基于超级参数调整的网络表示学习算法性能公平比较框架

郭梦影[1),2)] 孙振宇[1),2)] 朱妤晴[3),4)] 包云岗[1),2)]

[1)](中国科学院计算技术研究所先进计算机系统研究中心 北京 100190)

[2)](中国科学院大学 北京 100049)

[3)](北京信息科学与技术国家研究中心(清华大学) 北京 100084)

[4)](大数据系统软件国家工程实验室 北京 100084)

摘 要 网络结构数据在现今生活中广泛存在,但由于数据结构稀疏、规模较大等特性,难以直接利用现有的机器学习算法对数据进行分析.网络表示学习算法的出现,通过将高维数据映射到低维向量空间,解决了上述问题.但是网络表示学习算法中存在大量超级参数,参数的选择与数据分析任务密切相关且对算法性能有明显影响,如何针对数据分析任务,通用地对多种网络表示学习算法进行超级参数调整,以获取不同算法的最优性能,实现算法间性能的公平比较,从而选择出最优者对数据进行分析,是一个亟待解决的问题.此外,对算法进行超级参数调整通常需要花费较长时间,且由于网络结构数据规模通常较大,还会有内存占用过高问题的存在,因此如何能够在有资源限制(时间、内存占用)的条件下进行超级参数调整,是面临的另一个问题.基于上述两个问题,本文提出了基于超级参数调整的网络表示学习算法性能公平比较框架 JITNREv,能够在有资源限制的条件下通用对多种网络表示学习算法进行超级参数调整,通过获取不同算法针对相同数据分析任务的性能最优值,实现算法之间的性能公平比较.该框架具有 4 个松耦合且可扩展的组件,组件间仅通过数据流进行交互,并在闭环结构中完成样本的测试优化,满足了框架的通用性.JITNREv 基于拉丁超立方采样对超级参数进行采样;根据"当前最优值附近,有更大概率出现更优值"的假设对采样范围进行剪枝;针对超大规模数据集,提出了图粗化方式在保留数据结构的基础上压缩数据规模,满足了资源限制条件下对超级参数进行调整的要求.框架还融合了网络表示学习算法常用的评测数据集、评测指标和数据分析应用,实现了框架的易用性.实验证明 JITNREv 框架能够在资源限制条件下稳定提高算法性能,例如,针对 GCN 算法的节点分类任务相比默认参数设置,JITNREv 框架能够将性能提升 31%.

关键词 网络表示学习;网络嵌入;图卷积网络;自动化机器学习;超级参数调整

中图法分类号 TP18 **DOI 号** 10.11897/SP.J.1016.2022.00897

A Framework for Fair Comparison of Network Representation Learning Algorithm Performance Based on Hyperparameter Tuning

GUO Meng-Ying[1),2)] SUN Zhen-Yu[1),2)] ZHU Yu-Qing[3),4)] BAO Yun-Gang[1),2)]

[1)](*Center for Advanced Computer Systems, Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100190)

[2)](*University of Chinese Academy of Sciences, Beijing* 100049)

[3)](*Beijing National Research Center of Information Science and Technology (Tsinghua University), Beijing* 100084)

[4)](*National Engineering Laboratory of Big Data System Software, Beijing* 100084)

Abstract Network data are ubiquitous in real-world applications to represent complex relationships of objects, e.g., social networks, reference networks, and web networks, etc. However, due to

收稿日期:2021-04-29;在线发布日期:2021-11-24.本课题得到国家重点研发计划(2016YFB1000201)、国家自然科学基金项目(61420106013)资助.**郭梦影**,博士研究生,中国计算机学会(CCF)学生会员,主要研究方向为网络表示学习、超级参数调整.E-mail:guomengying@ict.ac.cn.**孙振宇**,博士研究生,中国计算机学会(CCF)学生会员,主要研究方向为网络表示学习、组合算法、计算复杂性.**朱妤晴**(通信作者),博士,助理研究员,中国计算机学会(CCF)会员,主要研究方向为分布式系统、数据管理和系统性能测试与优化.E-mail:zhuyuqing@tsinghua.edu.cn.**包云岗**,博士,教授,博士生导师,中国计算机学会(CCF)会员,主要研究领域为计算机体系结构、操作系统、系统性能建模与评估.

