引用:鲁彦,凌松松,钟婧,等. 基于解耦合图卷积的属性补全开发者推荐算法[J]. 聊城大学学报(自然科学版),2025,38(4):485-496.

Citation: LU Yan, LING Songsong, ZHONG Jing, YU Xu. Attribute completion developer recommendation algorithm based on decoupled graph convolution[J]. Journal of Liaocheng University (Natural Science Edition), 2025, 38(4): 485-496.

文章编号 1672-6634(2025)04-0485-12 **DOI** 10.19728/j.issn1672-6634.2024070012

基于解耦合图卷积的属性补全开发者推荐算法

鲁彦[1],凌松松[1],钟婧[2],于旭[1,2]

(1. 青岛科技大学 数据科学学院,山东 青岛 266061;2. 中国石油大学 青岛软件学院,山东 青岛 266580)

摘要 提出基于解耦合图卷积的属性补全开发者推荐算法(ADD)。ADD 由属性推断和属性补全两个模块组成。属性推断模块利用解耦合图卷积在多个潜在空间中推断开发者属性;属性补全模块则填补原始属性的缺失部分。补全后的属性重新输入属性推断模块,两模块循环迭代直至模型收敛。当模型收敛时,属性推断能力最强,补全属性最接近实际情况。实验结果表明,ADD 在 MAE、RMSE、Precision 和 Recall 等指标上显著优于现有模型,有效解决了众包平台开发者推荐中的信息缺失问题。

关键词 开发者推荐;解耦表示学习;属性补全;众包软件开发

中图分类号 TP391.3 **文献标志码** A **开放科学(资源服务)标识码(OSID)**

Attribute completion developer recommendation algorithm based on decoupled graph convolution

LU Yan[1], LING Songsong[1], ZHONG Jing[2], YU Xu[1,2]

(1. School of Data Science, Qingdao University of Science and Technology, Qingdao 266061, China; 2. Shandong, China; 2. Qingdao Institute of Software, China University of Petroleum, Qingdao, Shandong 266580, China)

Abstract The crowdsourced software development model harnesses global developer resources to enable efficient software development, yet information overload on platforms poses challenges in recommending suitable developers. Existing recommendation algorithms rely on developer attributes and interaction behaviors, but incomplete developer profiles in crowdsourced platforms hinder recommendation performance. To address this, we propose an Attribute-Completion Developer Recommendation algorithm via Decoupled Graph Convolution (ADD). ADD comprises two modules: an attribute inference module and an attribute completion module. The attribute inference module leverages decoupled graph convolution to deduce developer attributes across multiple latent spaces, while the attribute completion module fills missing attributes using inferred results. The completed attributes are iteratively fed back into the inference module, forming a cyclic optimization process until convergence. Upon convergence, the model achieves peak attribute inference capability, generating attributes that closely reflect real-world scenarios. Experiments

收稿日期:2024-07-15

基金项目:国家自然科学基金项目(62172249,62472441)资助

通信作者:于旭,男,汉族,博士,教授,博士生导师,研究方向:数据挖掘、智能软件工程,E-mail:yuxu0532@upc.edu.cn。

on real-world datasets demonstrate that ADD significantly outperforms state-of-the-art models in MAE, RMSE, Precision, and Recall metrics, effectively resolving information incompleteness in crowdsourced developer recommendations.

Keywords developer recommendations; disentangled representation learning; attribute completion; crowdsourced software development

0 引言

开发者推荐是众包软件平台的关键流程,旨在为特定任务匹配最合适的开发者。通过分析开发者的技能、经验、可用性和过往表现等显式特征,结合任务需求(如标签:后端、Java 等),实现精准推荐。开发者和任务属性(如个人资料、任务标签)是推荐的基础。现有研究提出属性增强协同过滤模型[1],通过增加偏倚项或建模特征交互来预测开发者偏好,提升推荐效果。这些方法依赖于完整的开发者属性特征向量,以实现更准确的匹配。众包软件开发模式通过整合全球开发者资源,打破了地理限制,显著提升了开发效率和灵活性。然而,平台信息过载问题使得为任务推荐合适开发者变得困难。现有方法在开发者属性不完整的情况下表现不佳,限制了推荐算法的性能。大多数属性推理模型假设属性值是完整的,但众包平台中开发者属性往往存在缺失。例如,部分开发者可能隐藏性别或年龄等信息。属性推理已成为平台的关键任务,涉及开发者搜索、分析和任务语义理解等。近年来,图卷积网络(GCN)在推荐系统中取得显著进展,通过结合节点属性和图结构学习密集的节点嵌入向量,用于属性推理[2]。然而,大多数 GCN 方法假设输入属性完整,无法有效处理缺失值。在端到端训练中,缺失属性通常用固定值填充,导致模型依赖不完整数据更新特征嵌入,影响性能表现。因此,亟需一种能够处理属性缺失的推荐方法,以提升开发者推荐的准确性和可靠性。

本文提出了一种基于解耦合图卷积的属性补全开发者推荐算法(ADD),以解决 GCN 中固定值填充缺失属性的局限性。ADD 通过解耦合图卷积网络(DisenGCN)和属性更新模块,自适应地推断和补全开发者属性,显著提升了推荐系统属性推断的准确性。与传统方法不同,ADD 不再使用固定值补全缺失属性,而是通过调整图卷积嵌入的学习参数,适应属性推理和开发者推荐任务,并利用估计属性值更新图嵌入。实验在两个真实数据集上进行,结果表明,ADD 在开发者推荐任务中性能提升了 5%,在属性推理任务中提升了超过 8%。

1 相关工作

1.1 开发者推荐研究现状

推荐系统是应对信息过载和实现个性化推荐的有效工具,由 Jussi Karlgren 教授于 1990 年首次提出。经过三十多年发展,其应用已从商品推荐扩展到新闻、电商、多媒体等领域。技术上,推荐系统从传统协同过滤[3]和矩阵分解[4],逐步演进到强化学习和深度学习。尽管在电商、社交网络、音乐视频等领域取得了显著进展,但在软件工程领域,尤其是资源配置和协作关系推荐方面的研究仍较为滞后。随着机器学习及深度学习技术在众包软件开发者推荐系统中的广泛应用[5],研究者致力于提高任务与开发者匹配的准确度。Yu 等人[6]提出了一种混合推荐算法,结合显式特征(任务、开发者信息)和隐式特征(任务-开发者成绩矩阵),缓解了描述信息不精确的问题。此外,Yu 等人[7]引入知识图谱,提出基于多关系知识增强的推荐算法,解决了任务-开发者交互数据稀疏的难题。近年来,Zhu 等人[8]提出了一个结合技能属性匹配和主题特征的排序学习框架,通过排序学习算法对开发者进行优先级排序,为软件众包任务推荐最匹配的开发者。Shao 等人[开发了 NNLSI 模型,结合任务特性的分类和数值特征训练神经网络,并利用任务描述训练潜在语义索引模型,以实现精确推荐。此外,新开发者因缺乏历史任务记录难以获得推荐,新任务也难以匹配开发者,导致冷启动问题成为系统主要挑战。针对这一问题,一些研究[9]通过分析 StackOverflow 开发者历史数据,结合 LDA 主题模型和协作投票机制,探索解决方案,推动开发者推荐系统的优化与进步。

