计算机科学
COMPUTER SCIENCE

MDGRec:基于多元关系融合的移动应用第三方库推荐方法

陈煜涵, 王健, 李段腾川, 郑超, 李兵

引用本文

陈煜涵, 王健, 李段腾川, 郑超, 李兵. MDGRec:基于多元关系融合的移动应用第三方库推荐方法[J]. 计算机科学, 2025, 52(11): 320-329.

CHEN Yuhan, WANG Jian, LI Duantengchuan, ZHENG Chao, LI Bing. MDGRec:Multi-relation Aware Third-party Library Recommendation with Dual Graph NeuralNetworks for Mobile Application Development [J]. Computer Science, 2025, 52(11): 320-329.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.241200129

MDGRec:基于多元关系融合的移动应用第三方库推荐方法

陈煜涵　王　健　李段腾川　郑　超　李　兵
武汉大学计算机学院　武汉 430072
(lctaba@whu.edu.cn)

摘　要　第三方库推荐系统旨在向开发者推荐合适的第三方库,以提高移动应用的开发效率。然而,现有的基于图神经网络的方法大多在一个异构交互图中同时传播移动应用和第三方库的节点信息,存在数据不平衡和特征混淆的问题。此外,现有方法忽视了第三方库推荐场景关系的复杂性,限制了推荐准确性。为此,提出了一种基于多元关系融合的移动应用第三方库推荐方法。模型使用双图结构分别对移动应用和第三方库进行建模,生成相应的嵌入向量。在此基础上,模型融合了第三方库推荐场景中的多元关系,在不同关系维度上传播节点信息,并使用自适应权重刻画不同关系在信息传播中的贡献,以生成细粒度的节点特征。在两个真实世界数据集上的实验结果表明,所提方法在各项指标上优于主流的基线模型。

关键词:推荐系统;图神经网络;多元关系;第三方库;移动应用
中图分类号　TP311

MDGRec:Multi-relation Aware Third-party Library Recommendation with Dual Graph Neural Networks for Mobile Application Development

CHEN Yuhan,WANG Jian,LI Duantengchuan,ZHENG Chao and LI Bing
School of Computer Science,Wuhan University,Wuhan 430072,China

Abstract　Third-party library(TPL) recommendation systems are designed to help developers select suitable libraries,thus improving the efficiency of mobile application(App) development. Most existing methods based on graph neural networks typically propagate information for both App and TPL nodes within a single heterogeneous interaction graph,leading to issues of data imbalance and feature confusion,limiting the recommendation accuracy. Moreover,these methods often fail to account for the complex relationships inherent in the context of TPL recommendation. To overcome these limitations,this paper proposes a multi-relation aware third-party library recommendation method with dual graph neural networks for mobile application development (MDGRec). The model employs a dual graph structure to separately model Apps and TPLs,generating distinct embeddings. Based on this,the model incorporates multiple relationships and uses adaptive weights to capture the contribution of each relation in information propagation,constructing fine-grained node representations. Experimental results on two real-world datasets show that the proposed model surpasses mainstream baselines across all metrics.

Keywords　Recommender systems,Graph neural network,Multi-relation,Third-party library,Mobile application

1　引言

近年来,移动互联网的快速发展带来了对移动应用(Application,App)的巨大需求。截至 2024 年第二季度,Google Play 上的 App 数量已超过 168 万[1)]。随着越来越多的 App 被开发,应用提供商之间的竞争愈发激烈。这意味着应用提供商必须提高 App 开发速率,以在激烈竞争的市场中占据一席之地。第三方库(Third-party Library,TPL)是一系列功能代码的合集,包括 UI 接口、社交媒体、广告推送等[1-4]。通过利用 TPL,开发者可以快速复用这些功能,降低实现复杂功能的成本[5-6]。然而,随着越来越多的 TPL 被开发,找到合适的 TPL 变得愈发耗时[7]。如果能向开发者推荐合适的 TPL,将显著提高开发效率。

近年来,众多专家学者提出了不同的 TPL 推荐方法[8-11]。基于协同过滤(Collabrative Filtering,CF)的方法[5,12-14]利用历史交互探索 App 和 TPL 之间的相似性,进而推荐合适的 TPL。然而,CF 仅限于识别 App 和 TPL 之间的相似性,缺乏显式建模它们之间协作信号的能力[15]。为突破

1) https://www.statista.com/statistics/266210

到稿日期:2024-12-17　返修日期:2025-03-24
基金项目:国家自然科学基金重点项目(62032016)
This work was supported by the Key Program of the National Natural Science Foundation of China(62032016).
通信作者:李兵(bingli@whu.edu.cn)

这一限制，一些学者[10-11,16]采用基于 App 和 TPL 交互图的图神经网络(Graph Neural Network，GNN)，以提取更高阶的邻域信息。然而，App 和 TPL 是两种不同类型的节点，它们所具备的特征不尽相同，且 App 的数量远大于 TPL 的数量。在单个交互图中同时传播两种不同节点的信息，占比少的 TPL 节点特征会被占比多的 App 节点特征同化，导致节点特征混淆。此外，许多现有的商品推荐系统建模了复杂的用户和物品之间的关系。Su 等[17]在模型中融入了用户信任关系与物品相似关系。EMRIGCN[18]建模了物品之间的可替代关系与互补关系，使得模型能够更细粒度地刻画节点特征，从而提升推荐性能。而上述基于 GNN 的 TPL 推荐方法忽略了 App 和 TPL 之间的复杂关系，仅依赖单一交互关系传播邻域信息，限制了模型对节点特征的表达能力。

针对上述问题，本文构建了一个基于多元关系融合的移动应用第三方库推荐模型(Multi-Relation Aware Third-party Library Recommendation Method with Dual Graph Neural Networks for Mobile Application Development，MDGRec)。该模型注重刻画 TPL 推荐场景中的 3 种关系。1)如果两个 App 同时使用了某个 TPL，表明它们在功能需求上存在一定的相似性，定义这种关系为 App 之间的共现关系。共现关系有助于模型识别功能相似 App，从而推荐与 App 功能更相关的 TPL。2)如果两个 TPL 被一个 App 同时使用，那么这两个 TPL 之间可能存在某些依赖关系或者功能上的协同作用，定义这种关系为 TPL 之间的互补关系。利用互补关系，可以挖掘 TPL 之间的潜在协作模式，从而向 App 推荐与已有 TPL 互补的 TPL。3)如果两个 TPL 没有同时被任何 App 使用，那么它们之间在功能或者兼容性上可能存在冲突，定义这种关系为 TPL 之间的互斥关系。引入互斥关系，能够减少推荐互斥的 TPL，从而提升推荐的合理性。MDGRec 将这 3 种关系作为边，并赋予这些边基于数据统计量的自适应权重，使得节点信息能够在不同关系维度上传播，从而得到更细粒度的节点特征。此外，MDGRec 采用了双图结构，使得 App 和 TPL 的节点信息能够在不同的特征空间中传播，解决了单图中 App 和 TPL 数据不平衡和特征混淆的问题，进一步提升了 TPL 推荐的准确性。

本文的主要贡献如下：

1)提出了 MDGRec 模型，通过构建双图结构分别建模 App 和 TPL，解决了 App 和 TPL 之间的特征混淆与数据不平衡问题；

2)融合了 TPL 推荐场景中的多元关系，并基于数据统计量赋予自适应权重，细粒度地传播 App 和 TPL 的节点特征，提高了推荐的准确性；

3)在真实世界数据集 MaLib[1)] 和 APPBrain 上进行了大量的实验，表明 MDGRec 在各项指标上都优于基线模型。

2 相关工作

2.1 基于 CF 的推荐方法

随着 CF 在商品推荐中被广泛应用[19-20]，许多学者将 CF 技术引入了 TPL 推荐领域。Thung 等[8]首次将 CF 引入 TPL 推荐系统，结合关联规则挖掘和 CF 技术发掘 TPL 之间的联合使用模式，从而进行 TPL 推荐。在此基础上，Nguyen 等[5]基于历史交互数据和 TF-IDF 算法计算开源项目与 TPL 之间的相似性，进一步提升推荐效果。Ren 等[21]在 CF 的基础上融入了 App 文本信息，捕捉历史调用数据外 App 与 TPL 间的潜在关联，从而提升推荐准确率和泛化能力。随着矩阵分解(Matrix Factorization，MF)方法在传统推荐系统中取得显著成效[22-23]，He 等[12]提出了 LibSeek，该方法在 MF 的基础上引入 TPL 的邻域信息，并结合加权技术缓解长尾效应，在推荐准确性和多样性方面均取得了良好的效果。