the large-scale and high-dimensional sparse representation of network datasets, it is hard to directly apply off-the-shelve machine learning methods for analysis. Network representation learning (NRL) can generate succinct node representations for large-scale networks, and serve as a bridge between machine learning methods and network data. It has attracted great research interests from both academia and industry. Despite the wide adoption of NRL algorithms, the setting of their hyperparameters remains an impacting factor to the success of their applications, as hyperparameters can influence the algorithms' performance results to a great extent. How to generate a task-aware set of hyperparameters for different NRL algorithms in order to obtain their best performance, achieve their performance fair comparison, and select the most suitable NRL algorithm to analyze the network data are fundamental questions to be answered before the application of NRL algorithms. In addition, hyperparameters tuning is a time-consuming task, and the massive scale of network datasets has further complicated the problem by incurring a high memory footprint. So, how to tune NRL algorithms' hyperparameters within given resource constraints such as the time constraint or the memory limit is also a problem. Regarding the above two problems, in this work, we propose an easy-to-use framework named JITNREv, to compare NRL algorithms fairly within resource constraints based on hyperparameters tuning. The framework has four loosely coupled components and adopts a sample-test-optimize process in a closed loop. The four main components are named hyperparameter sampler, NRL algorithm manipulator, performance evaluator, and hyperparameter sampling space optimizer. All components interact with each other through data flow. We use the divide-and-diverge sampling method based on Latin Hypercube Sampling to sample a set of hyperparameters, and trim the sample space around the previous best configuration according to the assumption that "around the point with the best performance in the sample set we will be more likely to find other points with similar or better performance". Massive scale of network data also brings great challenges to hyperparameter tuning, since the computational cost of NRL algorithms increases in proportion to the network scale. So we use the graph coarsening model to reduce data size and preserve graph structural information. Therefore, JITNREv can easily meet the resource constraints set by users. Besides, the framework also integrates representative algorithms, general evaluation datasets, commonly used evaluation metrics, and data analysis applications for easy use of the framework. Extensive experiments demonstrate that JITNREv can stably improve the performance of general NRL algorithms only by hyperparameter tuning, thus enabling the fair comparisons of NRL algorithms at their best performances. As an example, for the node classification task of the GCN algorithm, JITNREv can increase the accuracy by up to 31% compared with the default hyperparameter settings.

Keywords network representation learning; network embedding; graph convolutional network; automated machine learning; hyperparameter tuning

1 引 言

网络结构数据在现今生活中普遍存在且被广泛应用,例如社交网络[1]、学术文章的引用网络[2-3]、论文作者之间的合作网络[4]或网页之间所构成的跳转网络[5]等.对网络结构数据进行分析在数据分析领域也有重要应用,例如链接预测[6]、节点分类[7]等.但是由于网络结构数据通常具有规模较大、结构稀疏等特性,无法直接应用已成熟的数据分析方法进行分析,因此近几年有大量的网络表示学习算法提出[6-9].这些算法通过基于矩阵分解、基于随机游走或基于神经网络等方式将高维的网络结构数据映射到低维向量空间中,获取网络中节点(例如学术文章

引用网络中的作者)或整个小规模网络(例如蛋白质分子结构)的低维表示,并将学习得到的表示与下游数据分析应用相结合,对网络结构数据进行了有效分析.但是网络表示学习算法中超级参数普遍存在且数量众多,不同的参数配置对算法的性能有巨大影响,且参数的配置与输入数据集、下游数据分析应用所侧重的数据特征都存在相关性.当使用专家知识对网络表示学习算法进行参数调整时需要较高的人力成本,难以被广泛应用,而当不同算法都使用各自的默认参数执行相同的数据分析任务时,由于不是在算法各自的最优性能条件下进行的比较,比较的公平性难以得到保证.因此如何构建一个基于超级参数调整的框架,通用地对网络表示学习算法的性能进行调整,获取不同算法针对相同任务的最优性能值,从而进行算法之间的性能公平比较是一个亟待解决的问题.

