基于 k 核分解的网络嵌入

张和平[1],张和贵[2],谢晓尧[1],张太华[3],张思聪[1],喻国军[1*]

(1.贵州师范大学贵州省信息与计算科学重点实验室,贵州 贵阳 550001;
2.西南财经大学工商管理学院,四川 成都 611130;3.贵州师范大学机械与电气工程学院,贵州 贵阳 550025)

摘 要: 近年来,网络嵌入技术受到了广大研究者的关注。不过大多数网络嵌入算法并未考虑到处于相同层级结构的节点间的结构相似性,这些节点在网络中通常具有相同的重要性。因此,提出一种基于网络层级结构的网络嵌入算法,称为 KCNE。KCNE 算法使用网络节点间的层级结构信息来保持节点之间的结构相似性。该算法首先基于 k 核(k-core)分解方法将网络中的节点划分为不同的层级,并且使用定制的随机游走方法为每个节点生成游走序列,该序列可以有效捕获节点的一阶邻域及处于同层级中的高阶相似节点,随后将游走序列输入到 Skip-gram 模型中,使学习到的节点表示具有更好的区分性。基于多个真实数据集的实验结果表明,在链路预测和节点分类任务上,KCNE 算法相比于 8 个基准算法中的次优算法性能提升最高分别约 4%和 5%。参数敏感性分析实验也表明了 KCNE 算法具有较好的鲁棒性。此外,该算法在运行效率方面均优于 Role2Vec、RARE 和 GEMSEC 算法。

关键词: 网络嵌入;结构相似性;随机游走;链路预测;节点分类

中图分类号: TP391 **文献标志码:** A **DOI:** 10.19678/j.issn.1000-3428.0069028

Network Embedding Based on k-core Decomposition

ZHANG Heping[1], ZHANG Hegui[2], XIE Xiaoyao[1], ZHANG Taihua[3], ZHANG Sicong[1], YU Guojun[1*]

(1. Key Laboratory of Information and Computing Science of Guizhou Province, Guizhou Normal University, Guiyang 550001, Guizhou, China;
2. School of Business Administration, Southwestern University of Finance and Economics, Chengdu 611130, Sichuan, China;
3. School of Mechanical and Electrical Engineering, Guizhou Normal University, Guiyang 550025, Guizhou, China)

【Abstract】 In recent years, network embedding technology has attracted considerable attention from researchers. However, most network embedding algorithms have not adequately addressed the structural similarity among nodes within the same hierarchical level, even though these nodes typically share similar importance within the network. Therefore, this paper proposes a network embedding algorithm based on the hierarchical structure of a network, called KCNE. The KCNE algorithm utilizes hierarchical structural information among network nodes to preserve the structural similarity between nodes. Specifically, the algorithm initially employs the k-core decomposition method to categorize the nodes in the network into different levels. Subsequently, a customized random walk method is employed to generate a random walk sequence for each node. This sequence effectively captures the first-order neighborhood of nodes and high-order similar nodes within the same level. The generated random walk sequences are then input into a Skip-gram model to ensure that the learned node representations possess enhanced discriminative capabilities. Finally, experimental results on multiple real datasets demonstrate that, in link prediction and node classification tasks, the KCNE algorithm outperforms the second-best algorithm among eight benchmark algorithms by approximately 4% and 5%, respectively. Sensitivity analysis experiments further confirm the superior robustness of the KCNE algorithm. Additionally, the algorithm exhibits superior efficiency compared to the Role2Vec, RARE, and GEMSEC algorithms.

【Key words】 network embedding; structural similarity; random walk; link prediction; node classification

0 引言

通常,复杂网络用于建模社交系统、生物系统、信息传播[1]等领域的实体及其之间的关系。网络嵌入可以将网络中的节点和边映射为低维连续的向量,同时保留重要的结构和语义属性[2]。在学习到节点表示后,可将其应用于下游的网络分析任务,如节点分类[3]、链路预测[4]和网络重构[5]等。

收稿日期: 2023-12-14 **修回日期:** 2024-01-28

基金项目: 国家自然科学基金(72061006);贵州省科技计划项目(黔科合支撑[2023]一般 449);贵州师范大学学术新苗基金(黔师新苗[2021]A30 号)。

通信作者 E-mail: * 254579993@qq.com

近年来,已经有研究者提出了多种网络嵌入方法来学习节点表示。受自然语言处理中无监督表示学习的启发,DeepWalk 方法[6]首先在网络上执行随机游走,然后使用 Skip-gram 自然语言模型[7]学习节点的表示;类似地,node2vec 方法[8]则对 DeepWalk 方法进行了扩展,通过在随机游走时灵活地平衡广度优先搜索(BFS)和深度优先搜索(DFS)策略,从而获取到不同程度的上下文信息;而 LINE 方法[9]则通过最大化节点对之间的一阶和二阶接近度来学习节点表示。

为了在随机游走的过程中获取到更丰富的上下文信息,CARE 算法[10]使用社区检测算法将网络划分为多个社区,并在游走过程中通过超参数来控制对一阶邻居和社区节点的采样;类似地,NRL_RWCE 算法[11]在随机游走的过程中也将社区信息统一到节点表示学习中;而 Role2Vec 算法[12]通过属性随机游走方法来学习基于角色的表示;进一步,RARE 算法[13]则通过角色意识随机游走方法来捕获节点的社区信息和结构角色信息。虽然以上的算法在一定程度上能捕捉到更高阶的信息,但是,基于社区检测和角色发现的网络嵌入缺乏对具有相同层次节点的考虑。这类节点通常在网络中具有相同的位置和重要性。

与以上方法不同,本文通过 k 核(k-core)分解,将网络中的节点划分为不同层级,通过 k 核值为每个节点分配一个网络层级编号,同时设置两个参数来控制随机游走在不同层面的游走概率,由此实现对节点一阶邻居和同层级节点间的采样,从而使学习到的节点表示能有效保留网络的局部和层级结构信息。

总体来说,本文的贡献主要有以下几点:

1)本文提出了一种简单有效的网络嵌入方法 KCNE,其学习到的节点表示能够有效保留节点的近邻性和层级结构相似性。

2)KCNE 算法独立地为每个节点生成多条不同的随机游走序列。通过设置两个超参数,随机游走可以灵活地捕获节点的一阶邻居和同级成员,为每个节点生成丰富的上下文节点信息,从而将不同的结构信息嵌入到最终的节点表示中。

3)在两个监督学习任务——链路预测与节点分类上的实验结果表明,在多个真实数据集上,本文提出的 KCNE 算法均优于基准方法,并且在运行效率和鲁棒性方面也优于大部分基准方法。