基于 CF 的方法凭借着简单高效的优势在 TPL 推荐中取得了一定效果，尤其在大规模数据集上展现了良好的计算效率。但这些方法仅隐式地利用了 App 与 TPL 的交互信息，难以充分建模 App 和 TPL 的特征，存在一定的瓶颈。

2.2 基于 GNN 的推荐方法

随着 GNN[24-25]的兴起，将 GNN 与推荐系统相结合成为一个新的探索方向[26-27]。基于 GNN 的 NGCF[15]和 LightGCN[28]，已在商品推荐领域取得显著成效。NGCF 利用 GNN 在异构的用户-项目图中传播节点信息，通过多跳邻居节点提取 CF 方法难以捕捉的高阶交互信息，展现出了比基于 CF 的方法更加强大的特征建模能力，尤其是在数据稀疏的场景。在 NGCF 的基础上，Grec[10]首次将 GNN 应用于 TPL 推荐场景中，能够更好地捕捉 App 和 TPL 之间的高阶邻域特征，提升了推荐的效果。然而，由于推荐系统通常仅使用 ID 编码，没有具体的语义信息，NGCF 中的特征变换和非线性激活不仅不能带来好处，还增加了模型的复杂度，导致模型训练难度提升，限制了模型的性能。基于上述局限性，LightGCN 简化了 NGCF 中的 GNN 结构，移除了特征变换和非线性激活模块，加快训练速度的同时提升了推荐的准确性。Jin 等[11]将 LightGCN 扩展为 NLA-GNN 并运用于 TPL 推荐场景，使用一种嵌入同时表征 App 和 TPL，缓解了 App 和 TPL 之间特征异质性的影响，并基于交互关系汇聚交替信息，获取 App 和 TPL 的最终嵌入。NLA-GNN 在仅使用交互关系传播信息的场景下表现优异，然而其结构无法同时建模多种关系。

上述方法虽然都取得了良好的效果，但仅仅是将商品推荐算法进行了迁移，没有考虑 TPL 推荐场景的特异性，限制了模型效果的提升。为了进一步提高推荐精度，KG2Lib[29]与 PyRec[30]将知识图谱与 GNN 结合，使用丰富的额外知识信息提升了 TPL 推荐的准确性，且其在冷启动场景下也有良好的适应能力。这些方法依赖特定的领域知识和知识图谱的构建质量，适用于辅助信息丰富的推荐场景。

2.3 基于 Dual GNN 的推荐方法

随着推荐场景复杂性的不断增加，越来越多的实体和关系被引入。单 GNN 在不同节点之间传播信息时，未能考虑不同类型节点之间信息的差异，往往会引入额外的噪声[31]。同时，信息的重复传播使得不同类别的节点表示难以

1) https://github.com/malibdata/MALib-Dataset

区分[32]。因此，许多研究引入了双图神经网络(Dual GNN)以建模不同的节点和关系[33-35]。DVGRL[36]使用 Dual GNN 分别构建历史交互和社交网络，通过融合跨领域知识提升推荐性能，适用于包含多种用户交互模式的社交推荐场景。SComGNN[37]和 ECGN[38]利用 Dual GNN 捕捉物品之间的互补关系，为物品特征表示提供更全面的视角，尤其在物品间存在显著互补性的场景中表现优异。在 TPL 推荐场景中，HG-NRe[39]将交互图分解为两个同质图，并使用基于相似度的边过滤方法缓解了数据的长尾分布问题，突破了 App 和 TPL 之间异质特征融合的局限性，取得了良好的效果。

受上述方法启发，本文使用双图结构解决 TPL 推荐中特征混淆和数据不平衡的问题，并在此基础上融合多元关系传播节点特征，充分利用了 App 和 TPL 之间的复杂联系，提升了模型在 TPL 推荐场景下对 App 和 TPL 的节点表征能力。

3 MDGRec

MDGRec 模型结构如图 1 所示，总体上可分为 4 个部分。1)图构建层：通过 App 与 TPL 的交互关系构建 App 共现图和 TPL 互补互斥图；2)嵌入层：用于提取 App 和 TPL 的初始特征；3)传播层：利用 App 共现图、TPL 互补互斥图传播 App 之间以及 TPL 之间的节点信息，提取高阶节点特征；4)聚合与预测层：聚合嵌入层得到的初始特征与传播层得到的高阶特征，生成最终节点特征，并利用最终特征预测 App 与 TPL 之间的评分，从而完成 TPL 推荐。

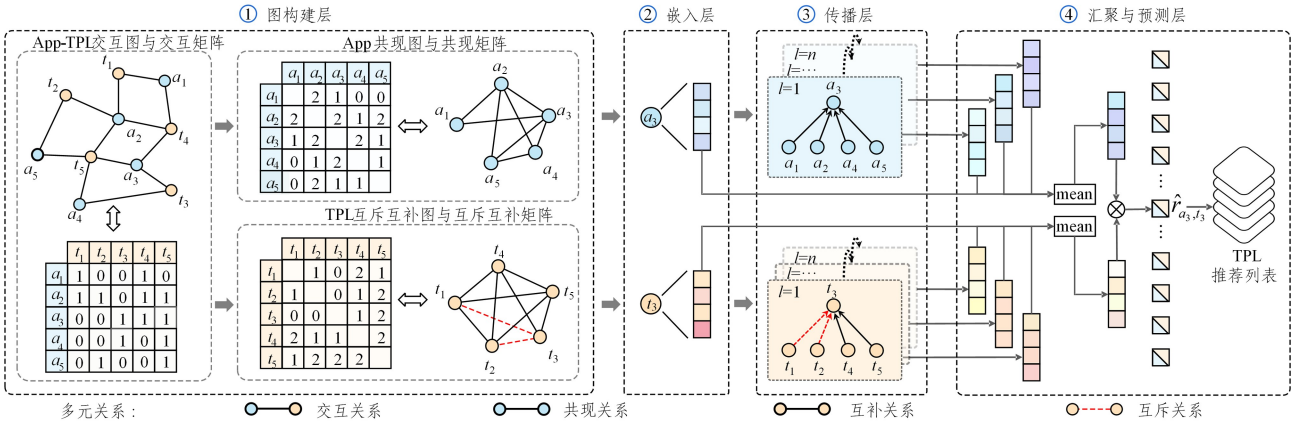

图 1 MDGRec 模型结构

Fig. 1 Architecture of MDGRec

3.1 图构建层

3.1.1 App 共现图

如图 1 中的图构建层所示，MDGRec 构造了一个无向图来表示 App 之间的共现关系，记为$G_A=(V_A,E_A)$，其中V_A代表图中的节点集合，每个节点代表一个 App；E_A是图中边的集合，$E_A=\{(u,v,w_{u,v})\mid u\in A,v\in A,w_{u,v}\in N^+\}$，其中 A 是所有 App 的集合，并且满足 App u 和 v 调用了同一个 TPL t，称 u 和 v 为邻居。将所有满足该条件的 TPL 集合记为$T_{u,v}$。$w_{u,v}$是边的权重，权重值与集合$T_{u,v}$中元素数量相等。如图 1 中示例所示，在 App-TPL 交互图中，App a_1，a_2同时调用了 TPL t_1，t_4，在 App-TPL 交互矩阵中，它们对应位置的元素为 1。根据上述对边权重的定义，$T_{a_1,a_2}=\{t_1,t_4\}$，因此在 App 共现图中，App a_1，a_2之间的边权重为 2，在 App 共现矩阵中，它们对应位置的元素为 2。基于共现图，App 可以通过共现邻居节点获取信息，以丰富本节点的特征表示。

3.1.2 TPL 互斥互补图

与 App 共现图类似，MDGRec 构造了一个具有两种边类型的无向图来表示 TPL 之间的关系，记为$G_T=(V_T,E_T)$。其中，V_T代表图中的节点集合，每个节点代表一个 TPL。E_T是图中边的集合，包含互补边与互斥边，分别记为E_T^+和E_T^-。其中，互补边E_T^+与 App 共现图中的边类似，$E_T^+=\{(s,t,w_{s,t}^+)\mid s\in T,t\in T,w_{s,t}^+\in N^+\}$，$T$ 是所有 TPL 的集合，并且满足 TPL s 和 t 同时被 App a 调用，称 s 和 t 为互补邻居。将所有满足该条件的 App 集合记为$A_{s,t}$，互补边权重$w_{s,t}^+$的值与集合$A_{s,t}$中元素数量相等。