近期对机器学习算法或大数据处理系统的超级参数或配置参数进行调整,提升他们的性能从而实现不同算法或系统之间性能的公平比较,在工业界和学术界受到了广泛的关注[10].在机器学习领域,自动化机器学习的兴起能够仅通过对算法中的超级参数进行调整获得稳定且更优的性能,在图片[11]和文本[12]数据分析领域都有很好的应用效果.在系统领域,通过自动优化框架对大规模数据库系统或数据处理系统的配置参数进行调整以获得更优的系统性能也受到了广泛关注[13-14].

但是现有对超级参数或配置参数进行自动调整的框架都无法直接应用到网络表示学习算法的超级参数调整中,主要有以下 3 个原因:(1)在自动机器学习领域,现有的超级参数调整框架主要是针对文本[12]或图片[11]结构数据,且无法设定在进行超级参数调整时所允许使用的最大资源[15-18],直接将已有的框架应用到网络表示学习算法的超级参数调整中通常需要较长的运行时间和较多的资源消耗(例如内存占用等);(2)现有的配置参数调整框架主要是针对大规模数据处理系统或数据库[12,19-20],即使框架的结构为松耦合,将现有的网络表示学习算法、常用的数据集和数据分析应用及相应的评测指标与已有的系统配置参数调整框架进行融合也需要较高人力成本和较长时间;(3)现有框架使用的参数调整方法通常为基于时序模型(例如贝叶斯优化)[21]或基于搜索(例如随机搜索或网格搜索)[22-23],由于参数和性能间的复杂关系,基于时序模型的方法通常无法对网络表示学习算法的性能函数进行准确拟合,而基于搜索的方法无法对获得的参数与性能之间的关系信息(例如,通常网络表示学习算法的性能与节点表示向量维度规模正相关)进行充分利用,因此需要对较大的参数空间进行搜索,无法在有资源限制(例如时间)的条件下,有效对不同网络表示学习算法的超级参数进行调整.

最新的研究工作[24]也注意到了超级参数对网络表示学习算法性能的影响,提出了面向静态网络节点表示学习算法的超级参数调整框架 AutoNE.该框架在子图上进行超级参数调整,并通过高斯过程将子图上的最优参数配置应用到原图上,解决了网络表示学习算法的超级参数调整问题.但是该框架仍存在以下 3 个主要问题:(1)AutoNE 无法在有资源限制的条件下完成超级参数调整过程,即仍旧需要较长的时间收敛或在多个子图上采集设定数目的样本,引入了额外参数信息,例如生成子图的数目,采集样本的数目等;(2)AutoNE 通过随机游走的方式生成多个子图,破坏了原始数据的整体结构,且只能对数值类型的超级参数进行处理,而在网络表示学习算法中非数值类型的超级参数也普遍存在[7],因此 AutoNE 的通用性无法得到保证;(3)AutoNE 在子图上采用随机搜索的方式获取多组超级参数配置和对应数据分析应用的性能值,并没有充分利用参数配置和算法性能之间的关联信息.此外 AutoNE 使用了基于贝叶斯优化的方式对网络表示学习算法的性能和配置参数之间的关系进行建模,而性能模型对初始阶段超级参数采样的样本及所对应的算法性能十分敏感,当初始样本出现波动时,模型也会发生较大变化,因此所构建性能模型的准确性和稳定性也无法得到保证,在此基础上进行超级参数调整获得的最优性能值也并不稳定,公平比较问题依旧无法很好解决.且 AutoNE 并没有将常用的网络表示学习算法、数据集、下游数据分析应用及评测指标融合形成完整工具,因此快速对不同网络表示学习算法进行超级参数调整所要求的易用性也无法得到满足.

为了在有资源限制的条件下解决基于超级参数调整的网络表示学习算法性能公平比较问题,我们提出了 JITNREv(Justice In Time Network Representation Evaluation)框架.框架整体为闭环结构,由超级参数采样、网络表示学习算法、应用性能评估和超级参数采样范围优化四个主要模块构成.为解决资源限制条件下的基于超级参数调整的网络表示学习算法性能公平比较问题,我们面临以下 2 个主

要挑战：

(1)通用性和易用性. 框架的通用性指能够对各类网络表示学习算法进行超级参数调整，且支持对各种类型超级参数(数值型、枚举型等)的调整. 框架的易用性指易于与现有的网络表示学习算法进行融合，而不需要进行复杂的编码工作. 因此 JITNREv 框架使用了闭环的模块化整体结构和基于拉丁超立方采样(LHS)的采样方法，框架中的不同模块之间仅通过信息流的方式进行信息传递，通过保留的接口能够快速与现有的各类网络表示学习算法进行融合，实现不同算法性能之间的公平比较.