1.2 解耦表示学习研究现状

解耦表示学习旨在从交织的行为数据中识别并提取潜在影响因素。早期研究多源于计算机视觉领

域[10]，例如 Higgins 等人[11]提出基于变分自动编码器的分散式表示方法。随着时间推移，社区对如何有效学习交互行为中独立因素的表示方法兴趣渐增。例如，Ma 等人[12]开发的解耦图卷积网络，通过邻居路由机制挖掘图结构数据中的潜在因素，并收集特定特征。在用户潜在意图学习方面，从隐式反馈中提取分离表征成为研究热点。一些研究利用变分自动编码器捕捉用户高级意图以提升推荐效果。例如，基于意图解耦的 DGCF[17]方法通过图神经网络实现解耦表示学习，而 DisenHAN[13]通过异构图注意网络探索用户及项目的解耦表示。KGIN[14]方法则利用物品知识图谱编码用户潜在意图，优化推荐结果。

1.3　图卷积神经网络研究现状

图卷积神经网络（GCN）因其强大的节点特征与图结构表征能力，在推荐系统中受到广泛关注。Wang 等人[15]利用 GCN 捕捉双分图中的高阶连接信息，通过节点特征传播显著提升了基于图的推荐性能。然而，现有研究过度依赖节点描述性特征。针对这一问题，Hu 等人[16]在新闻推荐场景中，将用户与新闻的交互行为建模为图结构，提出了一种结合用户长短期兴趣的图卷积网络模型，有效优化了推荐性能。该研究凸显了图结构数据在挖掘用户行为模式中的价值，为推荐系统发展提供了新视角。

1.4　解耦表示学习、解耦图卷积网络、图卷积网络的关系。

解耦表示学习与图卷积网络（GCN）是解耦图卷积网络的理论基础。解耦表示学习提供了提取数据独立特征的理论框架，而 GCN 提供了处理图结构数据的有效方法。解耦图卷积网络结合两者，利用解耦表示学习指导 GCN 的学习过程，以更好地理解和预测节点行为与属性。在本文提出的基于解耦图卷积的属性补全开发者推荐算法（ADD）中，我们利用解耦图卷积网络推断开发者潜在属性，补全缺失数据，从而提升推荐系统性能。

2　研究方法

2.1　模型描述

本文提出了一种基于解耦合图卷积的属性推理开发者推荐算法，ADD 的整体框架如图 1 所示。ADD 由两个模块组成：（1）属性推断模块；（2）属性补全模块。在属性推断模块中，使用 DisenGCN 推断多个潜在空间中的所有属性。DisenGCN 通过邻居路由机制自适应识别影响开发者和任务节点交互的潜在因素，并利用这些因素进行特征提取。具体来说，DisenGCN 在每一层更新节点的分解表示，通过迭代构建表示并推断属性值。在属性补全模块中，推断出的属性填补原始开发者属性的缺失部分。补全后的属性作为 DisenGCN 的输入反馈到推断模块，两个模块不断循环迭代，直到模型收敛。这种迭代机制使属性推断和补全相辅相成，最终提升属性推断的准确性。在推荐开发者的场景中，目标是预测开发者完成任务的评级。本文将其形式化如下。让 $U=\{u_1,u_2,\cdots,u_n\}$ 和 $V=\{v_1,v_2,\cdots,v_m\}$ 来表示开发者集合和任务集合。对于给定的任务 $v\in V$，开发者推荐模型应该找到一组合适的开发者 $U_v=\{u_1,u_2,\cdots,u_k\}$ 来选择任务 v，其中 $1\leqslant k\leqslant |U_v|$。在常见的推荐系统中，使用评级矩阵 $\boldsymbol{R}\in\mathbf{R}^{M\times N}$ 来表示开发者和任务之间的交互，如果开发者 u 完成了任务 v，则 $r_{uv}=1$，否则评级矩阵为 0。

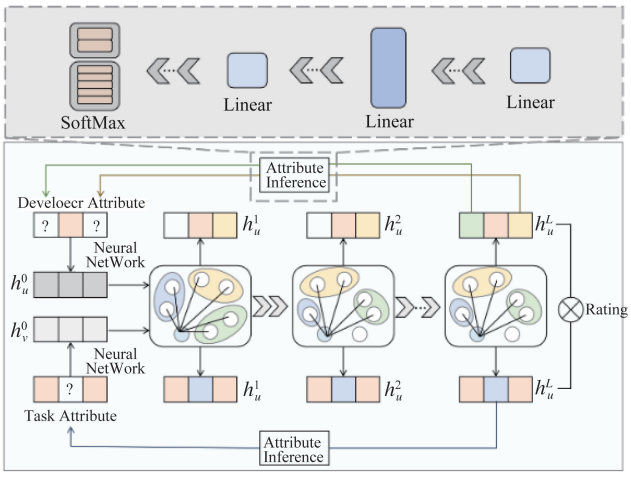

图 1　ADD 框架图

Fig. 1　ADD framework diagram

更详细地说，任何开发者（任务）都有许多来自自身或平台的属性 $A^u=\{a_1,a_2,\cdots,a_{d_u}\}$（$A^v=\{a_1,a_2,\cdots,a_{d_v}\}$），其中 d_u 和 d_v 是属性的数量。推荐系统通常通过复杂的神经网络 f_u 对 A^u（A^v）进行建模，其中 $|f_u|$ 表示神经网络的输出维度。然而，在现实世界中，属性值往往是不完整的，有许多属性值缺失。使用

一个指示矩阵 $\boldsymbol{A}^U \in \mathbf{R}^{d_u \times n}$ 来表示属性是否缺失,其中 $a_{iu}^U = 1$ 表示开发者 u 的第 i 个属性是存在的。反之,$a_{iu}^U = 0$ 表示属性缺失。