与 App 共现图不同，为了刻画 TPL 之间的互斥关系，本文在 TPL 图中引入了互斥边。互斥边E_T^-的定义如下：$E_T^-=\{(s,t,w_{s,t}^-)\mid s\in T,t\in T,w_{s,t}^-\in R^+\}$，其中 TPL s 和 t 从未被任何一个 App 同时调用，称 s 和 t 为互斥邻居。若两个被广泛使用的 TPL 从未被同一个 App 使用，则它们之间很可能存在互斥关系；若两个小众的 TPL 没被任何 App 同时使用，这更可能是因为它们本身就很少被使用，而非存在互斥关系。

因此，本文考虑使用 TPL 的流行度为 TPL 之间的互斥边加权。TPL 的流行度越大，TPL 之间的互斥边便被赋予更高的权重。TPL 的流行度定义为 TPL t 被所有 App 调用的次数，记为P_t。由于 TPL 的流行度呈现明显的长尾效应，头部 TPL 调用次数可达上万次，而尾部 TPL 仅被调用过几次。若直接使用流行度加权，将导致头部 TPL 主导模型的训练过程[40]。为了缓解这一问题，考虑使用 log 函数压缩P_t，使其落到一个合理的区间内，以减少极值对模型的影响。对于两个互斥的 TPL s 和 t，它们之间互斥边权重的计算式如下：

$$w_{s,t}^-=\log(P_s)\cdot\log(P_t) \tag{1}$$

在 GNN 信息传播中，利用互斥边可以使 TPL 学习到与其相斥的特征，从而在推荐过程中尽量避免选择冲突的 TPL。为了实现这一目标，MDGRec 基于负权值进行互斥信息的传播，使 TPL 能够有效学习到相斥特征。

3.2 嵌入层

与许多推荐算法类似[10,12,20,22]，如图 1 的中嵌入层所示，

MDGRec 将 App(TPL)映射到一个高维空间中，使用一个嵌入向量作为 App(TPL)的初始特征表示。记 App(TPL)的嵌入为$\boldsymbol{e}_a \in \mathbb{R}^d$($\boldsymbol{e}_t \in \mathbb{R}^d$)，其中 d 是特征嵌入的维度。对于所有 App(TPL)，其嵌入向量集合可以写作矩阵形式：

$$\boldsymbol{E}_A = [\boldsymbol{e}_{a_1}, \boldsymbol{e}_{a_2}, \cdots, \boldsymbol{e}_{a_N}] \tag{2}$$

$$\boldsymbol{E}_T = [\boldsymbol{e}_{t_1}, \boldsymbol{e}_{t_2}, \cdots, \boldsymbol{e}_{t_M}] \tag{3}$$

其中，N 是 App 的数量，M 是 TPL 的数量。App 和 TPL 的嵌入向量是本文模型所有的可训练参数，本文将采用端到端优化的方式获得 App 和 TPL 的最佳嵌入。

3.3 传播层

3.2 节中提到的嵌入向量仅仅表示了 App(TPL)的固有特征，而忽略了 App(TPL)之间可能存在的结构特征。对于 App 来说，结构特征包括功能相似性、协作性等；对于 TPL 来说，结构特征包括功能互补性、互斥性等。因此，本文使用 GNN 挖掘 App(TPL)之间的结构关系。通过在邻居节点之间传播信息，GNN 能够弥补 App(TPL)初始嵌入在特征描述上的不足，从而更精准地刻画 App(TPL)特征。

3.3.1 App 共现图传播

在 App 共现图中，App 节点通过 App 共现边进行信息传播。MDGRec 将 App 的初始嵌入作为 GNN 的输入，通过信息聚合得到高阶嵌入。考虑 n 层 GNN，第 l 层的输入记为$\boldsymbol{E}_A^{l-1}$，输出记为$\boldsymbol{E}_A^l$。将 App 初始嵌入作为 GNN 第一层的输入，即$\boldsymbol{E}_A^0 = \boldsymbol{E}_A$。在第 l 层，对于某个特定的 App u，来自其邻居的信息可以表示为：

$$\boldsymbol{m}_{u \leftarrow v}^l = \frac{w_{u,v}}{\sqrt{|\mathcal{N}_u| \, |\mathcal{N}_v|}} \boldsymbol{e}_v^{l-1} \tag{4}$$

其中，$w_{u,v}$是 App 共现图中节点和之间的边权重，$\boldsymbol{e}_v^{l-1}$是邻居节点在第 $l-1$ 层得到的嵌入。$1/\sqrt{|\mathcal{N}_u| \, |\mathcal{N}_v|}$是图拉普拉斯正则项，$\mathcal{N}_u$和$\mathcal{N}_v$分别代表 u 和 v 的邻居节点集合。如果一个节点邻居数量很多，每个邻居对其的影响将会变小，对应 $1/\sqrt{|\mathcal{N}_u|}$。它对邻居的影响也会被均摊，对应 $1/\sqrt{|\mathcal{N}_v|}$。该项用于削弱高连接度节点的影响，保持模型对所有节点贡献的敏感度。对节点 u，在第 l 层的特征嵌入是其所有邻居节点传播而来的信息总和：

$$\boldsymbol{e}_u^l = \alpha \sum_{v \in \mathcal{N}_u} \boldsymbol{m}_{u \leftarrow v}^l = \alpha \sum_{v \in \mathcal{N}_u} \frac{w_{u,v}}{\sqrt{|\mathcal{N}_u| \, |\mathcal{N}_v|}} \boldsymbol{e}_v^{l-1} \tag{5}$$

式(5)表示节点通过接收来自邻居节点的信息丰富自身特征。其中，α 是超参数，用于控制共现关系在信息传播中对节点特征的影响。

3.3.2 TPL 互补互斥图传播

TPL 图中存在互补边和互斥边，TPL 在互补边上的信息传播方式与 App 在共现边上的传播类似。在第 l 层，对于某个特定的 TPL，它接收的来自其互补邻居的互补信息为：

$$\boldsymbol{m}_{t \leftarrow s}^{l\,+} = \frac{w_{s,t}^+}{\sqrt{|\mathcal{N}_s^+| \, |\mathcal{N}_t^+|}} \boldsymbol{e}_s^{l-1} \tag{6}$$

其中，$w_{s,t}^+$是图中节点 s 和 t 之间的互补边的权重，$\mathcal{N}_s^+$和$\mathcal{N}_t^+$分别代表 s 和 t 的互补邻居节点集合，$\boldsymbol{e}_s^{l-1}$是互补邻居节点 s 在第 $l-1$ 层的特征嵌入。对节点 t，在第 l 层的互补特征嵌入是其所有互补邻居节点传播而来的信息的总和：

$$\boldsymbol{e}_t^{l+} = \beta \sum_{s \in \mathcal{N}_t^+} \boldsymbol{m}_{t \leftarrow s}^{l\,+} = \beta \sum_{s \in \mathcal{N}_t^+} \frac{w_{s,t}^+}{\sqrt{|\mathcal{N}_s^+| \, |\mathcal{N}_t^+|}} \boldsymbol{e}_s^{l-1} \tag{7}$$

其中，β 是超参数，用于控制互补关系在信息传播中对节点特征的影响。

互斥边蕴含的现实意义与互补边不同，故其权重的计算方式和信息传播的方法也与互补边不同。在第 l 层，对于某个特定的 TPL t，它接收的来自其互斥邻居的互斥信息为：

$$\boldsymbol{m}_{t \leftarrow s}^{l\,-} = -\frac{w_{s,t}^-}{(\sum_{i \in T} \log(P_i))^2} \boldsymbol{e}_s^{l-1} \tag{8}$$

其中，$(\sum_{i \in T} \log(P_i))^2$是所有 TPL 流行度对数和的平方，用于减小权重$w_{s,t}^-$的影响，防止部分流行度很高的 TPL 影响模型训练。鉴于互斥关系的特殊性，采用负权重的方式进行互斥信息的传播。对节点 t，在第 l 层的互斥特征嵌入是其所有互斥邻居节点传播而来的信息的总和：

$$\boldsymbol{e}_t^{l-} = \gamma \sum_{s \in \mathcal{N}_t^-} \boldsymbol{m}_{t \leftarrow s}^{l\,-} = \gamma \sum_{s \in \mathcal{N}_t^-} -\frac{w_{s,t}^-}{(\sum_{i \in T} \log(P_i))^2} \boldsymbol{e}_s^{l-1} \tag{9}$$