(2)资源限制. 资源限制指例如时间限制，内存限制等条件. JITNREv 在超级参数处理部分使用了基于 LHS 的采样方法，能够在有资源限制的条件下收敛到当前最优；对于超大规模的数据集，JITNREv 使用了基于多层布局算法(Multilevel Layout Algorithm)的图粗化(Graph Coarsening)模型对数据进行压缩，在保留数据结构信息的基础上减小数据规模，对于超大规模数据集进一步缩短了进行超级参数调整所需要的时间.

因此本文的主要贡献可以总结为以下 3 方面：

(1)我们提出了如何在有资源限制条件下对多种网络表示学习算法进行公平性能比较的问题，并提出了简单且有效的基于超级参数优化的框架 JITNREv.

(2)JITNREv 框架通过闭环结构，全自动且通用的对各类网络表示学习算法和各种超级参数类型进行调整，使用了基于拉丁超立方采样(LHS)的超级参数处理方式，能够保证在有资源限制的条件下收敛到当前最优，并且针对超大规模数据集通过图粗化方法对数据进行压缩，能够进一步缩短进行超级参数调整所需要的时间且降低运行时的内存需求.

(3)JITNREv 框架将 3 类网络表示学习算法基本实现的代表性算法、6 类网络表示学习算法的常用数据集、2 类常用的数据分析应用和 3 类应用性能评测指标结合到框架中保证了框架的通用性和易用性. 且通过实验展现了 JITNREv 优于基准方法的性能.

本文第 2 节对文章的相关工作进行介绍；第 3 节对所研究的问题进行公式化定义；第 4 节对 JITNREv 的具体实现方式进行介绍；第 5 节对框架的实验结果进行展示；最后我们将在第 6 节对工作进行总结.

2 相关工作

本文提出了基于超级参数调整的网络表示学习算法性能公平比较框架 JITNREv，即能够通用地对多种网络表示学习算法进行超级参数调整以获取不同算法的最优性能，达到对算法性能进行公平比较的目的. 框架涉及的主要内容包括网络表示学习算法和超级参数调整两部分，因此我们将从这两部分对 JITNREv 框架的相关工作进行介绍.

2.1 网络表示学习算法

由于网络结构数据的规模较大，通常无法直接使用现有的机器学习算法对数据进行分析，因此如何找到网络结构数据的合适表示是现在热门的研究方向. 网络表示学习[25-27]在过去的几年中在学术界和工业界[28]都引起了广泛关注和应用，根据所处理的网络结构数据状态，可以粗略的分为针对动态网络的表示学习[29-33]和针对静态网络的表示学习[34-40]这 2 类，根据网络表示学习算法的最终输出，又可以粗略的分为针对全图的表示学习[41-44]、针对子图的表示学习[45-48]、针对节点的表示学习[6-9]和针对边的表示学习[49-50]这 4 类，具体分类可参考文献[51].

我们与 AutoNE 相同以对静态网络中的节点进行表示学习的算法为例，对 JITNREv 框架的通用性进行介绍. 根据算法中所使用的基础模型，这些算法可以被粗略的分为基于矩阵分解，基于随机游走和基于神经网络这 3 大类：

(1)基于矩阵分解的方法. 这类方法通常使用输入数据的邻接矩阵构建对应网络的近似矩阵[34]并通过矩阵分解的方式进行节点表示学习. 由于奇异值分解(SVD)拥有低秩近似的特性，通常基于矩阵分解的算法都使用 SVD 对近似矩阵进行分解[35]；除了使用 SVD 进行分解外，为了便于保留数据的社区性质(Community Structure)，M-NMF[36]也使用非负矩阵分解对近似矩阵进行分解，获取网络中节点的表示.