1 相关工作

常见的基于随机游走的网络嵌入方法有 HAW[14]、MARU[15]、wGCN[16]、DiaRW[17]、FHNE[18]等算法。HAW 算法提出了一种异质匿名游走方式来区分网络中的异质结构,并通过预测节点邻域中发生的游走来学习节点嵌入;MARU 算法提出了双向扩展随机游走的方法来增强学习节点的表示;wGCN 算法利用随机游走获取到特定节点的高阶局部结构,并使用这些高阶局部信息来增强节点的表示;DiaRW 算法在采样的过程中,通过基于度数偏向回溯以及可变长度游走的方式来获取到节点序列,以此来学习节点表示;FHNE 算法通过对节点的结构和邻域信息进行编码,并设计了一种有偏随机游走方式获取到节点序列。

另外一种是将社区信息或角色信息统一到节点表示学习中的网络嵌入算法[19]。DMC 算法[20]通过最小化损失函数来同时学习节点和社区的嵌入,该损失函数捕捉了社区之间的连接数,使得学习到的节点表示保留了社区信息;CNRL 算法[21]则通过主题模型将社区检测与网络表示学习整合到统一的框架中,并通过社区增强机制在学习到的节点表示中保留了社区信息;类似地,MNMF[22]、GEMSEC[23]、PCNE[24]、CADNE[25]等算法也将社区信息统一到节点表示学习中;而 RiWalk[26]则通过角色识别方法将结构角色信息整合到节点表示学习中;类似地,RMNE 算法[27]面向多重网络,将角色信息整合到节点表示中;RED 算法[28]综合考虑了全局角色和局部角色表征的问题;RENCC[29]则是一种基于聚类系数的新型角色嵌入方法;HRFE 算法[30]通过在异构网络中提取节点特征并通过非负矩阵分解进行角色发现,从而得到基于角色的节点表示。

2 本文方法

在本节中,将介绍 KCNE 算法的基本组成部分及其相关技术。

2.1 问题描述

根据惯例,本文将一个无向网络定义为 $G(V,E)$,其中,V 代表节点集合,E 代表边集,$(v_i,v_j)\in E$ 表示节点 v_i 和节点 v_i 之间存在连边。

网络嵌入旨在学习到图 G 中每个节点的低维稠密向量表示[31],即 $v_i\in\mathbb{R}^{n\times d}$($d\ll n$),其中,$n=|V|$ 代表网络节点的个数。本文主要是基于网络的层级结构信息并结合定制的随机游走方式来学习节点嵌入,从而使得学习到的节点表示在下游的网络分析任务(如链路预测和节点分类)中具有较好的性能。

2.2 KCNE 算法

本文提出的 KCNE 算法如算法 1 所示。总体上可以将其划分为 3 个部分：

1)网络节点层级划分。该部分使用 k 核分解方法将整个网络划分为多个不相交的集合，且每个节点都具有唯一的编号，即为该节点的 k 核值。

2)层级结构随机游走。通过定制的随机游走方式对节点的一阶邻居和具有相同层级结构编号的节点进行有偏采样。参数 r 控制一阶邻居的采样概率，而参数 t 则控制同层级成员间的采样。

3)学习节点嵌入。基于层级结构随机游走获得的节点序列，使用 Skip-gram 模型来学习节点表示。

算法 1 KCNE 算法

输入 图 $G(V, E)$，嵌入维度 d，参数 r，参数 t，随机游走长度 L，游走次数 K，窗口大小 W

输出 节点嵌入矩阵 $f \in \mathbb{R}^{|V|\times d}$

1. k_list=Level_Assignment(G)//网络节点层级划分
2. 初始化 walks 为 null
3. while (i<K) do:
4. S=shuffle(V)
5. for v in V do:
6. walk=Hierarchical_structure-based_random_walk(G, k_list, L, v, r, t)//层级结构随机游走
7. walks.append(walk)
8. end for
9. i+=1
10. end while
11. f=SkipGram(W, d, walks)//学习节点表示
12. return f

KCNE 算法的总体流程如图 1 所示，其中不同形状的节点属于不同的层级。

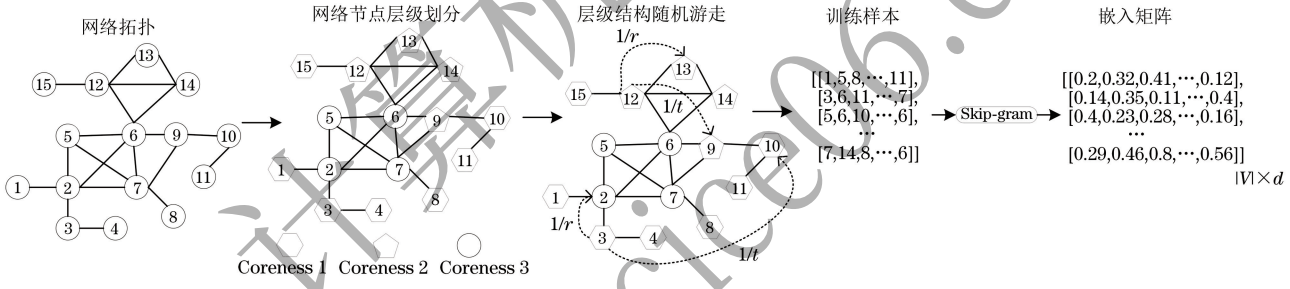


图 1 KCNE 算法的总体流程

Fig. 1 Overall workflow of the KCNE algorithm

2.2.1 网络节点层级划分

k 核分解是一种常用于网络结构分析的方法，它刻画了网络中节点的结构性和层次性[32]。

直观来说，对于任意一个无向图 $G(V,E)$，它的 k 核子图 G' 与原始图 G 存在如下关系：$\exists G'=(V',E')$，$V'\subseteq V$，$E'\subseteq E$，$\forall v\in V'$，$d(v)\geqslant k$，其中，$d(u)$ 代表节点 u 的度。如果节点 v 属于 k 核，而不属于 $k+1$ 核，则节点 v 的核值为 k，也就是说，同属于相同子图的节点具有同样的核值 k(同一个子图下的节点具有相同的层级编号)。此外，k 核分解的核心思想是迭代删除度数小于 k 的节点以及对应的边(也称为剪枝)。

2.2.2 层级结构随机游走

传统的随机游走主要是基于邻域的方式进行采样。假设当前时刻访问到节点 x，则在下一时刻将以一定的概率访问到它的一阶邻居节点 y，这种采样方式遵循马尔可夫链。但是，这样的随机游走是有限度的，通常会局限于某一社区内部，这导致随机游走不能获取到丰富的上下文信息，缺少对具有相似层级结构特征的节点的考虑。因此，本文提出了层级结构随机游走的采样策略来捕捉节点的局部结构信息和层次结构信息。该策略基于 k 核分解[32]为每一个节点分配一个层级编号，使得没有直接关联的节点通过层级编号强化了节点间的约束，从而在游走过程中能捕获到高阶且具有相似结构特征的上下文信息。