其中，γ 是超参数，用于控制互斥关系在信息传播中对节点特征的影响。

3.4 聚合与预测层

在传播层之后，可以得到 $n+1$ 个特征。对 App 来说，每一层的特征分别记为$\boldsymbol{E}_A^0, \boldsymbol{E}_A^1 \cdots \boldsymbol{E}_A^n$；对 TPL 来说，每一层的特征分别记为$\boldsymbol{E}_T^0, \boldsymbol{E}_T^1 \cdots \boldsymbol{E}_T^n$。每一层的特征代表 App(TPL)获取的不同层次的细粒度特征。MDGRec 分别将 App 和 TPL 每一层特征求平均，得到 App 和 TPL 的最终特征嵌入：

$$\boldsymbol{E}_A^* = \frac{1}{n+1} \sum_{i=0}^{n} \boldsymbol{E}_A^i, \boldsymbol{E}_T^* = \frac{1}{n+1} \sum_{i=0}^{n} \boldsymbol{E}_T^i \tag{10}$$

得到 App 和 TPL 的最终特征嵌入后，使用内积计算 App a 和 TPL t 之间的预测评分：

$$\hat{r}_{a,t} = \boldsymbol{e}_a^* \cdot \boldsymbol{e}_t^* \tag{11}$$

其中，$\cdot$ 代表内积操作。预测评分越高，代表 App a 调用 TPL t 的可能性越大。随后，MDGRec 将每个 App 得到的对所有 TPL 的预测评分进行排序，并选取预测值最大且未被 App 调用的 nr 个 TPL 作为推荐列表。依照先前的工作[10,12]，本文中 nr 取 5,10。

为了学习模型参数，BPR 损失[23]被用作 MDGRec 的损失函数。BPR 基于 TPL 之间的偏序关系构造损失，即假设 App 使用过的 TPL 预测评分应高于未使用过的 TPL。BPR 损失如下：

$$Loss = \sum_{(a,s,t) \in O} -\ln \sigma(\hat{r}_{a,s} - \hat{r}_{a,t}) + \lambda \| \Theta \|_2^2 \tag{12}$$

其中，$O = \{(a,s,t) | (a,s) \in R^+, (a,t) \in R^-\}$；$R^+$代表 App 和 TPL 之间交互的集合，$R^-$反之；$\sigma(\cdot)$是 sigmoid 函数；$\Theta$表示模型的参数，$\Theta = \{\boldsymbol{E}_A, \boldsymbol{E}_T\}$；$\lambda$ 是正则化参数。对于每个正例(a,s)，MDGRec 随机采样一个负例(a,t)使得$(a,s,t) \in O$，并将这个三元组作为输入来训练模型。

3.5 模型复杂度分析

空间复杂度：如 3.2 节所述，App 和 TPL 嵌入向量是本文所有的可训练参数。因此，MDGRec 的空间复杂度为 $O((M+N) \times d)$，其中 M 和 N 分别是 TPL 和 App 的数量，d 是特征嵌入的维度。对比主要的基线方法 Grec，其在图网络

的每一层中都引入了两个线性变换矩阵，因此 Grec 的空间复杂度为 $O((M+N)\times d+2ld^2)$，其中 l 是图神经网络层数，因此 MDGRec 在空间复杂度上优于 Grec。

时间复杂度：MDGRec 训练的时间复杂度主要由 3 部分构成。1)图信息传播。由式(5)、式(7)、式(9)可知，在每层中共现、互补和互斥信息传播的时间复杂度分别为 $O(2|E_{AC}|d)$，$O(2|E_{TC}|d)$，$O(2|E_{TI}|d)$，其中 $|E_{AC}|$，$|E_{TC}|$，$|E_{TI}|$ 分别是 App 共现关系、TPL 互补关系和 TPL 互斥关系的数量，系数 2 表明图中的信息传播是双向的。对于 l 层图网络，图信息传播的时间复杂度为 $O(2l\times(|E_{AC}|+|E_{TC}|+|E_{TI}|)\times d)$。2)特征聚合。特征聚合层是对图网络每层得到的 App 和 TPL 嵌入做平均。每一层得到的嵌入大小为 $(M+N)\times d$，因此聚合层的时间复杂度为 $O((l+1)\times(M+N)\times d)$。$l+1$ 代表最终嵌入是图网络每层得到的嵌入和初始嵌入的平均值。3)结果预测。在预测层中，通过内积计算 App 和 TPL 的最终评分。如式(12)所示，对于每个正例，都有一个相应的负例用于计算 BPR 损失。因此，预测层的时间复杂度为 $O(2|R^+|d)$，其中 $|R^+|$ 是 App 和 TPL 历史调用记录的数量。综上，MDGRec 训练的时间复杂度为 $O(2l\times(|E_{AC}|+|E_{TC}|+|E_{TI}|)\times d+(l+1)\times(M+N)\times d+2|R^+|d)$。

4 实验及分析

4.1 数据集

为了保证与先前工作的一致性以及实验比较的公平性，MaLib 被用作实验数据集之一。LibSeek[12]，Grec[10]，NLA-GNN[11]使用的都是该数据集。MaLib 从 AndroidZoo[1)]收集了 61 722 个安卓 App，包含 827 个 TPL 和 725 502 条 App-TPL 调用记录。依照 LibSeek[12]的做法，本文过滤了使用少于 10 个 TPL 的 App，最终得到的数据集包含 31 432 个 App、752 个 TPL 和 537 011 条 App-TPL 调用记录。

此外，从 APPBrain[2)]网站上收集安卓 App 数据作为另一数据集，称为 APPBrain。APPBrain 包含 22 977 个安卓 App，492 个 TPL 和 658 638 条 App-TPL 调用记录。同样地，使用与 MaLib 相同的数据处理方式过滤得到了包含 19 691 个 App、467 个 TPL 和 654 084 条 App-TPL 调用记录的数据集。

本文采用了与 LibSeek[12]相同的数据集划分方式。对于每个 App，随机选择 $rm(rm\in[1,3,5])$ 个被其使用过的 TPL 作为测试集，其余的作为训练集。简便起见，将上述 3 种不同划分方式的数据集分别命名为 MaLib＃1，MaLib＃3，MaLib＃5，APPBrain＃1，APPBrain＃3，APPBrain＃5。

4.2 实验设置

首先，对于训练集中的每条数据，随机采样一个负例用于计算 BPR 损失函数。接着，对于每个 App，模型会对除训练集外的所有 TPL 进行评分并排序，从中选择 $nr(nr\in[5,10])$ 个评分最高的 TPL 作为推荐列表。PyTorch 被用来实现本文模型，并在一台 Linux 服务器上进行实验。服务器的配置如下：Intel Xeon Gold 6342 CPU，256GB RAM 和 NVIDIA GeForce RTX 3090 GPU。使用 Adam[41]优化模型并使用 Xavier[42]初始化模型参数。为了公平比较，将特征嵌入的维度设为 128，这与之前的方法[10,12]是一致的。batch size 设置为 4 096。模型其余参数如下：学习率为 0.001，正则化参数为 0.001，App 图权重为 1，TPL 互补权重为 0.000 1，TPL 互斥权重为 5 000，GNN 层数为 1。

4.3 评价指标

本文使用以下几个被广泛用于 Top-N 推荐的评价指标，这些指标的值越高，说明其效果越好。

1)Mean Precision(MP)：给定一个 App，Precision 代表正确推荐的 TPL 占其推荐列表中 TPL 的比例。MP 是每个 App 的 Precision 的平均值，可以表示为：

$$MP=\frac{1}{|A|}\sum_{a\in A}\frac{|\mathcal{R}(a)\bigcap\mathcal{T}(a)|}{|\mathcal{R}(a)|} \tag{13}$$

其中，$\mathcal{R}(a)$代表 App 的推荐列表，$\mathcal{T}(a)$代表 App 对应的测试集。

2)Mean Recall(MR)：给定一个 App，Recall 代表正确推荐的 TPL 占测试集中 TPL 的比例。MR 是每个 App 的 Recall 的平均值，可以表示为：

$$MR=\frac{1}{|A|}\sum_{a\in A}\frac{|\mathcal{R}(a)\bigcap\mathcal{T}(a)|}{|\mathcal{T}(a)|} \tag{14}$$

3)Mean F1 Score(MF)：给定一个 App，F1 Score 是其 Precision 和 Recall 的调和平均数。MF 是每个 App 的 F1 Score 的平均值，可以表示为：

$$MF=\frac{1}{|A|}\sum_{a\in A}\frac{2\times Precision_a\times Recall_a}{Precision_a+Recall_a} \tag{15}$$