为方便叙述,首先就涉及的符号进行说明:U 表示开发者集合;V 表示任务集合;$\boldsymbol{R}$ 表示开发者和任务之间的交互;$\boldsymbol{A}^u$ 表示开发者属性指示矩阵;$\boldsymbol{A}^v$ 表示任务属性指示矩阵;f_u 表示神经网络;G 表示二部图;h_u 表示开发者隐式特征;h_v 表示任务隐式特征;$p_{t,k}$ 表示第 k 个潜在因素而与邻居 t 连接的概率;$\hat{r}_{uv}$ 表示预测分数;s 表示二部图中的开发者;t 表示二部图中与开发者存在交互的任务;$\hat{\boldsymbol{X}}$ 表示开发者属性矩阵;$\hat{\boldsymbol{Y}}$ 表示任务属性矩阵。

根据上文所述,有两个目标:(1) 推断缺失的属性值,使指示矩阵的值全部为 1。(2) 建立已完成属性的模型,以预测开发者和任务的匹配评分。

2.2 基于 DisenGCN 的属性推断模块

直观地说,开发者与任务之间的交互通常会受到他们自身或平台属性的影响。例如,一个完成了许多网络开发任务的开发者一定掌握了 HTML 和 JavaScript 等技术。在本模块中,本文采用 DisenGCN 来隐式地发现此类属性,这些属性可以从开发者与任务的交互中推断出来。本模块的输出 d_u 是任何开发者的维度表示,它将在下一个模块中用于补全缺失的属性。

要想通过 DisenGCN 成功发现缺失属性,首先需要创建一个开发者-任务交互图。在这里,将开发者与任务的历史交互数据表示为一个二部图 $G = \{(u,v) \mid u \in U, v \in V\}$,边 $(u,v) \in G$ 表示开发者 u 和任务 v 之间有历史交互记录。为了更直观地表示属性推断过程,引入了两个变量 s 和 t,分别代表开发者和任务。DisenGCN 框架包含一个邻居路由机制,它能自适应地识别影响开发者和任务节点之间交互的潜在因素,并利用这些因素进行特征提取。值得注意的是,潜在因素可以理解为属性。因此,DisenGCN 能够获取给定图的分解节点表示。DisenConv 层 $f(h)$ 在获得节点 s 及其邻居 $\{t\}$ 在第(1-1)层的节点分解表示 $\boldsymbol{h}_s^l$ 后,会更新节点 s 的分解表示

$$\boldsymbol{h}_s^l = f(\boldsymbol{h}_s^{l-1}, \{\boldsymbol{h}_t^{l-1} : (s,t) \in G\}), \tag{1}$$

式中输出 $\boldsymbol{h}_s^l = [\boldsymbol{h}_{s,1}^l, \boldsymbol{h}_{s,2}^l, \ldots, \boldsymbol{h}_{s,K}^l] \in \mathbf{R}^{K \times d}$ 可以理解为节点 s 在第 l 层 K 个方面下的分解表示,其中 $\boldsymbol{h}_{s,k}^l \in \mathbf{R}^d$ 表示节点在第 k 个方面的表示。

为了学习解耦表征,DisenGCN 需要计算节点 s 因第 k 个潜在因素而与邻居 t 连接的概率 $p_{t,k}$。在这里,需要注意 $\forall k \in [1,K], p_{t,k} \geqslant 0, \sum_{k=1}^{K} p_{t,k} = 1$。$p_{t,k}^l$ 表示邻居 t 参与形成特征表示 $\boldsymbol{h}_{s,k}^l$ 的概率。因此,邻居路由机制通过一个迭代过程来构建表示 $\boldsymbol{h}_{s,k}^l$ 并推断出 $p_{t,k}^l$,如

$$\boldsymbol{h}_{s,k}^l = \frac{\boldsymbol{h}_{s,k}^{l-1} + \sum_{t:(s,t)\in G} p_{t,k}^l \boldsymbol{h}_{t,k}^{l-1}}{\left\| \boldsymbol{h}_{s,k}^{l-1} + \sum_{t:(s,t)\in G} p_{t,k}^l \boldsymbol{h}_{t,k}^{l-1} \right\|}, \tag{2}$$

$$p_{t,k}^l = \frac{\exp(\boldsymbol{h}_{t,k}^{l-1} \cdot \boldsymbol{h}_{s,k}^l / \tau)}{\sum_{k=1}^{K} \exp(\boldsymbol{h}_{t,k}^{l-1}, \boldsymbol{h}_{s,k}^l / \tau)}, \tag{3}$$

式中超参数 τ 调节赋值的难易程度。结果 $\boldsymbol{h}_{s,k}^l$ 对应于第 l 层节点 s 的分解表示。

具体来说,初始嵌入 $\boldsymbol{h}_u^0 = f(\boldsymbol{A}^u)$ 和 $\boldsymbol{h}_v^0 = f(\boldsymbol{A}^v)$ 在经过一个浅层的神经网络映射后,被用作第一个 DisenConv 层的输入,分别对应图 G 中的开发者 u 和项目 v。第 L 层 DisenConv 完成后,第 L 层的隐式分解表示为 $\boldsymbol{h}_u = \boldsymbol{h}_u^L$ 和 $\boldsymbol{h}_v = \boldsymbol{h}_v^L$。通过上述邻居路由机制,可以根据开发者与任务交互图 G 获得开发者和任务的隐式分解表示。

为了更好地说明传播过程,将嵌入传播表述为矩阵形式。假设 $\boldsymbol{A}$ 表示节点为 $(M+N)$ 的开发者项目二部图的邻接矩阵

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{R} & 0^{N \cdot M} \\ 0^{M \cdot N} & \boldsymbol{R}^T \end{pmatrix}。 \tag{4}$$

2.3　属性更新模块

该模块由两部分组成：预测部分和属性更新部分。预测部分预测开发者对项目推荐和属性推理的偏好。属性更新部分根据推断的属性值更新缺失的开发者和项目属性。

预测部分。给定传播深度 L，经过 L 个迭代传播层后，可以得到第 v 个任务的最终嵌入 $\boldsymbol{h}_v^L$ 和开发者 u 的最终嵌入 $\boldsymbol{h}_u^L$。因此，开发者 u 对任务 v 的优先级可以定义为

$$\hat{r}_{uv}=<\boldsymbol{h}_v^L,\boldsymbol{h}_u^L>, \tag{5}$$

式中<，>表示向量内积运算。

在属性推理任务中，充分利用了传播层输出的学习到的开发者和项目嵌入信息。利用最终的嵌入 $\boldsymbol{h}_u^L$ 开发者 u 和最终的嵌入 $\boldsymbol{h}_v^L$ 对于第 v 项，以如下方式推断属性的缺失