具体来说，本文采用灵活的采样方式来获取节点的上下文信息。假设在 t 时刻采样得到的节点为 x，则在下一时刻，采样得到的节点 y 遵循式(1)：

$$p(v_{t+1}=y \mid v_t=x)=\begin{cases}\dfrac{\delta_{xy}}{Z}, & (x,y)\in E \text{ or } H_x=H_y\\ 0, & \text{otherwise}\end{cases} \tag{1}$$

式中：δ_{xy} 代表节点 x 和节点 y 之间的采样概率；Z 代表归一化常数；H_x 代表节点 x 的层级编号。在 KCNE 算法中，使用参数 r 和 t 控制随机游走过程中对节点的一阶邻域及相同层级节点的采样。假设节点 x 和 y 之间的转移概率为 $\delta_{xy}=\beta_{rt}(x,y)\cdot w_{xy}$，其中，$w_{xy}$ 代表节点 x 和节点 y 之间的权重，对于非权重网络，$w_{xy}=1$，β_{rt} 可用式(2)来表示：

$$\beta_{rt}(x,y)=\begin{cases}1/r, & (x,y)\in E\\ 1/t, & H_x=H_y\end{cases} \tag{2}$$

层级结构随机游走获取节点序列的步骤如算法 2 所示。

算法 2 Hierarchical_structure-based_random_walk

输入 无向图 G,网络层级列表 k_list,游走长度 L,初始节点 v,参数 r 和 t

输出 长度为 L 的节点序列 walk

1. 初始化 walk 列表为 v
2. while (len(walk)<L) do:
3. cur=walk[-1]
4. 获取到当前节点 cur 的邻居节点和同层级节点,记为 ψ_{cur}
5. 根据式(1)和式(2)计算当前节点 cur 与邻域节点和同层级节点间的转移概率,记为 ξ_{cur}
6. u =AliasSample(ψ_{cur},ξ_{cur})//采样得到节点 u
7. walk.append(u)
8. end while
9. return walk

2.2.3 学习节点嵌入

对于层级结构随机游走获取到的一组节点序列 $S=\{s_1,s_2,\cdots,s_k\}$,可以将其中每一个节点的游走序列表示为 $s_i=\{v_1,v_2,\cdots,v_L\}$,然后将节点序列 s_i 作为词序列输入到 Skip-gram 模型[7]中来学习节点的向量表示。

将给定长度为 L 的节点序列 $s_i=\{v_1,v_2,\cdots,v_L\}$ 作为 Skip-gram 模型的输入,该模型的优化目标函数可定义如下:

$$\Gamma(s_i)=\frac{1}{L}\sum_{t=1}^{L}\sum_{-w\leqslant j\leqslant w,j\neq 0}\log_a p(v_{t+j}\mid v_t) \tag{3}$$

式中:w 代表节点 v_t 的局部上下文大小;v_{t+j} 表示节点 v_t 的上下文节点。w 越大则获取到的训练样本越多。对于条件概率 $p(v_{t+j}\mid v_t)$,使用 Softmax 函数来近似,该函数定义如下:

$$p(v_{t+j}\mid v_t)=\frac{1}{1+\mathrm{e}^{-\boldsymbol{f}(v_t)\boldsymbol{f}(v_{t+j})}} \tag{4}$$

式中:$\boldsymbol{f}(v_t)$ 代表节点 v_t 的嵌入向量,$\boldsymbol{f}(v_{t+j})$ 也类似。

3 实验

本节在两个有监督学习任务——链路预测和多标签节点分类上评估本文所提出的 KCNE 算法的性能,同时还分析了不同参数对 KCNE 算法性能的影响。

3.1 数据集简介

本文使用不同领域的网络来验证 KCNE 算法的性能。对于节点分类任务,使用交通网络[33](europe、usa)和电子邮件网络[34](email-eu-core)。交通网络的节点被划分为 4 个类别;而电子邮件网络的节点被分类为 42 个类别。对于链路预测任务,则使用交通网络(europe, usa)、博客网络(polblogs)、社交网络(socfb-simmons81)、交互网络(ia-fb-message)和电子邮件网络(email-unix)。这些网络可在文献[35]中获取。数据集的详细统计信息如表 1 所示。

表 1 真实数据集的详细统计信息

Table 1 Detailed statistical information of the real datasets

数据集	节点数/个	边数/条	节点类别表示
europe	399	5 993	4
usa	1 190	13 599	4
email-eu-core	1 005	25 571	42
polblogs	1 490	17 139	#
socfb-simmons81	1 518	32 988	#
ia-fb-message	1 266	6 451	#
email-unix	1 133	5 451	#

3.2 基准算法

为了评估 KCNE 算法的性能,本文使用 8 个经典的网络嵌入算法作为基准方法。各算法的详细信息如下:

DeepWalk[6]:该算法基于随机游走的方式将获取到的节点序列作为训练样本输入到 Skip-gram 模型[7]中学习得到节点表示。

node2vec[8]:该算法在随机游走的过程中,通过设置两个超参数控制 BFS 和 DFS 的采样策略,然后将节点序列输入到 Skip-gram 模型中学习得到节点表示。

CARE[10]:该算法通过定制的随机游走方式对当前节点的一阶邻域和社区节点进行采样,使得学习到的节点嵌入保留了节点的社区结构信息。

NRL_RWCE[11]:该算法是一种基于随机游走和社区增强的节点嵌入算法,使得学习到的节点表示保留了社区信息。

Role2Vec[12]:该算法使用属性随机游走的方式来捕捉节点的行为角色,使得学习到的节点表示保留了节点的结构相似性。

RARE[13]:该算法通过角色随机游走的方式来捕获节点的社区和角色结构信息,使得学习到的节点嵌入保留了两者的结构信息。

GEMSEC[23]:该算法是一种基于序列采样的学习模型,同时考虑了嵌入和聚类的问题。

NodeSketch[36]：该算法通过递归草图的方式使得学习到的节点表示保留了高阶节点的近邻性。

3.3 参数设置与评估指标

在对比实验中，统一设置节点的嵌入维度 $d=128$。对于 DeepWalk 与 node2vec 算法，设置游走长度 $L=80$，设置每一个节点的随机游走次数 $K=10$，设置窗口大小 $W=10$。对于 node2vec 算法，分别设置参数 $p=4$，$q=1$。对于 Role2Vec 算法，分别设置参数 $p=1$，$q=0.5$。对于 CARE 算法，设置参数 $\alpha=0.1$。对于 RARE 算法，分别设置参数 $r=0.25$，$t=4$。其余参数使用原始文章中的默认值。对于本文提出的 KCNE 算法，分别从{0.25，0.5，1，2，4}中选择参数 r 和 t 的值，其余参数的设置与 DeepWalk 算法相同。