4)Mean Average Precision(MAP)：给定一个 App，Average Precision(AP)不仅考虑了正确推荐的 TPL 数量，还考虑它们排名的相对位置。MAP 是每个 App 的 AP 的平均值，可以表示为：

$$AP_a=\frac{1}{\sum_{i=1}^{nr}rel(i)}\sum_{i=1}^{nr}\frac{\sum_{j=1}^{i}rel(i)}{i}\times rel(i) \tag{16}$$

$$MAP=\frac{1}{|A|}\sum_{a\in A}AP_a \tag{17}$$

其中，$rel(i)$代表推荐列表的第 i 个位置是否命中。如果命中，那么 $rel(i)=1$；否则 $rel(i)=0$。

4.4 对比方法

为了说明 MDGRec 的有效性，将其与以下方法进行对比。

1)POP：该方法为 App 推荐其未使用过的最流行的 TPL。它对 App 和 TPL 没有任何特异性的建模，可以视为所有方法的基准方法。

2)BPR[23]：该方法是基于隐式反馈的一种矩阵分解算法。它假设 App 使用过的 TPL 要比未使用过的 TPL 更能满足 App 需求，并使用最大后验估计优化模型参数。

3)LibRec[8]：一种结合了关联规则挖掘和协同过滤技术进行 TPL 推荐的方法。

1) https://androzoo.uni.lu

2) https://www.appbrain.com/

4）LibSeek[12]：一种先进的 TPL 推荐方法。它运用了基于隐式反馈的矩阵分解的技术，并使用自适应权重和邻域信息解决 TPL 推荐中的长尾效应。

5）Grec[10]：一种先进的 TPL 推荐方法。它使用 GNN 在 App 与 TPL 节点之间传播信息，从而提取 App 和 TPL 之间的高阶交互信息。

6）NLA-GNN[11]：一种先进的 TPL 推荐方法。它使用简化的 GNN 网络加快了训练的速度，并使用一种交替信息提取方法缓解了 App 和 TPL 特征之间异质性的影响。NLA-GNN 有两种不同的变体，本文选择了效果更好的一种作为对比方法。

7）HGNRec[39]：一种先进的 TPL 推荐方法。它使用同构图分别提取 App 和 TPL 的高阶信息，将 App 和 TPL 映射到不同的知识空间中，以提升推荐的准确性。

4.5 性能评估

4.5.1 总体性能比较

MDGRec 模型与对比方法在两个数据集上的实验表现如表 1 所列，其中，最优结果加粗表示，次优结果用下划线表示。实验结果表明，MDGRec 的各项指标在 2 个数据集的 3 种划分方式上都表现出了最好的效果。在 $nr=5$ 的情况下，与次优模型（NLA-GNN）相比，MDGRec 在 APPBrain＃3 数据集上的 *MF* 提升了 5.07%，*MAP* 提升了 5.88%；在 MaLib＃3 数据集上的 *MF* 提升了 3.40%，*MAP* 提升了 2.49%。这证明了 MDGRec 在 TPL 推荐任务上的有效性。

表 1 总体性能对比

Table 1 Comparison of overall performance

数据集	方法	$nr=5$				$nr=10$			
		MP	*MR*	*MF*	*MAP*	*MP*	*MR*	*MF*	*MAP*
MaLib＃1	POP	0.0753	0.3765	0.1255	0.0316	0.0457	0.4565	0.0831	0.2949
	BPR	0.1326	0.6629	0.2210	0.5150	0.0754	0.7544	0.1371	0.5274
	LibRec	0.1267	0.6335	0.2112	0.4622	0.0668	0.6682	0.1215	0.4669
	LibSeek	0.1348	0.6741	0.2247	0.5236	0.0755	0.7553	0.1373	0.5346
	Grec	0.1521	0.7607	0.2536	0.6269	0.0828	0.8283	0.1506	0.6360
	HGNRec	0.1525	0.7630	0.2543	0.6357	0.0824	0.8239	0.1498	0.6440
	NLA-GNN	0.1544	0.7721	0.2574	0.6322	0.0835	0.8345	0.1517	0.6409
	MDGRec	**0.1581**	**0.7909**	**0.2636**	**0.6700**	**0.0848**	**0.8484**	**0.1543**	**0.6777**
MaLib＃3	POP	0.2147	0.3579	0.2684	0.5931	0.1341	0.4468	0.2063	0.5682
	BPR	0.3497	0.5904	0.4387	0.7209	0.2095	0.7068	0.3229	0.6829
	LibRec	0.2789	0.4648	0.3486	0.6883	0.1542	0.5142	0.2373	0.6864
	LibSeek	0.3710	0.6183	0.4637	0.7280	0.2158	0.7193	0.3320	0.6971
	Grec	0.4099	0.6915	0.5142	0.7977	0.2337	0.7879	0.3602	0.7605
	HGNRec	0.4189	0.6982	0.5236	0.8115	0.2359	0.7865	0.3630	0.7763
	NLA-GNN	0.4176	0.7046	0.5237	0.8080	0.2350	0.7930	0.3621	0.7742
	MDGRec	**0.4332**	**0.7220**	**0.5415**	**0.8281**	**0.2442**	**0.8139**	**0.3757**	**0.7899**
MaLib＃5	POP	0.3383	0.3383	0.3383	0.7413	0.2180	0.4360	0.2907	0.6813
	BPR	0.5052	0.5121	0.5083	0.7838	0.3238	0.6555	0.4333	0.7253
	LibRec	0.4400	0.4400	0.4400	0.6922	0.2434	0.4868	0.3245	0.6890
	LibSeek	0.5291	0.5291	0.5291	0.7896	0.3293	0.6587	0.4391	0.7396
	Grec	0.5868	0.5945	0.5902	0.8397	0.3613	0.7312	0.4834	0.7856
	HGNRec	0.6136	0.6136	0.6136	0.8650	0.3691	0.7383	0.4922	0.8131
	NLA-GNN	0.6073	0.6151	0.6108	0.8567	0.3683	0.7452	0.4927	0.8054
	MDGRec	**0.6301**	**0.6301**	**0.6301**	**0.8714**	**0.3819**	**0.7638**	**0.5092**	**0.8184**
APPBrain＃1	POP	0.0984	0.4918	0.1640	0.3490	0.0569	0.5692	0.1035	0.3593
	BPR	0.1330	0.6648	0.2216	0.5281	0.0753	0.7530	0.1369	0.5399
	LibRec	0.1171	0.5853	0.1951	0.4713	0.0694	0.6944	0.1263	0.4868
	LibSeek	0.1382	0.6910	0.2303	0.5255	0.0771	0.7713	0.1402	0.5362
	Grec	0.1433	0.7166	0.2389	0.5676	0.0700	0.7997	0.1454	0.5786
	HGNRec	0.1445	0.7226	0.2409	0.5781	0.0804	0.8036	0.1461	0.5890
	NLA-GNN	0.1454	0.7269	0.2423	0.5865	0.0802	0.8019	0.1458	0.5966
	MDGRec	**0.1507**	**0.7533**	**0.2511**	**0.6277**	**0.0826**	**0.8260**	**0.1502**	**0.6374**
APPBrain＃3	POP	0.2770	0.4617	0.3463	0.6177	0.1658	0.5526	0.2551	0.5937
	BPR	0.3672	0.6121	0.4590	0.7810	0.2169	0.7230	0.3337	0.7350
	LibRec	0.3174	0.5291	0.3968	0.7658	0.1980	0.6600	0.3046	0.7211
	LibSeek	0.3796	0.6326	0.4745	0.7599	0.2221	0.7404	0.3417	0.7201
	Grec	0.3887	0.6477	0.4858	0.7919	0.2280	0.7599	0.3507	0.7451
	HGNRec	0.3993	0.6656	0.4992	0.8137	0.2296	0.7652	0.3532	0.7695
	NLA-GNN	0.4008	0.6679	0.5010	0.8056	0.2298	0.7662	0.3536	0.7638
	MDGRec	**0.4212**	**0.7019**	**0.5264**	**0.8530**	**0.2396**	**0.7985**	**0.3685**	**0.8067**
APPBrain＃5	POP	0.4256	0.4256	0.4256	0.7359	0.2699	0.5398	0.3599	0.6919
	BPR	0.5562	0.5562	0.5562	0.8609	0.3458	0.6915	0.4610	0.7988
	LibRec	0.4972	0.4972	0.4972	0.8495	0.3194	0.6389	0.4259	0.7945
	LibSeek	0.5647	0.5647	0.5647	0.8276	0.3536	0.7071	0.4714	0.7741
	Grec	0.5719	0.5719	0.5719	0.8536	0.3577	0.7154	0.4769	0.7937
	HGNRec	0.5907	0.5907	0.5907	0.8670	0.3656	0.7312	0.4875	0.8083
	NLA-GNN	0.5936	0.5936	0.5936	0.8660	0.3626	0.7253	0.4835	0.8112
	MDGRec	**0.6307**	**0.6307**	**0.6307**	**0.8998**	**0.3814**	**0.7627**	**0.5085**	**0.8427**