$$\hat{x}_u=\mathrm{softmax}(\boldsymbol{h}_u^L\cdot\boldsymbol{W}_x), \tag{6}$$

$$\hat{y}_v=\mathrm{softmax}(\boldsymbol{h}_v^L\cdot\boldsymbol{W}_y), \tag{7}$$

式中 $\boldsymbol{W}_x$ 和 $\boldsymbol{W}_y$ 是需要学习的变换矩阵。

在推导出开发者属性矩阵和任务属性矩阵 $\hat{\boldsymbol{X}}$ 和 $\hat{\boldsymbol{Y}}$ 之后，用推导出的结果迭代地更新缺失的属性值。值得注意的是，已有的属性值保持不变。然后，开发者和项目属性的第$(l+1)$次更新可表示为

$$\begin{aligned}\boldsymbol{X}^{l+1}&=\boldsymbol{X}^l\cdot\boldsymbol{A}^x+\hat{\boldsymbol{X}}\cdot(\boldsymbol{I}^x-\boldsymbol{A}^x),\\ \boldsymbol{Y}^{l+1}&=\boldsymbol{Y}^l\cdot\boldsymbol{A}^y+\hat{\boldsymbol{Y}}\cdot(\boldsymbol{I}^y-\boldsymbol{A}^y),\end{aligned} \tag{8}$$

式中 $\boldsymbol{I}^x\in\mathbf{R}^{d_x\cdot M}$ 和 $\boldsymbol{I}^y\in\mathbf{R}^{d_y\cdot N}$ 是所有元素为 1 的矩阵。将更新后的开发者属性矩阵和项目属性矩阵发送回解耦合图卷积模块进行下一次训练迭代$(l+1)$次迭代，直到收敛。

2.4　模型训练

由于本文的任务包含两个目标，因此优化也包括两个部分：项目推荐损失和属性推理损失。计算各部分的目标函数，并将其组合起来进行最终优化。

(1) 项目推荐损失。我们采用基于 BPR 损失的两两排序[18]，假设观测项目的预测值应该高于未观测项目。目标函数可以表示为

$$\underset{\theta_r}{\mathrm{argmin}}L_r=\sum_{u=0}^{M-1}\sum_{(i,j)\in D_a}-\ln\left[\mathrm{sigmoid}(\hat{r}_{ui}-\hat{r}_{uj})\right]+\lambda\,||\boldsymbol{\theta}_1||^2, \tag{9}$$

式中 θ_r 为项目推荐中设置的参数，$\boldsymbol{\theta}_1$ 为开发者和任务嵌入矩阵。λ 是一个正则化参数，限制了开发者和任务的潜在嵌入矩阵的复杂性。$D_u=\{(i,j)\mid i\in R_u,j\notin R_u\}$ 表示开发者 u 的训练数据，R_u 表示开发者 u 已评级的项集。

(2) 属性推断损失。由于属性推理任务可以表述为一个分类任务，使用交叉熵损失来度量推理属性的误差，其损失函数如

$$\underset{\theta_u}{\mathrm{argmin}}L_u=\mathrm{loss}(\boldsymbol{X},\hat{\boldsymbol{X}},\boldsymbol{A}^x)+\mathrm{loss}(\boldsymbol{Y},\hat{\boldsymbol{Y}},\boldsymbol{A}^y)=\sum_{j=0}^{M-1}\sum_{i=0}^{d_x-1}-x_{ij}\log\hat{x}_{ij}a_{ij}^x+\sum_{j=0}^{N-1}\sum_{i=0}^{d_y-1}-y_{ij}\log\hat{y}_{ij}a_{ij}^y, \tag{10}$$

式中 θ_u 是属性更新模块中设置的参数。

在获得两个任务的目标函数后，通过设置 γ 作为超参数来平衡任务推荐损失和属性推理损失。最终的优化公式为

$$\underset{\theta_u}{\mathrm{argmin}}L=L_r+\gamma L_u, \tag{11}$$

式中 θ_u 是最终目标函数中的所有参数。使用 Pytorch 来实现本文提出的模型。我们利用 Adam 优化器来训练模型。

正如在更新属性部分所强调的，训练过程是一个循环迭代的过程，而不是端到端形式。因为推断的开发者（任务）属性将更新初始化的开发者（任务）属性，所以重复这个过程，直到模型收敛。

2.5 算法流程

算法:ADD

Input:开发者-任务 G,图传播深度 K

Output:解耦图卷积模块参数 θ_r,属性更新模块 θ_u

随机初始化模型参数
设置迭代次数 $l=0$
计算开发人员的初始属性 $\boldsymbol{X}^l$ 和 $\boldsymbol{Y}^l$
进入循环,直到模型收敛:**while** not converged **do**
  选择一个批次的训练数据
  更新 $A^u=\{a_1,a_2,\cdots,a_{d_u}\}$ 和 $A^v=\{a_1,a_2,\cdots,a_{d_v}\}$.
  for $k=0$ to K-1 **do**
  更新 $\boldsymbol{h}_u^L$ 和 $\boldsymbol{h}_v^L$.
  end for
  预测评分矩阵:$\hat{r}_{uv}=<\boldsymbol{h}_v^L,\boldsymbol{h}_u^L>$.
  预测属性值:$\hat{x}_u=\mathrm{softmax}(\boldsymbol{h}_u^L\cdot\boldsymbol{W}_x)$,$\hat{y}_v=\mathrm{softmax}(\boldsymbol{h}_v^L\cdot\boldsymbol{W}_y)$.
  根据 $\arg\min_{\theta}L=L_r+\gamma L_u$ 更新参数
  更新属性 $\boldsymbol{X}^{l+1}$ 和 $\boldsymbol{Y}^{l+1}$.
  $l=l+1$.
end while.
Return θ_r and θ_u.

3 实验

3.1 数据集说明

为了评估本文提出的模型的有效性,在两个收集的数据集上进行了实验:Topcoder-Dataset,ZhuBajie-Dataset。

Topcoder-Dataset:这是在 Topcoder 众包软件平台爬取的具有用户属性的数据集。进一步过滤掉了信息不完整的数据,最后共选取了 10 890 个任务和 2 586 个开发者作为原始数据集。该数据集包含三个用户属性:性别、年龄和职业。这三个属性都是单标签属性。在数据预处理过程中,我们对这些单标签属性进行 One-hot 编码。

ZhuBajie-Dataset:这个数据集包含开发者参与的 1 600 万条任务交互记录,以及任务属性。将被开发者评价的任务视为正反馈,每个任务都有多个属性,如前端、Java、小程序等。因此,开发者和任务的属性是多标签的,我们对其进行 One-hot 编码。

对于所有数据集,过滤掉少于 5 条评级记录的用户。在数据预处理之后,将历史交互按 8∶1∶1 的比例随机分成训练、验证和测试部分。对于每种类型的属性,随机以 α 速率删除属性。由于原始数据集包含了所有的属性值,我们将 α 设为 0.5,随机删除 50%的属性值。表 1 表示了两个数据集经过预处理后的统计情况。