(2)基于随机游走的方法. 网络中点的邻接向量对这个点的一阶邻居进行了编码，即保存了与这个点直接相连的邻居节点信息，但是除了一阶邻居外，每个点在网络中的局部结构特征在对这个点进行表示学习时也有十分重要的作用，所以产生了基于随机游走的网络表示学习算法，通过随机游走的方式对点的一阶相似性和二阶相似性[6]进行保留. 例如代表工作 Deepwalk[8]主要是受到了自然语言

处理领域 Word2vec[52] 方式的启发，将网络中的节点类比为自然语言处理中的单词，通过随机游走生成的路径作为句子进行节点表示学习. Node2vec[38] 对 Deepwalk 做了进一步改进，通过使用有偏差的随机游走(Biased Random Walk)保留节点间的更多种连接方式. LINE[6] 也是基于 Word2vec 实现的网络表示学习算法，与 Deepwalk 的区别在于构建句子时通过边采样的方式代替了随机游走采样.

(3) 基于神经网络的方法. 网络表示学习算法的核心就是将高维空间中的网络结构数据映射到低维向量空间，而神经网络作为在其他领域获得很大成功的非线性学习方法，在网络表示学习领域也有广泛应用. 其中的代表方法有 SDNE[9]、SDAE[39]、SiNE[40]、TriDNR[53] 等. 除了上述基于传统神经网络的节点表示学习算法外，GCN[7]、GraphSage[37]、GAT[54]、Mixhop[55] 等通过卷积神经网络对包含了节点属性信息的网络结构数据，针对节点分类应用进行端到端学习的方法也受到了越来越多的关注.

多种网络表示学习算法分别对网络结构数据的不同属性进行了保留，但是这样也引入了大量的超级参数. 针对不同的数据分析应用，不同的数据属性所发挥的作用不同；针对不同的数据集，不同数据属性的重要性也不相同，因此导致对算法中超级参数的选择也不同[38]. 此外，对不同的网络表示学习算法，相同参数的不同取值对整个算法的性能和运行效率的影响也不同[56]. 而目前并没有专门针对网络表示学习算法且支持多种参数类型超级参数调整框架，无法通用的对网络表示学习算法性能进行超级参数调整，无法针对相同应用获取不同算法的最优性能，实现算法间性能的公平比较，从而选择出算法中的最优者对数据进行分析，因此我们提出了针对静态网络节点表示学习算法的性能公平比较框架 JITNREv.

2.2　超级参数调整

随着大数据分析系统和机器学习算法的日趋复杂，在工业界和学术界人们都逐渐开始重视参数对整个系统或算法性能的影响. 在大数据处理系统领域有许多工作通过对系统中的配置参数进行调整优化系统的性能，例如 BestConfig[13] 提出了统一的配置参数调整框架，对不同系统使用搜索方式快速进行参数调整，减少复杂系统对专家的依赖，以达到在相同运行环境下仅通过参数的调整获得更好系统性能的目的. OtterTune[19] 也通过对复杂系统的配置参数进行调整达到获取更好系统性能的目的，但是使用了基于贝叶斯优化的模型进行，CDBTune[20] 则通过强化学习的方式达到目的，与基于搜索和贝叶斯优化的方式相比需要花费更多的时间.

在机器学习领域，自动化机器学习[31] 中的超级参数优化部分也是使用机器代替专家，对机器学习算法中的超级参数进行调整，其中的代表工作有 Autotune[15]、Auto-Keras[16] 和 Optuna[17] 等，值得注意的是，最新工作 HOpt[18] 与我们工作相似，也使用了基于拉丁超立方采样的方式对超级参数进行处理. 但是这些方法构建的超级参数调整框架并不能直接应用于网络表示学习算法中(例如还需要融合常用的数据集、评测指标、评测应用等)，且没有针对网络结构数据进行额外的优化(例如为了缩短对大规模网络结构数据进行参数调整的运行时间，而进行的特定优化等).