对于多标签节点分类任务，本文训练一个 One-vs-Rest 逻辑回归模型来构建一个多类别分类器，配置使用 LIBLINEAR 求解器，并进行 10 000 次最大迭代来寻找最佳模型参数。然后使用已训练的分类器模型预测节点标签，并使用 Micro-F1 和 Macro-F1 指标来评估算法性能[37]。

对于链路预测任务，通过训练一个逻辑回归分类器，并使用 L2 正则化来防止过拟合，使用 LBFGS 求解器来进行参数优化，最多进行 500 次迭代来找到最佳的模型参数。使用标准的链路预测评估指标——曲线下面积(AUC)[27]来评估算法的性能。此外，使用 L2 算子来构造网络中链路的表示，具体构造公式如下：

$$\|\boldsymbol{f}(u)-\boldsymbol{f}(v)\|_2^{-i}=|f(u)_i-f(v)_i|^2 \tag{5}$$

式中：$f(u)_i$ 和 $f(v)_i$ 分别代表节点 u 和节点 v 的第 i 个向量值。

3.4 实验结果分析

对于链路预测与多标签节点分类，本文对每一个数据集运行 10 次算法，并取 10 次结果的均值和标准差作为最终的实验结果，由于篇幅限制，节点分类的标准差不在表格中给出。

3.4.1 链路预测任务

对于链路预测任务，本文随机选取网络中 50% 的边作为测试样本，剩下的边作为训练样本。此外，通过构造与正样本相同数量且不存在于网络中的边作为负样本。

在 6 个数据集上执行本文提出的算法与基准方法，实验结果如表 2 所示，表中加粗的数据表示最优结果，横线标记的数据表示次优结果。在 europe 和 usa 数据集中，KCNE 算法相比次优的 NodeSketch 算法性能提升约 2%；在 polblogs 数据集中，KCNE 算法相比次优的 RARE 算法有 1%的性能提升；在 email-unix 数据集中，KCNE 算法相比次优的 Role2Vec 性能提升约 2%；而在 socfb-simmons81 和 ia-fb-message 数据集中，KCNE 算法相比次优的 NRL_RWCE 算法和 RARE 算法性能分别提升了约 4%和 1%。

表 2 不同算法在不同数据集上的 AUC 值

Table 2 AUC values of different algorithms on various datasets

算法	europe	usa	polblogs	socfb-simmons81	ia-fb-message	email-unix
DeepWalk	0.788 5(0.004)	0.836 5(0.006)	0.667 7(0.012)	0.749 2(0.003)	0.640 8(0.007)	0.587 7(0.011)
node2vec	0.783 6(0.006)	0.831 0(0.006)	0.679 0(0.012)	0.744 1(0.002)	0.643 9(0.012)	0.586 0(0.008)
Role2Vec	0.706 6(0.008)	0.819 9(0.008)	0.706 8(0.006)	0.677 0(0.005)	0.563 6(0.009)	<u>0.689 0(0.005)</u>
GEMSEC	0.755 0(0.004)	0.706 1(0.003)	0.571 2(0.007)	0.657 5(0.002)	0.591 4(0.009)	0.560 8(0.000)
NodeSketch	<u>0.806 4(0.003)</u>	<u>0.837 4(0.000)</u>	0.805 3(0.001)	0.624 6(0.000)	0.613 8(0.007)	0.665 8(0.003)
CARE	0.777 2(0.016)	0.775 0(0.008)	0.649 4(0.010)	0.726 2(0.004)	0.534 3(0.008)	0.677 8(0.007)
NRL_RWCE	0.621 9(0.035)	0.822 9(0.016)	0.811 6(0.028)	<u>0.779 2(0.006)</u>	0.688 3(0.005)	0.603 3(0.036)
RARE	0.745 5(0.013)	0.810 7(0.010)	<u>0.824 8(0.010)</u>	0.735 8(0.011)	<u>0.748 8(0.020)</u>	0.598 0(0.018)
KCNE	**0.821 2(0.003)**	**0.850 5(0.003)**	**0.830 5(0.007)**	**0.817 8(0.011)**	**0.755 9(0.010)**	**0.702 8(0.006)**

3.4.2 多标签节点分类任务

表 3～表 8 分别展示了在不同数据集上以不同的训练比例进行多标签节点分类的结果。在不同的训练比例下，本文随机选取一定比例的节点作为训练样本，而其余的节点则构成测试样本。表中加粗的数据表示最佳结果，横线标记的数据表示次优结果。

从这些结果中，可以得到如下的结论：

1)本文提出的 KCNE 算法的性能总体上优于基准方法。在 europe 数据集中，KCNE 算法在 Micro-F1 和 Macro-F1 指标下，相比次优的 RARE 算法性能提升最高分别约 5%和 4%；在 usa 数据集中，对于 Micro-F1 和 Macro-F1 指标，KCNE 算法在多个训练比例下相比次优的 RARE 算法性能最

大提升约 2%；而在 email-eu-core 数据集中，对于指标 Micro-F1，虽然在训练比例为 10%～30%时，NRL_RWCE 算法性能优于本文提出的 KCNE 算法，但是在其他训练比例下 KCNE 算法仍然优于所有的基准方法。类似地，对于 Macro-F1 指标，除了训练比例为 10%的情况，KCNE 算法均优于所有的基准方法。

2)从训练比例 10%～90%的情况来看，在大多数情况下，KCNE 算法性能均优于基准算法，表明了 KCNE 算法在处理网络稀疏性上具有较好的性能。

总体而言，本文提出的 KCNE 算法在多标签节点分类任务中相比于基准算法性能上有较大的改善，并且在处理网络稀疏性上具有不错的效果。

表 3 europe 数据集的节点分类精度(Micro-F1)

Table 3 Node classification accuracy on the europe dataset (Micro-F1)