表 1 数据集的统计信息

Tab. 1 Data set statistics

Datasets	Task	Developer	Interaction	Sparsity	Attributes
Topcoder Dataset	10 890	2 586	273 507	98.028%	Gender, Age, Front-end, Java, Applets, Back-end, Android, Ios, Games
ZhuBajie Dataset	11 174	1 951	224 741	97.933%	Gender, Age, Design, UI, Design, Develop, HTML5, JavaScript CSS, AngularJS, Node.js

3.2 数据集说明

对于所有基于潜在嵌入的模型,以均值为 0,标准方差为 0.01 的高斯分布初始化嵌入矩阵。对于这些

基于深度学习的方法，使用 Adam 作为优化方法，在模型学习过程中，初始学习率为 0.001，批量大小为 512。当验证过程中，数据的性能下降时，停止模型的训练。在提出的 ADD 模型中，嵌入维度的大小 D 被设为[16,32,64,128]，发现 $D=64$ 的嵌入获得最好的性能表现。正则化参数 λ 被设为[0.1,0.01,0.001,0.000 1]，当 $\lambda=0.01$ 达到最佳性能。与许多基于 GCN 的推荐模型相似[19]，在区间[0,1,2,3,4]中设置神经网络深度参数 K，并在实验中分析其影响。基线方法中还存在其他参数，通过多次实验调优这些参数，以确保基线方法的最佳性能，来进行公平的比较。

3.3　评价指标

对于开发者推荐任务，为每个开发者推荐可能存在交互的 *Top-N* 个任务。我们采用两种广泛使用的排名评价指标：命中率(HR) H_{R} 和归一化折现累积增益(NDCG) N_{DCG}。具体来说，HR 度量在测试数据中开发者喜欢的 *Top-N* 任务排名列表中成功预测的数量。NDCG 考虑的是任务的命中位置，如果命中的物品在顶部位置，NDCG 会给予较高的分数。在实践中，选择所有未被评分的任务作为每个开发者的负样本，并在排名过程中将其与开发者喜欢的正样本任务相结合。

$$H_{\mathrm{R}}=\frac{1}{N}\sum_{i=1}^{N}\mathrm{hits}(i), \tag{12}$$

式中 N 为样本的数目，可以理解为开发者的需求任务的数目。hits(i) 表示第 i 个需求任务是否包含在模型推荐的任务列表中。若在，则其值为 1；否则为 0。

$$N_{\mathrm{DCG}}=\frac{1}{N}\sum_{i=1}^{N}\frac{1}{\log_2(p_i+1)}, \tag{13}$$

式中 N 为样本的数目，可以理解为开发者的需求任务的数目；(p_i+1) 为第 i 个需求任务在模型推荐的任务列表中的位置；若第 i 个需求项不在推荐列表中，则 $\frac{1}{\log_2(p_i+1)}$ 为 0。

此外，还使用 A_{CC}(Accuracy，ACC)和 M_{AP}(Mean Average Precision，MAP)两个评价指标来评价 ADD 模型的属性推断性能和整体性能，它们表示如

$$A_{\mathrm{CC}}=\frac{|N_{\mathrm{TN}}|+|N_{\mathrm{TP}}|}{|N_{\mathrm{FN}}|+|N_{\mathrm{FP}}|+|N_{\mathrm{TN}}|+|N_{\mathrm{TP}}|}, \tag{14}$$

式中 N_{TP}(真正例)是正确推荐的认为 u 数量，N_{FP}(假正例) 是非推荐任务预测为推荐的数量，N_{TN}(真负例)是正确分类的非推荐任务数量，N_{FN}(假负例) 是将推荐列表中的任务预测为非推荐的数量。

$$M_{\mathrm{AP}}=\frac{1}{n}\sum_{i=1}^{n}\rho_i, \tag{15}$$

式中 n 表示开发者数目，先计算每个开发者的平均精度 $\rho=\int_0^1 P(r)\mathrm{d}r$，然后再求平均值。

3.4　对比方法

本文将 ADD 模型与以下最先进的推荐基线进行比较。BPR[20]：对于基于隐式反馈的推荐算法，它是一个竞争性的潜在因素模型。它设计了一个基于排名的损失函数，假设用户更喜欢他们观察到的物品而不是未观察到的物品。FM[21]：该模型是一个统一的基于潜在因素的模型，利用用户和项目属性。由于 FM 需要完整的用户(项目)属性，因此在实践中使用最佳属性推理模型来补全缺失的属性值。NGCF[22]：这是一种最先进的基于图的协同过滤模型。NGCF 通过聚合前一层邻居的嵌入，迭代地学习用户和物品的表示。PinNGCF[23]：PinNGCF 是一种设计好的基于混合图的推荐模型，其中每个用户(项目)的输入层是嵌入和节点属性的串联。BLA[24]：这是一个结合了社交图上的链接预测和属性推理的联合模型。它给图的边赋予不同的权值，然后利用图的构造进行项目推荐。

本文将 ADD 与以下最先进的属性推断基线进行比较。LP[25]：Label Propagation(LP)是一种经典的基于图结构推断节点属性的算法，通过将每个节点的属性传播到其邻居。该模型没有对不同属性的相关性进行建模。GR[26]：图正则化(GR)扩展了经典的监督学习算法，增加了一个正则化术语，以强制连接节点共享相似的属性值。在实际应用中，监督模型用相邻属性的平均值来填充缺失的属性输入，并采用简单的线性模型作为预测函数。Semi-GCN[27]：这是一种最先进的用于半监督分类的图卷积网络。由于 Semi-GCN 的输

入中存在缺失特征值,所以用邻居值的平均值来补全这些缺失值。

4 实验结果分析

4.1 推荐性能分析

对于开发者推荐任务,专注于为每个 k 开发者推荐 $Top\text{-}k$ 的任务,我们采用了两个广泛使用的基于排名的指标:命中率(HR)和归一累积增益(NDCG)。具体来说,HR 表示测试数据中开发者交互的 $Top\text{-}k$ 排名列表中成功预测的任务的数量。NDCG 会考虑任务的命中排名位置,如果命中的任务位于顶部位置,则会给出更高的分数。在实践中,我们选择所有未评级的任务作为每个开发者的负面样本,并在排名过程中将它们与开发者喜欢的正面任务结合起来。