实验结果还表明，MDGRec 和基线方法在所有指标上都优于 POP，这主要是因为这些方法都对 App 和 TPL 进行了个性化建模。基于图的 Grec，NLA-GNN，HGNRec 和 MDGRec 显著优于其他方法，凸显了 GNN 挖掘 App 和 TPL 之间潜在关系的强大能力。值得注意的是，相比于其他方法，MDGRec 在 $rm=3$ 和 $rm=5$ 时取得了比 $rm=1$ 更大的提升，表明 MDGRec 在处理稀疏数据时具有较强的适应能力。

我们推测这种性能提升主要源于 3 个方面：1）考虑到 TPL 推荐任务的特性，MDGRec 引入了多种关系，从而更准确地刻画了 App 和 TPL 的特征；2）使用双图结构区分 App 与 TPL 节点，减少了不同类型节点之间特征混淆和数据不平衡对模型的影响；3）对图中的边赋予了自适应权重，从而有效区分了不同节点在不同关系维度上的影响强弱，有助于挖掘节点之间的联系。为了验证这些猜想，围绕上述几个方面展开了实验，实验结果与分析将在下文中详细呈现。由于在 APPBrain 和 MaLib 数据集上得到的实验结果相似，后续章节将仅展示基于 MaLib 数据集的实验结果。

4.5.2 α,β,γ 的影响

超参数 α,β,γ 用于调节文中 3 种关系对模型影响的强度。本文进行了充足的实验，来探究这 3 个超参数对模型性能的影响。对于每个超参数，固定其他超参数，并在一定范围内采样。具体而言，α 的采样范围为[0.01，0.1，1，10]；β 的采样范围为[0.000 01，0.00 01，0.001，0.01]；γ 的采样范围为[50，500，5 000，50 000]。图 2 展示了在 $nr=5$ 的情况下，MDGRec 在 MaLib 数据集上的 MF 的表现，其余指标的实验结果与 MF 类似。可以发现，在 3 个数据集上，MF 分别在 α，β,γ 为 1，0.000 1，5 000 时达到最大值，随后性能开始显著下降。因此，本文选择 $\alpha=1,\beta=0.000\,1,\gamma=5\,000$ 作为 MDGRec 的超参数。值得注意的是，α,β,γ 的取值会受到数据集统计量的影响，因为数据集的大小以及边的数量会影响图拉普拉斯正则项的大小。通过调整 α,β,γ 的大小，MDGRec 可以轻松地扩展到其他数据集或推荐场景。

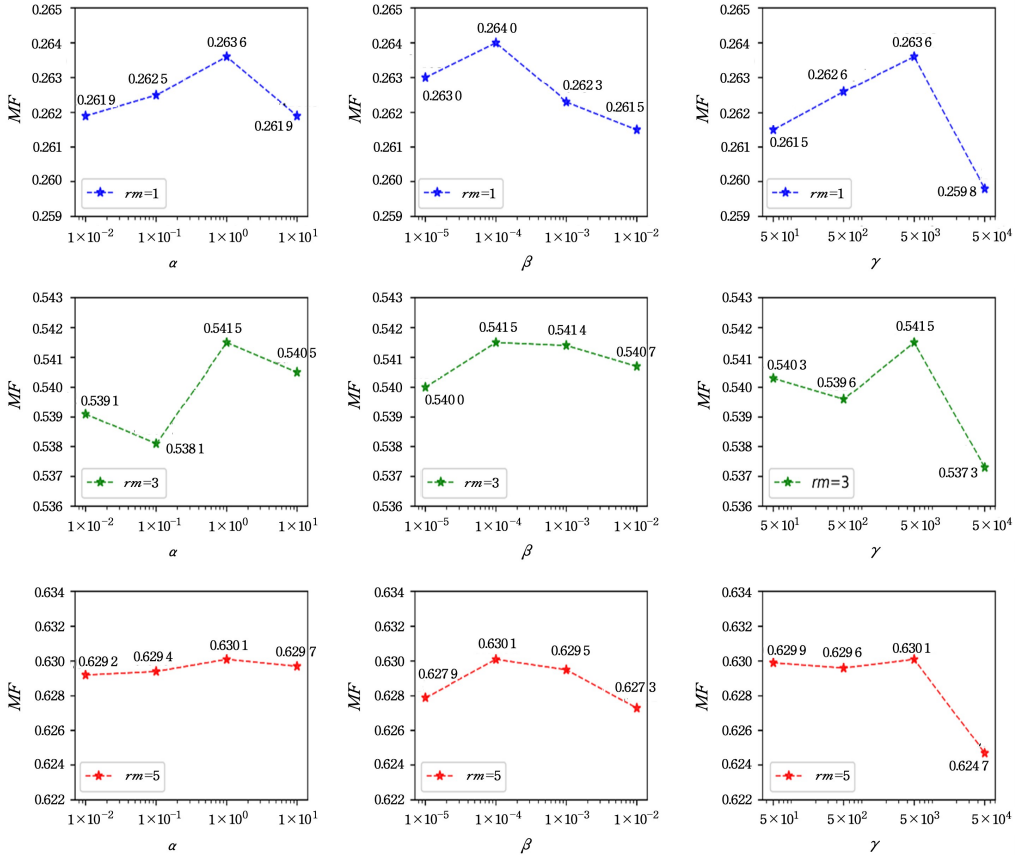

图 2 超参数 α,β,γ 对模型的影响

Fig. 2 Impact of hyperparameter α,β,γ

4.5.3 特征嵌入维度的影响

为了探索特征嵌入维度对模型性能的影响，固定了模型的其他参数，将特征嵌入维度 d 在[64，128，256，512]内调整。图 3 展示了 MaLib 数据集在 $nr=5$ 和 $nr=10$ 情况下模型在 MF 指标上的表现，其余指标的结果与 MF 相似。当 $nr=5$ 时，d 从 64 提升到 128 时，MDGRec 的性能明显提升；然而随着 d 继续增大，模型性能开始出现缓慢的下降。当 $nr=10$，$rm=3$ 时，MF 随 d 的变化与 $nr=5$ 时类似。不同的是，在 $rm=1$ 和 $rm=5$ 时，MF 随 d 的增大而增加。这说明在 $d=256$ 和 $d=512$ 时，模型推荐排名前 5 的 TPL 的性能有所下降，而推荐排名在 6－10 位 TPL 的性能有所提升。可能的原因是，当 d 增大时，特征嵌入包含的 App 和 TPL 信息更加丰富。对于排名前 5 TPL，其本身的特征已经足够丰富，更多的信息可能意味着更多的噪声。然而，排位更靠后如排名 6－10 的 TPL 的信息可能不足以进行准确匹配，而更多的信息能够帮助模型发现 App 和 TPL 之间的潜在关联。

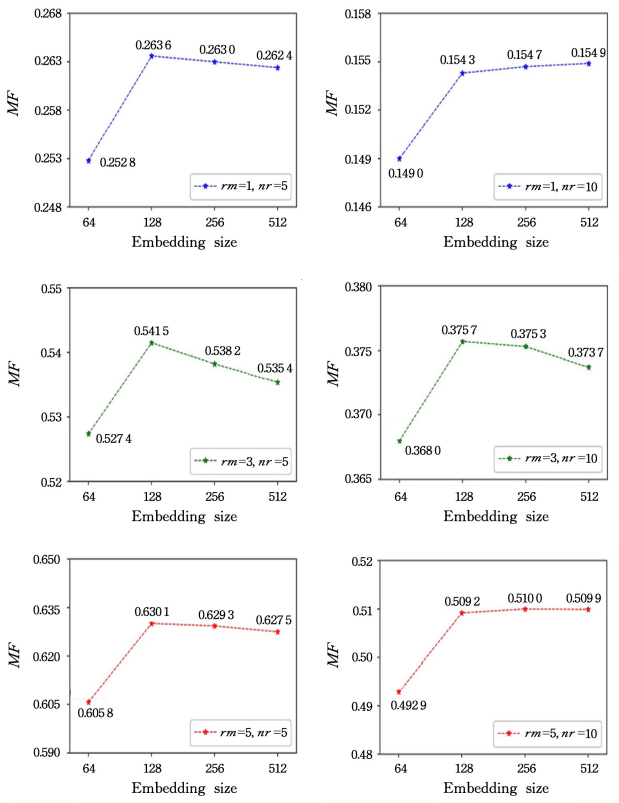

图 3 嵌入维度对模型的影响

Fig. 3 Impact of the embedding size

4.5.4 GNN 层数的影响

GNN 层数在模型节点信息提取深度方面发挥着关键作用，是影响模型性能的重要因素。为了研究 GNN 的层数对 MDGRec 性能的影响，固定模型的其他参数，将 GNN 层数 n 在 1，2，3 中调整，分别在 3 个数据集上进行实验。图 4 展示了 MaLib 数据集在 $nr=5$ 的情况下，不同 GNN 层数时 MDGRec 在 MF 上的表现。其余指标的实验结果与 MF 类似。