算法	10%训练比例	20%训练比例	30%训练比例	40%训练比例	50%训练比例	60%训练比例	70%训练比例	80%训练比例	90%训练比例
DeepWalk	0.330 3	0.348 1	0.350 0	0.357 6	0.365 9	0.373 7	0.390 1	0.404 9	0.407 5
node2vec	0.330 3	0.346 5	0.344 0	0.353 5	0.368 2	0.378 0	0.371 5	0.372 5	0.343 5
NodeSketch	0.324 4	0.348 4	0.371 0	0.379 1	0.392 4	0.398 7	0.393 3	0.428 7	0.422 5
GEMSEC	0.311 1	0.349 6	0.371 4	0.368 3	0.368 0	0.374 3	0.389 1	0.405 0	0.407 5
NRL_RWCE	0.287 5	0.297 3	0.309 8	0.316 0	0.317 7	0.328 3	0.329 5	0.343 5	0.336 5
Role2Vec	0.275 0	0.290 5	0.301 7	0.306 8	0.310 3	0.312 7	0.320 3	0.320 5	0.321 7
CARE	0.302 4	0.319 1	0.337 6	0.344 3	0.358 9	0.367 2	0.372 9	0.382 2	0.400 9
RARE	0.385 0	0.412 8	0.429 5	0.445 0	0.454 5	0.457 6	0.464 3	0.466 3	0.475 0
KCNE	**0.419 8**	**0.451 7**	**0.464 4**	**0.472 3**	**0.475 5**	**0.483 0**	**0.491 5**	**0.501 7**	**0.523 0**

表 4 europe 数据集的节点分类精度(Macro-F1)

Table 4 Node classification accuracy on the europe dataset (Macro-F1)

算法	10%训练比例	20%训练比例	30%训练比例	40%训练比例	50%训练比例	60%训练比例	70%训练比例	80%训练比例	90%训练比例
DeepWalk	0.328 2	0.347 7	0.349 8	0.358 2	0.366 4	0.372 2	0.387 8	0.398 9	0.397 8
node2vec	0.328 5	0.346 3	0.344 1	0.353 1	0.367 6	0.377 2	0.371 9	0.370 1	0.337 8
NodeSketch	0.320 9	0.346 4	0.368 0	0.376 7	0.388 7	0.397 3	0.389 6	0.419 3	0.416 2
GEMSEC	0.290 0	0.336 0	0.354 9	0.351 4	0.350 4	0.354 7	0.364 7	0.377 9	0.373 7
NRL_RWCE	0.240 1	0.265 0	0.286 1	0.295 6	0.298 4	0.310 0	0.309 9	0.321 2	0.306 2
Role2Vec	0.250 5	0.283 1	0.298 2	0.304 4	0.308 1	0.310 3	0.317 1	0.315 4	0.312 0
CARE	0.289 4	0.315 0	0.336 1	0.343 7	0.359 0	0.367 5	0.372 6	0.381 3	0.394 8
RARE	0.343 1	0.390 9	0.414 0	0.430 7	0.442 0	0.444 6	0.450 5	0.452 2	0.454 2
KCNE	**0.388 7**	**0.425 2**	**0.438 6**	**0.447 6**	**0.451 5**	**0.459 5**	**0.464 8**	**0.474 3**	**0.488 8**

表 5 usa 数据集的节点分类精度(Micro-F1)

Table 5 Node classification accuracy on the usa dataset (Micro-F1)

算法	10%训练比例	20%训练比例	30%训练比例	40%训练比例	50%训练比例	60%训练比例	70%训练比例	80%训练比例	90%训练比例
DeepWalk	0.414 4	0.422 1	0.439 2	0.457 3	0.481 7	0.502 4	0.511 2	0.523 1	0.536 6
node2vec	0.418 3	0.425 5	0.444 6	0.478 0	0.495 0	0.515 4	0.527 0	0.533 6	0.542 3
NodeSketch	0.364 1	0.402 7	0.420 3	0.422 4	0.424 9	0.429 6	0.439 6	0.441 5	0.453 7
GEMSEC	0.473 6	0.487 4	0.486 2	0.493 2	0.496 9	0.500 8	0.504 7	0.515 1	0.523 5
NRL_RWCE	0.555 5	0.583 1	0.596 4	0.604 8	0.609 1	0.614 2	0.614 8	0.614 2	0.610 7
Role2Vec	0.360 9	0.374 9	0.376 2	0.386 8	0.398 4	0.397 0	0.410 0	0.414 7	0.415 9
CARE	0.387 9	0.398 6	0.409 8	0.429 1	0.443 3	0.457 6	0.468 6	0.471 5	0.468 9
RARE	**0.594 3**	0.609 8	0.617 3	0.621 9	0.623 2	0.624 0	0.625 8	0.628 7	0.630 0
KCNE	0.589 5	**0.610 0**	**0.620 6**	**0.627 8**	**0.633 2**	**0.639 6**	**0.643 6**	**0.646 8**	**0.645 4**

表 6 usa 数据集的节点分类精度(Macro-F1)

Table 6 Node classification accuracy on the usa dataset (Macro-F1)

算法	10%训练比例	20%训练比例	30%训练比例	40%训练比例	50%训练比例	60%训练比例	70%训练比例	80%训练比例	90%训练比例
DeepWalk	0.411 9	0.421 3	0.439 2	0.456 4	0.480 0	0.499 9	0.508 4	0.519 0	0.530 3
node2vec	0.416 8	0.424 3	0.443 8	0.476 5	0.492 9	0.513 1	0.525 6	0.531 5	0.539 8
NodeSketch	0.357 2	0.395 0	0.410 6	0.413 8	0.415 9	0.420 1	0.430 1	0.434 9	0.443 6
GEMSEC	0.459 4	0.472 1	0.471 0	0.476 2	0.479 5	0.483 8	0.487 0	0.497 6	0.504 9
NRL_RWCE	0.531 2	0.558 3	0.572 2	0.580 4	0.584 6	0.589 5	0.589 9	0.589 4	0.586 9
Role2Vec	0.353 2	0.371 4	0.372 7	0.383 9	0.396 3	0.394 1	0.406 5	0.410 6	0.410 5
CARE	0.386 1	0.399 3	0.409 9	0.428 9	0.442 0	0.455 7	0.466 9	0.468 6	0.463 1
RARE	**0.573 2**	0.594 9	0.605 4	0.611 0	0.612 7	0.613 6	0.615 1	0.616 9	0.617 7
KCNE	0.572 0	**0.597 4**	**0.609 8**	**0.617 7**	**0.623 5**	**0.629 9**	**0.634 4**	**0.637 3**	**0.634 8**

表 7 email-eu-core 数据集的节点分类精度(Micro-F1)

Table 7 Node classification accuracy on the email-eu-core dataset (Micro-F1)

