和淘宝2015数据集上的 Recall@20 结果。

蒸馏策略	数据集	算法	增量 1	增量 2	增量 3	平均召回率@20
SGCT	Gowalla	SAIL-PIW-no-wg	0.1535	0.1807	0.2221	0.1854
		SW-AIW-click-uni-interest	0.1525	0.1828	0.2169	0.1841
		SAIL-PIW-no-trans	0.1497	0.1842	0.2225	0.1855
		SAIL-PIW-hard	0.1573	0.1860	0.2212	0.1882
		SAIL-PIW	0.1599	0.1892	0.2321	0.1937
	淘宝2015	SAIL-PIW-no-wg	0.1022	0.0998	0.1013	0.1011
		SW-AIW-click-uni-interest	0.1019	0.0992	0.1012	0.1008
		SAIL-PIW-no-trans	0.1021	0.0987	0.1000	0.1003
		SAIL-PIW-hard	0.0972	0.0975	0.0989	0.0979
		SAIL-PIW	0.1040	0.0999	0.1027	0.1022
LWC-KD	Gowalla	SAIL-PIW-no-wg	0.1561	0.1861	0.2350	0.1924
		SW-AIW-click-uni-interest	0.1596	0.1865	0.2337	0.1933
		SAIL-PIW-no-trans	0.1631	0.1875	0.2288	0.1931
		SAIL-PIW-hard	0.1604	0.1889	0.2246	0.1913
		SAIL-PIW	0.1698	0.1978	0.2425	0.2033
	淘宝2015	SAIL-PIW-no-wg	0.1007	0.1000	0.1032	0.1013
		SW-AIW-click-uni-interest	0.1022	0.1006	0.1021	0.1016
		SAIL-PIW-no-trans	0.1004	0.1007	0.1035	0.1015
		SAIL-PIW-hard	0.1032	0.1012	0.1031	0.1025
		SAIL-PIW	0.1044	0.1045	0.1052	0.1047

7 案例研究——补充详情

我们进行了案例研究，以更深入地考察三个模型在两组不同用户上的表现。我们研究的三个模型分别是微调模型、不使用自适应权重的 GraphSAIL，以及使用自适应权重的 GraphSAIL。我们确定了两类用户：(i) 静态 用户，其兴趣偏移极小；以及动态 用户，其兴趣偏移显著。随后，我们使用历史数据（即来自先前时间块的数据）对模型进行测试。案例研究中确定用户兴趣偏移的详细步骤如下：

1. 对从 GraphSAIL 模型中 $\boldsymbol{h}_i^{t-1}$ 获得的物品嵌入执行 K 均值，以识别 M 个簇。这里，t 为训练块编号。2. 统计每个用户在每个簇中交互过的物品数量，以构建用户在时间块 t 和数据块 $t-1$中的兴趣直方图。我们计算 $\tilde{\boldsymbol{I}}^t \in \mathbb{R}^{U\times M}$ 和 $\tilde{\boldsymbol{I}}^{t-1} \in \mathbb{R}^{U\times M}$。3. 对 $\tilde{\boldsymbol{I}}$ 进行归一化以计算 $\boldsymbol{I}_m = \tilde{\boldsymbol{I}}_m / \sum_{m'\in M} \tilde{\boldsymbol{I}}'_m$。 4. 计算用户的兴趣转移指标分数：$ISS_u = \frac{1}{M}\sum_M ||\boldsymbol{I}_u^t - \boldsymbol{I}_u^{t-1}||^2$。5. 选取兴趣转移指标分数排名前 20% 的用户组成动态组，并选取排名后 20% 的用户组成静态组，同时记录相应的用户索引。6. 分别计算两组用户在微调、GraphSAIL 和 GraphSAIL-PIW 下的平均召回率。

表 4：聚类数量 K 的消融分析。SGCT 和 LWC-KD 蒸馏算法在 Gowalla 和淘宝2015 数据集上的 Recall@20 结果。

蒸馏策略	数据集	K	增量 1	增量 2	增量 3	平均召回率@20
SGCT	Gowalla	5	0.1608	0.1846	0.2192	0.1882
		10	0.1599	0.1892	0.2321	0.1937
		15	0.1601	0.1956	0.2244	0.1934
		20	0.1557	0.1904	0.2284	0.1915
		25	0.1549	0.1918	0.2259	0.1909
	淘宝2015	5	0	0.0981	0.1007	0.1014	0.1001
		10	0.1040	0.0999	0.1027	0.1022
		15	0.0996	0.1000	0.1040	0.1012
		20	0.1025	0.1005	0.1032	0.1021
		25	0.1018	0.0983	0.1016	0.1006
LWC-KD	Gowalla	5	0.1727	0.2028	0.2346	0.2034
		10	0.1698	0.1978	0.2425	0.2033
		15	0.1663	0.1945	0.0.2411	0.2006
		20	0.1704	0.1992	0.2399	0.2032
		25	0.1707	0.1950	0.2371	0.2009
	淘宝2015	5	0.0993	0.1014	0.1043	0.1017
		10	0.1044	0.1045	0.1052	0.1047
		15	0.1045	0.1039	0.1039	0.1041
		20	0.1017	0.1028	0.1025	0.1023
		25	0.1017	0.1028	0.1041	0.1029

参考文献

Ahrabian, K.; Xu, Y.; Zhang, Y.; Wu, J.; Wang, Y.; 和 Coates, M. 2021年。面向增量学习的结构感知经验回放——适用于基于图的推荐系统。载于 *ACM*国际信息与知识管理会议论文集，2832–2836。Kingma, D. P.; 和 Ba, J. 2015。Adam：一种随机优化方法。*CoRR*, abs/1412.6980。Sun, J.; Zhang, Y.; Ma, C.; Coates, M.; Guo, H.; Tang, R.; 和 He, X. 2019。多图卷积协同过滤。载于电气电子工程师学会数据挖掘国际会议（*ICDM*）论文集，1306–1311。

Wang, Y.；Zhang, Y.；以及 Coates, M. 2021年。面向推荐系统增量学习的图结构感知对比知识蒸馏。载于*ACM*国际信息与知识管理会议论文集，3518–3522。

Xu, Y.；Zhang, Y.；Guo, W.；Guo, H.；Tang, R.；以及 Coates, M. 2020年。GraphSAIL：面向推荐系统的图结构感知增量学习。载于*ACM*国际信息与知识管理会议论文集，2861–2868。