表 2 和表 3 报告了在 HR@N 和 NDCG@N 上使用不同 $Top\text{-}N$ 值的推荐结果。可以观察到,所有模型都优于 BPR,因为它们利用了额外的数据或使用更高级的特征建模方法。具体来说,NGCF 将开发者-任务交互行为作为输入的基线方法,与 BPR 相比,它显示出了巨大的改进。虽然大多数属性值是不完整的,但是通过用预先训练的模型填充这些缺失的值,属性增强模型比那些不利用属性数据的模型有更好的性能。FM 改进了 BPR,PinNGCF 进一步改进了 NGCF。在比较两种属性增强推荐模型时,基于图的模型 PinNGCF 仍然优于 FM。由于 BLA 是基于经典的属性推理和推荐模型,所以 BLA 的性能优于 BPR 和 FM 两个任务。然而,BLA 的性能不如这些基于 GCN 的基线。因此,也可以从实验结果中看出使用 GCN 进行推荐的优越性,因为 GCN 可以注入高阶图结构,从而更好地进行开发者(任务)的嵌入表示学习,这也是使用 GCN 作为本文所提出的模型的基础模型的原因。将本文提出的 ADD 模型与基线进行比较时,我们发现,在两个数据集上,ADD 优于所有基线。具体的提升因数据集的不同而变化,但总体趋势是相同的。例如,在 Topcoder-Dataset 数据集上,与 HR@10 和 NDCG@10 的最佳基线(即 PinNGCF)相比,ADD 提高了 8%和 7%。由于 PinNGCF 可以看作是一个简化版的 ADD,不需要属性更新和联合建模步骤,结果表明了具有属性更新和联合建模的 ADD 的优越性。

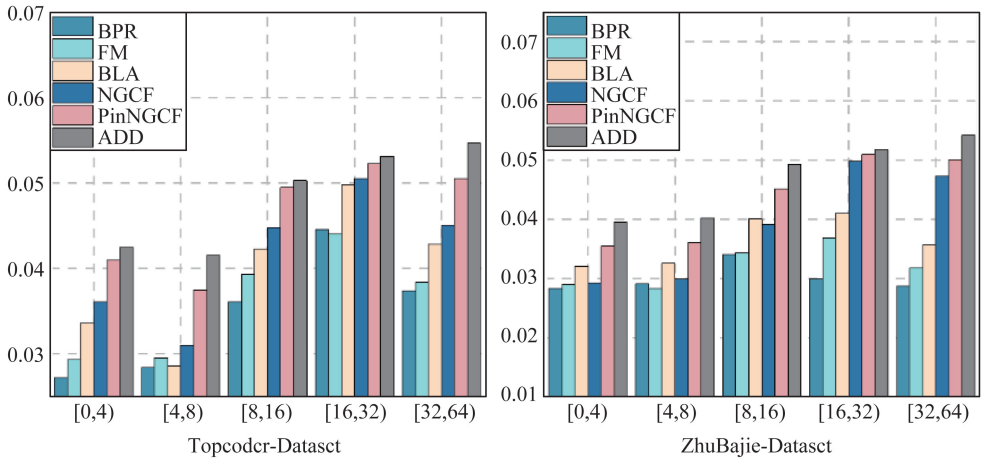

图 2 发者-任务交互记录对模型性能的影响

Fig. 2 The impact of developer-task interaction records on model performa

表 2 不同 Top-N 值进行推荐的 HR@N 比较

Tab. 2 Comparison of recommended HR@N with different Top-N values

Model	Topcoder-Dataset			ZhuBajie-Dataset		
	$N=10$	$N=20$	$N=30$	$N=10$	$N=20$	$N=30$
BPR	0.058 1	0.091 1	0.121 1	0.079 6	0.106 5	0.141 6
FM	0.061 3	0.092 4	0.136 3	0.083 4	0.108 1	0.159 4
BLA	0.064 2	0.097 7	0.136 6	0.098 5	0.114 3	0.159 8

(续)表 2 不同 Top-N 值进行推荐的 HR@N 比较

Tab. 2　Comparison of recommended HR@*N* with different Top-*N* values

Model	Topcoder-Dataset			ZhuBajie-Dataset		
	$N=10$	$N=20$	$N=30$	$N=10$	$N=20$	$N=30$
NGCF	0.073 7	0.113 2	0.142 5	0.097 9	0.132 4	0.166 7
PinNGCF	0.078 1	0.126 5	0.151 2	0.103 3	0.148 2	0.176 9
ADD	0.081 3	0.129 1	0.168 8	0.106 8	0.151 1	0.197 4

表 3　不同 Top-*N* 值进行推荐的 NDCG@*N* 比较

Tab. 3　Comparison of recommended NDCG@*N* with different Top-*N* values

Model	Topcoder-Dataset			ZhuBajie-Dataset		
	$N=10$	$N=20$	$N=30$	$N=10$	$N=20$	$N=30$
BPR	0.032 1	0.041 1	0.421 1	0.034 9	0.044 7	0.458 9
FM	0.033 3	0.042 4	0.436 3	0.036 2	0.046 2	0.475 5
BLA	0.034 2	0.047 7	0.436 6	0.037 2	0.051 9	0.475 8
NGCF	0.043 7	0.053 2	0.542 5	0.047 6	0.057 9	0.591 3
PinNGCF	0.048 1	0. 056 5	0.551 2	0.052 4	0.058 3	0.600 8
ADD	0.051 3	0. 059 1	0.568 8	0.055 9	0.061 2	0.619 9

4.2　属性推理性能分析

在属性推理任务中,使用 ACC 评估单标签属性推理性能,使用 MAP 评估多标签属性推理性能。具体来说,ACC 描述了正确预测的样本占总样本的比例。MAP 描述推断属性的平均精度,广泛应用于多标签分类任务中[27]。对于这两个指标,值越大,性能越好。必须注意,由于数据的限制,不能在每个数据集上同时执行开发者属性推断和任务属性推断。本文提出的 ADD 可以根据可用的数据灵活地预测它们中的任何一个。

表 4　属性推理的性能比较

Tab. 4　Performance comparison of attribute reasoning

Model	Topcoder-Dataset		ZhuBajie-Dataset	
	AGE(ACC)	Gender (MAP)	AGE(ACC)	Gender (MAP)
LP	0.192 1	0.441 1	0.434 9	0.444 7
GR	0.143 3	0.442 4	0.536 2	0.546 2
Semi-GCN	0.154 2	0.447 7	0.637 2	0.651 9
BLA	0.163 7	0.553 2	0.647 6	0.657 9
ADD	0.201 3	0. 659 1	0.715 9	0.661 2

表 4 显示了本文提出的模型与基线相比的性能。所提出的模型在不同的数据集上对不同的属性推理都表现出最佳的性能。例如,本文的模型在 Topcoder-Dataset 的设计、价格和主题属性上分别提高了 4.8%、28.0%和 11.6%。在 ZhuBajie-Dataset 上,本文的模型在性别、年龄和职业属性上分别提高了 5.1%、13.8%和 16.4%。