算法	10%训练比例	20%训练比例	30%训练比例	40%训练比例	50%训练比例	60%训练比例	70%训练比例	80%训练比例	90%训练比例
DeepWalk	0.537 3	0.631 6	0.658 4	0.680 5	0.686 4	0.691 6	0.702 7	0.710 7	0.733 5
node2vec	0.560 2	0.638 8	0.669 5	0.693 4	0.704 9	0.717 6	0.728 3	0.737 3	0.740 4
NodeSketch	0.121 7	0.155 8	0.182 7	0.219 2	0.245 8	0.267 8	0.287 5	0.317 1	0.319 1
GEMSEC	**0.592 2**	0.626 2	0.645 7	0.659 4	0.666 1	0.666 8	0.676 0	0.677 2	0.693 9
NRL_RWCE	0.588 4	**0.668 1**	**0.700 3**	0.716 1	0.726 4	0.736 6	0.739 4	0.746 1	0.749 3
Role2Vec	0.250 7	0.313 3	0.353 1	0.373 9	0.389 0	0.399 2	0.404 3	0.408 0	0.416 1
CARE	0.526 4	0.605 9	0.638 5	0.656 8	0.668 7	0.675 4	0.681 4	0.687 2	0.686 0
RARE	0.510 7	0.623 0	0.668 2	0.695 2	0.713 6	0.724 8	0.734 0	0.744 5	0.754 7
KCNE	0.557 3	0.653 9	0.693 8	**0.716 9**	**0.733 2**	**0.744 9**	**0.752 8**	**0.757 9**	**0.759 4**

表 8 email-eu-core 数据集的节点分类精度(Macro-F1)

Table 8 Node classification accuracy on the email-eu-core dataset (Macro-F1)

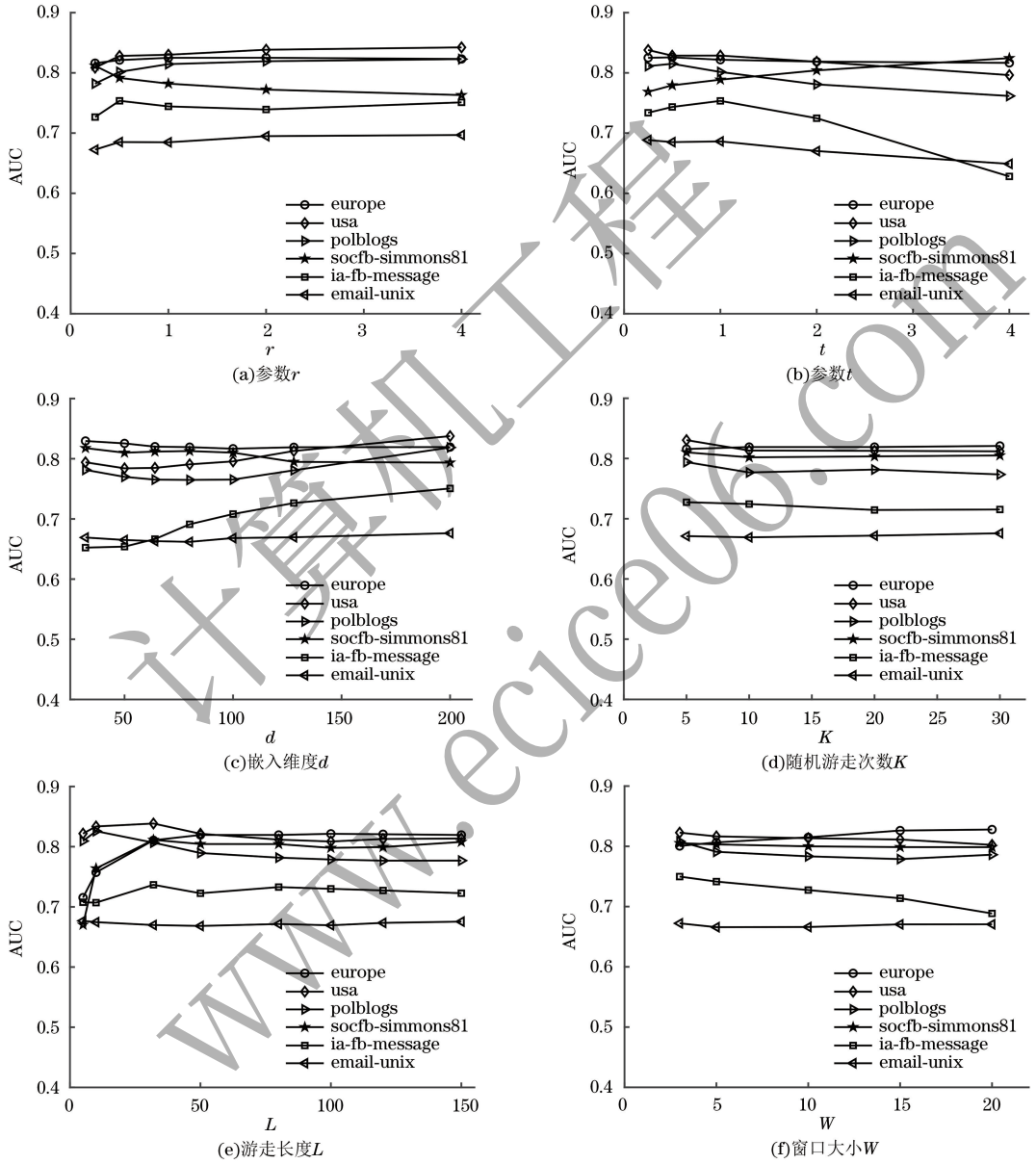
算法	10%训练比例	20%训练比例	30%训练比例	40%训练比例	50%训练比例	60%训练比例	70%训练比例	80%训练比例	90%训练比例
Deepwalk	0.301 7	0.392 2	0.431 4	0.468 5	0.476 5	0.492 1	0.509 4	0.524 3	0.554 6
node2vec	0.308 2	0.388 0	0.436 8	0.473 7	0.499 8	0.513 5	0.530 2	0.546 4	0.576 2
NodeSketch	0.060 9	0.081 1	0.104 9	0.125 9	0.139 1	0.154 1	0.164 9	0.188 0	0.186 4
GEMSEC	**0.343 3**	0.384 0	0.413 0	0.428 6	0.442 1	0.450 8	0.462 9	0.467 3	0.526 6
NRL_RWCE	0.307 6	0.393 3	0.440 6	0.467 4	0.490 5	0.511 9	0.532 0	0.557 2	0.575 7
Role2Vec	0.088 0	0.127 5	0.156 1	0.172 2	0.192 2	0.200 9	0.214 1	0.226 7	0.241 2
CARE	0.282 6	0.366 2	0.408 8	0.436 5	0.460 5	0.475 9	0.490 6	0.497 4	0.525 6
RARE	0.263 6	0.362 7	0.411 2	0.449 4	0.474 8	0.494 7	0.518 9	0.545 8	0.582 7
KCNE	0.306 1	**0.401 5**	**0.452 0**	**0.484 5**	**0.511 4**	**0.530 7**	**0.552 1**	**0.576 0**	**0.593 1**