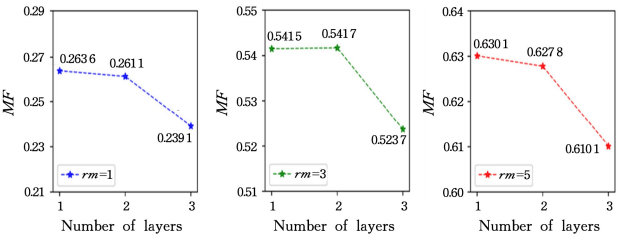

图 4 GNN 层数对模型的影响

Fig. 4 Impact of the GNN layer number

图 4 中的结果显示，在 $rm=1$ 和 $rm=5$ 时，模型的性能随着 n 的增大而下降，在 $n=1$ 时取得最好的性能。在 $rm=3$ 的情况下，相比 $n=1$，$n=2$ 时性能略有提升，但基本持平；而在 $n=3$ 时模型性能显著下降。因此，将 GNN 的层数置为 1。

值得注意的是，在先前的工作中[10-11]，模型在 $n=3$ 时取得最佳性能，而 MDGRec 在 $n=1$ 时就达到了最佳效果。这是因为 MDGRec 使用的 App 共现关系与 TPL 互补互斥关系是依据 App-TPL 交互关系构建的，可以视为在图构建过程中预先进行了一次信息传播，隐含了 App 与 TPL 的高阶关系。因此，MDGRec 在 $n=1$ 时提取的高阶信息与先前工作在 $n=2$ 时基本一致。由于 GNN 的过平滑问题[43]，当 n 增大之后，MDGRec 的性能呈现下降趋势。考虑到只需一层 GNN 即可达到最佳性能，MDGRec 的训练速度要快于先前的工作。在 MaLib#1，MaLib#3 和 MaLib#5 这 3 个数据集上，Grec 平均每轮的训练时间为 22.9 s，22.9 s，22.2 s，而 MDGRec 仅需 12.8 s，12.7 s，12.7 s。

4.5.5 消融实验

1）3 种关系的影响

为了探究 3 种关系对模型性能的影响，设计了保留 MDGRec 模型中的一种关系或两种关系的变体，记为 MDGRec-x，x 代表模型中保留的关系类型。例如，MDGRec-$\alpha\beta$ 代表模型中仅保留 App 共现关系与 TPL 互补关系。实验结果如表 2 所列。结果表明，MDGRec 的性能优于所有变体，保留两种关系变体的性能优于保留一种关系变体的性能。这表明融合的 3 种关系均能提升模型对节点的表征能力，从而提高推荐的准确率。

表 2 3 种关系的消融实验结果

Table 2 Ablation results for three types of relations

方法	MF			排名
	MaLib#1	MaLib#3	MaLib#5	
MDGRec	0.2636	0.5415	0.6301	1
MDGRec-α	0.2589	0.5365	0.6232	6
MDGRec-β	0.2570	0.5340	0.6231	7
MDGRec-γ	0.2604	0.5379	0.6260	5
MDGRec-$\beta\gamma$	0.2624	0.5388	0.6276	3
MDGRec-$\alpha\gamma$	0.2620	0.5404	0.6281	2
MDGRec-$\alpha\beta$	0.2615	0.5373	0.6276	4

此外，通过变体的性能排名可以发现，在单关系变体中，MDGRec-γ 性能最优，MDGRec-α 次之，MDGRec-β 性能最差。这表明引入的 3 种关系中，互斥关系对 MDGRec 性能的提升最大，其次是共现关系，互补关系的提升最小。从双关系变体的实验结果中也可以得出类似的结论。可能的原因在于，融入互斥关系使得模型能够有效识别高流行度且容易被错误推荐的 TPL。若不显式建模互斥关系，高流行度 TPL 往往由于其广泛的交互记录而被模型优先推荐，但这些 TPL 可能与当前 App 中已有的 TPL 互斥，导致错误推荐，影响模型性能。虽然这些 TPL 在总数上占比较少，但由于其流行度高，它们对模型性能的影响非常显著。然而，由于互斥关系的复杂性和独特性，模型仅依赖历史交互数据难以有效识别这种关系。相比之下，共现和互补关系更容易被模型通过学习 App 和 TPL 的历史交互数据隐式捕捉，而显式建模这两种关系仅在一定程度上强化了模型对其的利用效果。因此，尽管显式地引入共现关系和互补关系能在一定程度上提升模型的性能，但引入互斥关系对模型性能的提升更为显著。

2）边权重的影响

为了探究 MDGRec 中自适应边权重的有效性，本文设计了一个变体 MDGRec-w。MDGRec-w 忽略了所有关系的强弱，将 App 图和 TPL 图中所有的边权重置为 1。

实验结果如表 3 所列，MDGRec-w 的效果略逊于 MDGRec，说明自适应边权重有助于提升模型性能。这是因为，通过赋予自适应边权重，模型可以区分不同节点之间关系

的强弱，以更细粒度地刻画节点特征。同时，即使没有使用边权重，仅使用 3 种关系和双图结构，MDGRec-w 的效果仍优于基线中的次优模型 NLA-GNN 和 HGNRec，说明本文提出的双图结构和多关系建模更加符合 TPL 推荐场景的特性。

表 3 自适应边权重的消融实验结果

Table 3 Ablation results for adaptive edge weights

数据集	模型	$nr=5$				$nr=10$			
		MP	*MR*	*MF*	*MAP*	*MP*	*MR*	*MF*	*MAP*
MaLib#1	MDGRec	**0.1581**	**0.7909**	**0.2636**	**0.6700**	**0.0848**	**0.8484**	**0.1543**	**0.6777**
	MDGRec-w	0.1560	0.7799	0.2600	0.6487	0.0840	0.8398	0.1527	0.6568
	HGNRec	0.1525	0.7630	0.2543	0.6357	0.0824	0.8239	0.1498	0.6440
	NLA-GNN	0.1544	0.7721	0.2574	0.6322	0.0835	0.8345	0.1517	0.6409
MaLib#3	MDGRec	**0.4332**	**0.7220**	**0.5415**	**0.8281**	**0.2442**	**0.8139**	**0.3757**	**0.7899**
	MDGRec-w	0.4300	0.7167	0.5375	0.8203	0.2432	0.8106	0.3741	0.7825
	HGNRec	0.4189	0.6982	0.5236	0.8115	0.2359	0.7865	0.3630	0.7763
	NLA-GNN	0.4176	0.7046	0.5237	0.8080	0.2350	0.7930	0.3621	0.7742
MaLib#5	MDGRec	**0.6301**	**0.6301**	**0.6301**	**0.8714**	**0.3819**	**0.7638**	**0.5092**	**0.8184**
	MDGRec-w	0.6259	0.6259	0.6259	0.8688	0.3802	0.7605	0.5070	0.8155
	HGNRec	0.6136	0.6136	0.6136	0.8650	0.3691	0.7383	0.4922	0.8131
	NLA-GNN	0.6073	0.6151	0.6108	0.8567	0.3683	0.7452	0.4927	0.8054

结束语 本文提出了一种基于多元关系融合的移动应用第三方库推荐方法 MDGRec，通过双图神经网络分别对 App 和 TPL 进行建模，以解决 TPL 推荐场景中特征混淆和数据不平衡的问题。MDGRec 将 TPL 推荐场景中的 3 种不同的关系作为边，关系的强弱作为边的权重，从而细粒度地在 App 和 TPL 之间传播信息，提高模型对节点表征的学习能力。在 MaLib 和 AppBrain 数据集上的实验结果表明，MDGRec 优于所有基线方法，同时消融实验验证了模块的有效性。

但是该模型也具有一定局限性。MDGRec 训练依赖 App 与 TPL 之间的历史调用数据，对冷启动场景的处理能力较弱。未来将考虑融合多模态的辅助信息，增强 App 的功能表示，提升模型对冷启动场景的适应能力。

参 考 文 献

[1] LI M H, WANG W, WANG P, et al. Libd: Scalable and precise third-party library detection in android markets[C]// 2017 IEEE/ACM 39th International Conference on Software Engineering(ICSE). New York: IEEE, 2017: 335-346.