通过比较用于属性推理的 LP、GR 和 Semi-GCN 的基线,我们发现 GR 和 Semi-GCN 表现为在大多数情况下,性能优于 LP,因为这些模型利用了建模过程中剩余的相关属性。Topcoder-dataset 数据集的价格属性预测有一个例外,LP 是最强的基线。我们猜测一个可能的原因是,价格与其他属性(如性别和语言类型)并不相关,而引入其他属性将为其余基线添加干扰信息。GR 模型和 Semi-GCN 模型在用预先计算的数据填充缺失的属性值时表现出相似的性能。通常,BLA 是联合建模这两个任务的最佳基线。

4.3 超参数对性能的影响

表5总结了不同传播深度 L 下基于解耦合图卷积的属性补全开发者推荐算法(ADD)的实验结果,其中 L 的取值为{0, 1, 2, 3, 4},并记录了不同 L 值下的 HR@10 和 NDCG@10 指标。特别地,当 $L=0$ 时,ADD 未融入任何图结构到模型学习中。随着 L 值的增加,模型逐步引入更高阶的图结构来进行节点表示,从而在所有数据集上实现了显著的性能提升。这一改进强调了一阶邻居信息在减轻数据稀疏性问题中的关键作用。

然而,一个明显的趋势是,尽管性能最初随着 K 的增加而提升,但超过某一阈值后开始下降。具体来说,Topcoder 数据集在 $L=3$ 时达到最优性能,而在 ZhuBajie 数据集上,$L=2$ 时观察到性能的峰值。这种差异可能归因于 Topcoder 数据集极度稀疏,其中仅 1.972%的密度表明,聚集更多的邻居有助于更有效的节点嵌入学习。相反,对于 ZhuBajie 数据集,过多地引入 3 阶邻居导致图过度平滑,因此性能相对于 $L=2$ 有所下降。当 L 增加到 4 时,在所有数据集上都出现了过拟合现象,这归因于图卷积作为特征平滑操作,过度应用导致表示过度平滑。这一现象在其他基于 GCN 的推荐任务中也有观察到,强调了在引入图层以避免过度平滑并实现模型最优性能方面需要采取平衡的方法。

表5 不同传播深度K在两个数据集上的性能比较

Tab. 5 Performance comparison of different propagation depths K on two data sets

Depths	Topcoder-Dataset		ZhuBajie-Dataset	
	HR@10	NDCG@10	HR@10	NDCG@10
$L=0$	0.0635	0.0355	0.212	0.194
$L=1$	0.0732	0.0397	0.224	0.213
$L=2$	0.0831	0.0452	**0.228**	**0.219**
$L=3$	**0.0864**	**0.0464**	0.215	0.193
$L=4$	0.0843	0.0456	0.223	0.174

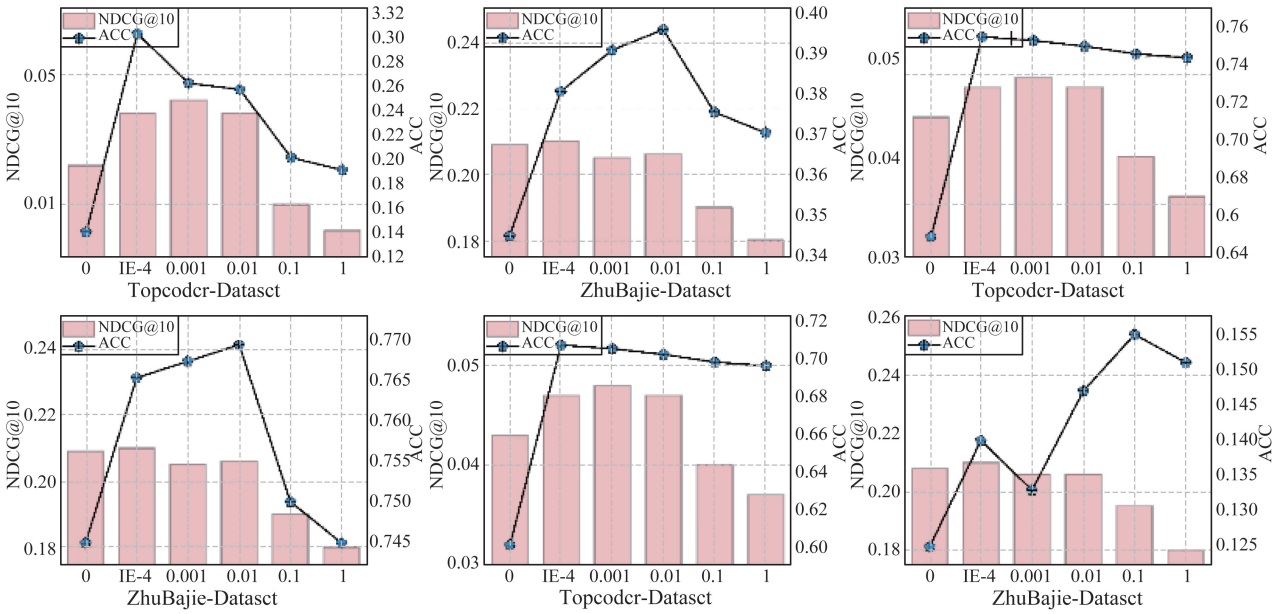

图3 在两个不同数据集上不同 γ 的属性推理性能

Fig. 3 Attribute reasoning performance with different γ on two different datasets

公式(11)所控制着 ADD 中评级损失函数和属性推理损失之间的相对权重。随着γ的增加,模型越来依赖于属性推理损失来进行联合任务预测。特别地,当$\gamma=0$时,属性推理损失不再起作用,预测完全依赖于偏好预测损失。本节将探讨任务平衡参数 γ 如何影响开发者推荐和属性推理两个任务的性能。更具体地,

$\gamma=0$ 意味着仅通过基于评级的损失来进行开发者推荐，而 $Lr=0$ 则表明仅考虑属性推理损失来执行属性推理。这样无法有效捕捉两任务之间的关联性，导致这两个任务被独立建模而未能实现互补优化。