3.4.3 参数敏感性与嵌入生成时间

本节在 6 个数据集上对链路预测任务进行参数敏感性分析与实验。本文随机选取网络中 50%的链路作为测试样本,剩下的链路作为训练样本用来生成节点的表示。在进行参数敏感性分析时,将其他参数固定,改变特定的参数。对于每一次参数的改变,运行 10 次 KCNE 算法,并取 10 次结果的均值作为最终的实验结果。

图 2 显示了不同参数对 KCNE 算法的影响。对于参数 r,从图 2(a)中可以看出,当 $r>0.5$ 时,所有数据集的 AUC 值趋于稳定;对于参数 t,如图 2(b)所示,当 $t=1$ 时,ia-fb-message 数据集取得最优值,对于 socfb-simmons81 数据集,随着 t 值的增大,AUC 值也逐渐增大,其他数据集在不同的 t 值下,AUC 值从整体上看波动幅度较小;对于参数 d,如图 2(c)所示,对于大多数数据集来说,随着嵌入维度的增大,AUC 值

的曲线波动较小，但是对于 ia-fb-message 数据集，AUC 值呈现逐渐上升的趋势；对于参数 K，如图 2(d)所示，该参数对所有数据集 AUC 值的影响都较小；对于参数 L，如图 2(e)所示，随着 L 值的增加，随机游走能捕捉到许多结构不同的节点上下文信息，使得在大部分数据集中的 AUC 值得到了明显的改善，但是当 $L>50$ 时，所有数据集的 AUC 值逐渐趋于稳定；参数 W 的影响趋势如图 2(f)所示。



(a)参数r　(b)参数t　(c)嵌入维度d　(d)随机游走次数K　(e)游走长度L　(f)窗口大小W

图 2　KCNE 算法在不同数据集上的参数敏感性分析

Fig. 2　Parameter sensitivity analysis of KCNE algorithm on different datasets

不同算法在不同数据集上的嵌入生成时间如表 9 所示。GEMSEC 算法在所有数据集上耗费最长的嵌入生成时间，这主要归因于其涉及聚类过程，导致时间成本较高。本文提出的 KCNE 算法在所有数据集上的嵌入生成时间明显优于基于角色结构的 Role2Vec 算法和 RARE 算法，其中，角色发现与采样的过程耗费了较多时间。针对 KCNE 算法在 polblogs 和 socfb-simmons81 数据集上与 RARE 算法的嵌入生成时间的比较，以边数较多为例，当节点个数相差不大时，随着边数的增多，KCNE 算法在进行剪枝操作时所需的时间较多。在 socfb-simmons81 数据集上，KCNE 算法生成嵌入的时间与 RARE 算法相比差距不大。然而，相对于 socfb-simmons81 数据集在边数减少约一半的 polblogs 数据集上，KCNE 算法生成嵌入的时间与 RARE 算法所需时间的差距就显得更为明显。此外，采用 Skip-gram 模型来学习节点表示的算法，包括 DeepWalk、node2vec、CARE 和 NRL_RWCE 算法，嵌入生成时间较短。DeepWalk 基于邻域的随机采样策略；node2vec 则基于两个可调参

数进行随机采样；而 CARE 算法则利用一个超参数控制邻域与同社区节点间的采样；NRL_RWCE 算法采用一种自适应游走参数的采样方式，从而显著节约了采样时间。与这些算法相比，KCNE 算法在网络节点层级划分和计算当前节点转移概率的采样中消耗过多时间。实验结果表明，本文提出的算法在运行效率上具有较好的竞争性。值得注意的是，本文为了确保实验的公平性，所有的实验都使用同一台计算机上的中央处理器（CPU）运行。

表 9 不同算法在不同数据集上的嵌入生成时间

Table 9 Embedding generation time of different algorithms on various datasets 单位：s

算法	europe	usa	polblogs	socfb-simmons81	ia-fb-message	email-unix
DeepWalk	3.419	9.378	17.006	13.224	9.887	7.992
node2vec	6.441	27.294	32.794	32.698	18.890	18.335
Role2Vec	25.872	68.964	101.274	115.799	90.258	70.120
GEMSEC	123.529	337.747	428.274	433.829	361.987	364.895
NodeSketch	10.848	17.085	28.797	50.285	6.923	3.925
CARE	5.184	13.942	17.480	21.956	14.923	13.590
NRL_RWCE	1.098	2.187	4.210	8.630	1.008	1.345
RARE	11.092	62.642	83.179	87.718	62.641	49.051
KCNE	9.264	53.703	66.919	84.291	60.770	44.717

4 结束语

本文提出了一种简单有效的网络嵌入方法 KCNE。该方法首先将网络节点划分为多个不同的层次集合；然后通过两个超参数控制对节点的一阶邻居和同层级节点成员的采样，这种采样方式能有效地捕获节点间的相似层级结构特征。在 7 个真实数据集上的实验结果表明，KCNE 算法在链路预测和多标签节点分类任务上的性能优于基准方法，在运行效率上优于 GEMSEC、Role2Vec 和 RARE 算法，同时该算法也表现出较好的鲁棒性。本文算法具有较好的可扩展性，下一步将重点研究该算法在属性网络、多重网络、时序网络等中的应用。

参考文献

[1] ZHANG H G, CHEN X L, PENG Y, et al. The interaction of multiple information on multiplex social networks[J]. Information Sciences, 2022, 605: 366-380.

[2] 白明昌. 基于折叠路径聚合的属性网络节点嵌入方法[J]. 计算机工程, 2023, 49(7): 76-84.
BAI M C. Node embedding method based on folded path aggregation on attributed network [J]. Computer Engineering, 2023, 49(7): 76-84. (in Chinese)

[3] KAZIENKO P, KAJDANOWICZ T. Label-dependent node classification in the network[J]. Neurocomputing, 2012, 75(1): 199-209.

[4] KUMAR A, SINGH S S, SINGH K, et al. Link prediction techniques, applications, and performance: a survey[J]. Physica A-Statistical Mechanics and Its Applications, 2020, 553: 124289.

[5] PIO G, CECI M, PRISCIANDARO F, et al. Exploiting causality in gene network reconstruction based on graph embedding[J]. Machine Learning, 2020, 109(6): 1231-1279.

[6] PEROZZI B, AL-RFOU R, SKIENA S. DeepWalk: online learning of social representations[C]//Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA: ACM Press, 2014: 701-710.