[2] BACKES M, BUGIEL S, DERR E. Reliable third-party library detection in android and its security applications[C]// Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM, 2016: 356-367.

[3] ZHAN X, LIU T M, FAN L L, et al. Research on third-party libraries in android apps: A taxonomy and systematic literature review[J]. IEEE Transactions on Software Engineering, 2021, 48(10): 4181-4213.

[4] ZHANG Y H, WANG J C, HUANG H X, et al. Understanding and conquering the difficulties in identifying third-party libraries from millions of android apps[J]. IEEE Transactions on Big Data, 2021, 8(6): 1511-1523.

[5] NGUYEN P T, DI ROCCO J, DI RUSCIO D, et al. CrossRec: Supporting software developers by recommending third-party libraries[J]. Journal of Systems and Software, 2020, 161: 110460.

[6] SALZA P, PALOMBA F, DI NUCCI D, et al. Third-party libraries in mobile apps: When, how, and why developers update them[J]. Empirical Software Engineering, 2020, 25: 2341-2377.

[7] HENRIQUES H, LOURENÇO H, AMARAL V, et al. Improving the developer experience with a low-code process modelling language[C]// Proceedings of the 21th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems. New York: ACM, 2018: 200-210.

[8] THUNG F, LO D, LAWALL J. Automated library recommendation[C]// 2013 20th Working Conference on Reverse Engineering(WCRE). New York: IEEE, 2013: 182-191.

[9] ZHAO X Q, LI S P, YU H, et al. Accurate library recommendation using combining collaborative filtering and topic model for mobile development[J]. IEICE Transactions on Information and Systems, 2019, 102(3): 522-536.

[10] LI B, HE Q, CHEN F F, et al. Embedding app-library graph for neural third party library recommendation[C]// Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. New York: ACM, 2021: 466-477.

[11] JIN Y, ZHANG Y, ZHANG Y W. Neighbor Library-Aware Graph Neural Network for Third Party Library Recommendation[J]. Tsinghua Science and Technology, 2023, 28(4): 769-785.

[12] HE Q, LI B, CHEN F F, et al. Diversified third-party library prediction for mobile app development[J]. IEEE Transactions on Software Engineering, 2020, 48(1): 150-165.

[13] YU H, XIA X, ZHAO X Q, et al. Combining collaborative filtering and topic modeling for more accurate android mobile app library recommendation[C]// Proceedings of the 9th Asia-Pacific Symposium on Internetware. New York: ACM, 2017: 1-6.

[14] OUNI A, KULA R G, KESSENTINI M, et al. Search-based software library recommendation using multi-objective optimization[J]. Information and Software Technology, 2017, 83: 55-75.

[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. New York: ACM, 2019: 165-174.

[16] ZOU C M, FAN Z F. GELIBREC: Third-Party Libraries Recommendation Using Graph Neural Network[C]// International Conference on Database Systems for Advanced Applications. Cham: Springer, 2022: 332-340.

[17] SU J, ZHAO T, WU J, et al. Graph Convolution Recommendation Algorithm Integrating Multi-relationship Preferences[C]// International Conference on Intelligent Computing. Singapore:

Springer，2024：167-177.
[18] CHEN H，HE J，XU W，et al. Enhanced multi-relationships integration graph convolutional network for inferring substitutable and complementary items[C]// Proceedings of the AAAI Conference on Artificial Intelligence. Palo Alto，CA：AAAI，2023：4157-4165.
[19] SCHAFER J B，FRANKOWSKI D，HERLOCKER J，et al. Collaborative filtering recommender systems[M]// The Adaptive Web：Methods and Strategies of Web Personalization. Berlin：Springer，2007：291-324.
[20] HE X N，LIAO L Z，ZHANG H W，et al. Neural collaborative filtering[C]// Proceedings of the 26th International Conference on World Wide Web. New York：ACM，2017：173-182.
[21] REN Q，LI B，WANG J，et al. Hybrid Recommendation Method of Third-party Library for Mobile Application Development[J]. Journal of Chinese Mini-Micro Computer Systems，2019，40(9)：1809-1814.
[22] KOREN Y，BELL R，VOLINSKY C. Matrix factorization techniques for recommender systems[J]. Computer，2009，42(8)：30-37.
[23] RENDLE S，FREUDENTHALER C，GANTNER Z，et al. BPR：Bayesian personalized ranking from implicit feedback[J]. arXiv：1205.2618，2012.
[24] FU S H，LIU W F，ZHANG K，et al. Semi-supervised classification by graph p-Laplacian convolutional networks[J]. Information Sciences，2021，560：92-106.
[25] WU F，SOUZA A，ZHANG T Y，et al. Simplifying graph convolutional networks[C]// International Conference on Machine Learning. New York：PMLR，2019：6861-6871.
[26] MAO K L，ZHU J M，XIAO X，et al. UltraGCN：ultra simplification of graph convolutional networks for recommendation [C]// Proceedings of the 30th ACM International Conference on Information & Knowledge Management. New York：ACM，2021：1253-1262.
[27] FAN W Q，MA Y，LI Q，et al. Graph neural networks for social recommendation[C]// The World Wide Web Conference. New York：ACM，2019：417-426.
[28] HE X N，DENG K，WANG X，et al. Lightgcn：Simplifying and powering graph convolution network for recommendation[C]// Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. New York：ACM，2020：639-648.
[29] ZHAO J，ZHANG X，GAO C，et al. KG2Lib：knowledge-graph-based convolutional network for third-party library recommendation[J]. The Journal of Supercomputing，2023，79(1)：1-26.
[30] LI B，QUAN H，WANG J，et al. Neural Library Recommendation by Embedding Project-Library Knowledge Graph[J]. IEEE Transactions on Software Engineering，2024，50(6)：1620-1638.
[31] ZHOU L，CHEN W Y，ZENG D Y，et al. DPGNN：Dual-perception graph neural network for representation learning [J]. Knowledge-Based Systems，2023，268：110377.
[32] LIU M，GAO H Y，JI S W. Towards deeper graph neural networks[C]// Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. New York：ACM，2020：338-348.
[33] LI X，FU C F，ZHAO Z Y，et al. Dual-Channel Multiplex Graph Neural Networks for Recommendation[J]. arXiv：2403.11624，2024.
[34] ZHANG R Y，MA H F，LI Q F，et al. Dual-view self-supervised co-training for knowledge graph recommendation[C]// International Conference on Database Systems for Advanced Applications. Cham：Springer，2023：113-128.
[35] ZHUANG C Y，MA Q. Dual graph convolutional networks for graph-based semi-supervised classification[C]// Proceedings of the 2018 World Wide Web Conference. New York：ACM，2018：499-508.
[36] ZHANG Y，ZHANG Y W，ZHAO Y C，et al. Dual Variational Graph Reconstruction Learning for Social Recommendation[J]. IEEE Transactions on Knowledge and Data Engineering，2024，36(11)：6002-6015.
[37] LUO H，MENG X，WANG S，et al. Spectral-Based Graph Neural Networks for Complementary Item Recommendation[C]// Proceedings of the AAAI Conference on Artificial Intelligence. AAAI，2024：8868-8876.
[38] WU B，ZHONG L H，LI H，et al. Efficient complementary graph convolutional network without negative sampling for item recommendation [J]. Knowledge-Based Systems，2022，256：109758.
[39] LI D T C，GAO Y X，WANG Z H，et al. Homogeneous graph neural networks for third-party library recommendation[J]. Information Processing & Management，2024，61(6)：103831.
[40] NGUYEN P T，RUBEI R，DI ROCCO J，et al. Dealing with Popularity Bias in Recommender Systems for Third-party Libraries：How far Are We? [C]// 2023 IEEE/ACM 20th International Conference on Mining Software Repositories (MSR). New York：IEEE，2023：12-24.
[41] KINGMA D P. Adam：A method for stochastic optimization[J]. arXiv：1412.6980，2014.
[42] GLOROT X，BENGIO Y. Understanding the difficulty of training deep feedforward neural networks[C]// Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. New York：JMLR，2010：249-256.
[43] RONG Y，HUANG W B，XU T Y，et al. Dropedge：Towards deep graph convolutional networks on node classification[J]. arXiv：1907.10903，2019.

CHEN Yuhan，born in 1999，postgraduate. His main research interests include recommender systems，software engineering and service computing.

LI Bing，born in 1969，Ph.D，professor，Ph.D supervisor，is a distinguished member of CCF (No. 06539D). His main research interests include software engineering，service computing and artificial intelligence.

（责任编辑：李亚辉）