图 3 展示了不同 γ 值下两个任务性能的变化，其中右侧 y 轴代表属性推理结果，左侧 y 轴反映了开发者推荐的性能。从图中可以看出，x 轴最左侧部分揭示了在没有任何任务相关建模下的性能（即项目推荐时 $\gamma=0$，属性推理时 $Lr=0$）。随着 γ 从 0 增至 0.001，两任务的性能显著优于 x 轴最左侧的情况，这强调了在统一框架下对两个任务进行建模的有效性。γ 值继续增加时，两任务的表现首先是提升，但当 γ 达到 0.1 或 1 时，大部分情况下性能开始下降。不同任务在不同 γ 值下达到最优性能，且不同类型属性的最优性能亦随 γ 值变化。因此，在实践中，应为每个任务选择最合适的 γ 参数以优化性能。

5　结论

本文提出的基于解耦合图卷积的属性补全开发者推荐算法（ADD）来解决众包软件平台中常见的属性缺失问题。通过迭代进行图嵌入学习和属性更新，ADD 算法的主要贡献在于能够有效地结合已有的属性值和通过模型估计得到的属性值，自适应调整图学习过程。这一机制不仅为属性推理和开发者推荐提供了弱监督信号，还增强了模型对缺失数据的处理能力。在两个真实世界数据集上的实验验证了 ADD 模型的有效性，显示出其在推荐准确性和属性推理方面的优越性。尽管 ADD 算法在处理属性缺失问题上表现出色，但它可能在面对极端稀疏数据或属性高度不相关的情况时表现不佳。此外，模型的计算复杂度较高，可能需要更多的计算资源。未来的工作将聚焦于探索更复杂的图结构表示和学习机制，以更好地捕捉开发者属性与任务需求之间的细微关系。此外，考虑到众包软件平台的动态性，未来研究还将尝试引入时间序列分析，以实现对开发者能力变化的实时跟踪和预测。

参　考　文　献

[1] CHEN T C, ZHANG W N, LU Q X, et al. SVDFeature: a toolkit for feature-based collaborative filtering[J]. Journal of Machine Learning Research, 2012, 13(1): 3619-3622.

[2] HAMILTON W L, YING R, LESKOVEC J. Inductive representation learning on large graphs[EB/OL]. arXiv preprint arXiv:1706, 02216,2017.

[3] BALABANOVIĆ M, SHOHAM Y. Fab: content-based, collaborative recommendation[J]. Communications of the ACM, 1997, 40(3): 66-72.

[4] LIU Z, FENG X D, WANG Y C, et al. Self-paced learning enhanced neural matrix factorization for noise-aware recommendation[J]. Knowledge-Based Systems, 2021, 213: 106660.

[5] ZHANG W, ZHAO J P, PENG R, et al. SusRec: an approach to sustainable developer recommendation for bug resolution using multi-modal ensemble learning[J]. IEEE Transactions on Reliability, 2022, 72(1): 61-78.

[6] 于旭，何亚东，杜军威，等. 一种结合显式特征和隐式特征的开发者混合推荐算法[J]. 软件学报，2022，33(5)：1635-1651.

[7] 杜军威，王昭哲，于旭，等. 基于多关系知识增强的开发者推荐算法[J]. 电子学报，2023，51(11)：3111-3119.

[8] ZHU J G, SHEN B J, HU F H. A learning to rank framework for developer recommendation in software crowdsourcing[C]//Proceedings of the Asia-Pacific Software Engineering Conference, 2015: 285-292.

[9] SHAO W, WANG X N, JIAO W P. A developer recommendation framework in software crowdsourcing development[C]//Software Engineering and Methodology for Emerging Domains: 15th National Software Application Conference, NASAC 2016, Kunming, Yunnan, November 3-5, 2016, Proceedings. Springer Singapore, 2016: 151-164.

[10] HIGGINS I, MATTHEY L, PAL A, et al. Beta-VAE: learning basic visual concepts with a constrained variational framework[C]//Proceedings of the International Conference on Learning Representations, 2016.

[11] HIGGINS I, AMOS D, PFAU D, et al. Towards a definition of disentangled representations[EB/OL]. arXiv preprint arXiv:1812.02230, 2018.

[12] MA J X, ZHOU C, CUI P, et al. Learning disentangled representations for recommendation[EB/OL]. arXiv preprint arXiv:1910.

14238, 2019.

[13] WANG Y F, TANG S Y, LEI Y T, et al. DisenHAN: disentangled heterogeneous graph attention network for recommendation[C]// Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020. DOI: 10. 1145/3340531. 3411996.

[14] WANG X, HUANG T L, WANG D X, et al. Learning intents behind interactions with knowledge graph for recommendation[C]//Proceedings of the Web Conference 2021, 2021: 878-887.

[15] WANG X, HE X N, WANG M, et al. Neural graph collaborative filtering[C]//Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019: 165-174.

[16] HU L M, LI C, SHI C, et al. Graph neural news recommendation with long-term and short-term interest modeling[J]. Information Processing & Management, 2020, 57(2): 102142.

[17] VAN DEN BERGR, KIPF T N, WELLING M. Graph convolutional matrix completion[EB/OL]. arXiv preprint arXiv: 1706. 02263, 2017.

[18] MA J, CUI P, KUANG K, et al. Disentangled graph convolutional networks[C]// Proceedings of the International Conference on Machine Learning, 2019, 97: 4212-4221.

[19] BILLSUS D, PAZZANI M J. A hybrid user model for news story classification[C]// Proceedings of the 7th International Conference on User Modeling, Adaptation, and Personalization, 1999: 99-108.

[20] WANG P P, LI L, WANG R, et al. Learning persona-driven personalized sentimental representation for review-based recommendation [J]. Expert Systems with Applications, 2022, 203: 117317.

[21] RENDLE S. Factorization machines[C]//Proceedings of the IEEE International Conference on Data Mining, 2010: 995-1000.

[22] GUO H F, TANG R M, YE Y M, et al. DeepFM: a factorization-machine based neural network for CTR prediction[EB/OL]. arXiv preprint arXiv: 1703. 04247, 2017.

[23] YING R, HE R N, CHEN K F, et al. Graph convolutional neural networks for web-scale recommender systems[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018: 974-983.

[24] YANG C, ZHONG L, LI L J, et al. Bi-directional joint inference for user links and attributes on large social graphs[C]//Proceedings of the 26th International Conference on World Wide Web Companion, 2017: 564-573.

[25] ZHUŕ X, GHAHRAMANIŕH Z. Learning from labeled and unlabeled data with label propagation[J]. ProQuest Number: Information To All Users, 2002. DOI: 10. 1109/IJCNN. 2002. 1007592.

[26] BELKIN M, NIYOGI P, SINDHWANI V. Manifold regularization: a geometric framework for learning from labeled and unlabeled examples[J]. Journal of Machine Learning Research, 2006, 7(11): 2399-2434.

[27] DENG J, DONG W, SOCHER R, et al. ImageNet: a large-scale hierarchical image database[C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2009: 248-255.

(责任编辑:赵海军)