[7] MIKOLOV T, CHEN K, CORRADO G, et al. Efficient estimation of word representations in vector space[EB/OL]. [2023-11-01]. https://arxiv.org/abs/1301.3781v1.

[8] GROVER A, LESKOVEC J. node2vec: scalable feature learning for networks[C]//Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA: ACM Press, 2016: 855-864.

[9] TANG J, QU M, WANG M Z, et al. LINE: large-scale information network embedding[C]//Proceedings of the 24th International Conference on World Wide Web. Florence, Italy: International World Wide Web Conferences Steering Committee, 2015: 1067-1077.

[10] KEIKHA M M, RAHGOZAR M, ASADPOUR M. Community aware random walk for network embedding[J]. Knowledge-Based Systems, 2018, 148: 47-54.

[11] GUO K, WANG Q Z, LIN J Q, et al. Network representation learning based on community-aware and adaptive random walk for overlapping community detection [J]. Applied Intelligence, 2022, 52(9): 9919-9937.

[12] AHMED N K, ROSSI R A, LEE J B, et al. role2vec: role-based network embeddings[EB/OL]. [2023-11-01]. http://ryanrossi.com/pubs/role2vec-DLG-KDD.pdf.

[13] ZHANG H G, KOU G, PENG Y, et al. Role-aware random walk for network embedding[J]. Information Sciences, 2024, 652: 119765.

[14] GUO X, JIAO P F, ZHANG W, et al. Representation learning on heterostructures via heterogeneous anonymous walks[J]. IEEE Transactions on Neural Networks and Learning Systems, 2024, 35(7): 9538-9552.

[15] JIANG J Y, LI Z Y, JU C J T, et al. MARU: meta-context aware random walks for heterogeneous network representation learning[C]//Proceedings of the 29th ACM International Conference on Information & Knowledge Management. New York, USA: ACM Press, 2020: 575-584.

[16] LI X, WEI W, ZHANG R Z, et al. Representation learning of enhanced graphs using random walk graph convolutional

network[J]. ACM Transactions on Intelligent Systems and Technology, 2023, 14(3): 1-21.
[17] ZHANG Y Y, SHI Z, FENG D, et al. Degree-biased random walk for large-scale network embedding[J]. Future Generation Computer Systems, 2019, 100: 198-209.
[18] LIU Q, SHU H, YUAN M, et al. Fuzzy hierarchical network embedding fusing structural and neighbor information[J]. Information Sciences, 2022, 603: 130-148.
[19] 焦鹏飞, 潘婷, 金弟, 等. 角色导向的网络表示学习综述[J]. 计算机学报, 2023, 46(2): 274-303.
JIAO P F, PAN T, JIN D, et al. A survey on role-guided network representation learning[J]. Chinese Journal of Computers, 2023, 46(2): 274-303. (in Chinese)
[20] DUONG C T, NGUYEN T T, HOANG T D, et al. Deep MinCut: learning node embeddings by detecting communities[J]. Pattern Recognition, 2023, 134: 109126.
[21] TU C C, ZENG X K, WANG H, et al. A unified framework for community detection and network representation learning[J]. IEEE Transactions on Knowledge and Data Engineering, 2019, 31(6): 1051-1065.
[22] WANG X, CUI P, WANG J, et al. Community preserving network embedding[J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2017, 31(1): 203-209.
[23] ROZEMBERCZKI B, DAVIES R, SARKAR R, et al. GEMSEC: graph embedding with self clustering[C]// Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. New York, USA: ACM Press, 2020: 65-72.
[24] 吕少卿, 赵雪莉, 张潘, 等. 一种保留社区结构信息的网络嵌入算法[J]. 计算机工程, 2021, 47(12): 122-130.
LÜ S Q, ZHAO X L, ZHANG P, et al. A network embedding algorithm preserving community structure information[J]. Computer Engineering, 2021, 47(12): 122-130. (in Chinese)
[25] 李亚芳, 梁烨, 冯韦玮, 等. 基于社区优化的深度网络嵌入方法[J]. 计算机应用, 2021, 41(7): 1956-1963.
LI Y F, LIANG Y, FENG W W, et al. Deep network embedding method based on community optimization[J]. Journal of Computer Applications, 2021, 41(7): 1956-1963. (in Chinese)
[26] MA X W, QIN G, QIU Z Y, et al. RiWalk: fast structural node embedding via role identification[C]//Proceedings of the IEEE International Conference on Data Mining (ICDM). Washington D. C., USA: IEEE Press, 2019: 478-487.
[27] ZHANG H G, KOU G. Role-based multiplex network embedding[C]// Proceedings of the 39th International Conference on Machine Learning. New York, USA: PMLR, 2022: 26265-26280.
[28] WANG X, JIAN S L, LU K, et al. RED: learning the role embedding in networks via discrete-time quantum walk[J]. Applied Intelligence, 2022, 52(2): 1493-1507.
[29] LI S, HUANG F H. A node role embedding method based on neighborhood clustering coefficient[C]//Proceedings of the 6th International Conference on Data Storage and Data Engineering (DSDE). Washington D. C., USA: IEEE Press, 2023: 1-5.
[30] SUN Y H, JIA M Y, LIU C, et al. Heterogeneous network representation learning based on role feature extraction[J]. Pattern Recognition, 2023, 144: 109870.
[31] 李泽水, 冀俊忠, 杨翠翠. 基于边权重信息深度网络嵌入的 PPIN 功能模块检测[J]. 计算机工程, 2023, 49(8): 69-76.
LI Z S, JI J Z, YANG C C. Functional module detection based on deep network embedding of edge weighing information in PPIN[J]. Computer Engineering, 2023, 49(8): 69-76. (in Chinese)
[32] KONG Y X, SHI G Y, WU R J, et al. k-core: theories and applications[J]. Physics Reports, 2019, 832: 1-32.
[33] RIBEIRO L F R, SAVERESE P H P, FIGUEIREDO D R. struc2vec: learning node representations from structural identity[C]// Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA: ACM Press, 2017: 385-394.
[34] YIN H, BENSON A R, LESKOVEC J, et al. Local higher-order graph clustering[C]//Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA: ACM Press, 2017: 555-564.
[35] ROSSI R, AHMED N. The network data repository with interactive graph analytics and visualization[J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2015, 29(1): 4292-4293.
[36] YANG D Q, ROSSO P, LI B, et al. NodeSketch: highly-efficient graph embeddings via recursive sketching[C]// Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA: ACM Press, 2019: 1162-1172.
[37] MO Y J, LEI Y J, SHEN J L, et al. Disentangled multiplex graph representation learning[C]//Proceedings of the 40th International Conference on Machine Learning. New York, USA: PMLR, 2023: 24983-25005.

编辑 金胡考