近期，人们逐渐对网络表示学习算法中超级参数对性能造成的影响产生了重视. 文献[56]对基于随机游走的网络表示学习算法提出了结合注意力模型，通过向后传递的方式训练注意力参数，以代替固定超级参数(例如随机游走生成序列的长度)，引导随机游走在采样时根据数据集的不同更关注长距离或短距离依赖，使相同的算法可以根据数据集的特征(例如数据集的稀疏性等性质)使用不同的超级参数获得更好的性能. 除了专门针对基于随机游走网络表示学习算法进行的改进外，最新研究工作 AutoNE[21] 将自动机器学习应用到了静态网络节点表示学习算法的超级参数调整中并构建了统一框架. 他不仅能够对不同算法的超级参数进行调整，还针对网络结构数据进行了额外优化. 通过随机游走的方式对原图进行采样生成子图(Subgraphs)，在子图上进行超级参数调整，收集超级参数集合和网络表示学习算法在对应子图上的性能值，通过高斯过程构建超级参数集合、子图与原图特征相似性和网络表示学习算法性能之间的函数关系，选择可能在原始网络结构数据上有最好性能的多个超级参数配置集合运行，最终输出最优性能值和对应的超级参数集合. 这个方法的缺点在于：(1) AutoNE 无法在有资源限制的条件下完成超级参数调整过程，即仍旧需要较长的时间收敛或在多个子图上采集设定数目的样本，需要额外信息作为输入，例如生成子图的数目，且生成的子图数目通常与网络中节点的类别数相关，因此需要已知数据中节点的类别信息；(2) AutoNE 采用随机游走的方式生成子图，破坏了原始数据的整体结构，且使用转换学习的方式将子

图上最适配的超级参数直接应用到原图上，因此只能对数值类型的超级参数进行处理，而在网络表示学习算法中非数值类型的超级参数也普遍存在[7]；(3) AutoNE 在子图上采用随机搜索的方式获取多组超级参数配置和对应数据分析应用的性能值，并没有充分利用参数配置和算法性能之间的关联信息. 此外由于网络表示学习算法性能与算法超级参数集合、待分析数据集数据特征之间的复杂关系，通过构建性能模型的方式无法准确对网络表示学习算法的性能进行预测从而调整算法的超级参数配置.

上述相关工作中的所有超级参数调整模块，可以根据所使用模型的不同，被粗略分为使用搜索的方式和通过构建模型的方式进行调整这两大类. 随机搜索和网格搜索[22]都是使用搜索的方式进行超级参数调整的代表，其中 AutoNE 中对子图进行超级参数调整就使用了随机搜索方式，自动化机器学习中的 HOpt 框架也基于搜索方式进行. 这类方法直接在参数空间中进行最优参数配置的搜索，但是主要存在以下 2 个缺点：(1) 在超级参数搜索时没有使用过去的经验信息；(2) 对于高维的超级参数集合会产生组合爆炸问题. 通过构建模型的方式进行超级参数调整使用了基于序列的优化模型[23]，相较于单纯的搜索方法该模型利用过去经验信息减少了运行次数，其中的代表工作是贝叶斯优化[21]. 贝叶斯优化假设多个超级参数之间符合联合高斯分布，通过高斯过程对超级参数和性能之间的关系进行建模，根据过去经验信息，以最大化期望性能提高的概率，选择下一次运行的超级参数配置，现有的许多参数调整框架都在其中使用了贝叶斯优化作为标准的参数调整算法，例如自动化机器学习中的 Auto-Keras 框架和 AutoNE 对原始数据进行超级参数调整部分. 但是通过构建模型的方式进行调整也存在以下 2 个明显缺点：(1) 超级参数之间的相互影响十分复杂，构建性能模型的准确率无法保证；(2) 在构建模型的冷启动阶段需要采集初始样本，而这些初始样本对最终构建得到性能模型的准确率有十分显著的影响.

3 问题定义

本文针对网络表示学习算法的性能公平比较问题，提出了基于超级参数调整的框架 JITNREv. 在这一节中，我们首先对文章中所使用的基本概念和符号进行定义，再将研究问题进行公式化描述.

3.1 基本定义

定义 1. 网络结构数据. 作为图结构数据的一种，表示为 $G=(V,E,\boldsymbol{A}_V,\boldsymbol{A}_E,L)$，其中 V 表示网络中节点的集合，$|V|$ 表示 G 中节点的数目. $E\subseteq(V\times V)$ 表示网络中边的集合，$\boldsymbol{A}_V\in\mathbb{R}^{|V|\times m}$ 表示 G 中点的属性矩阵，其中 m 表示点属性的维度. $\boldsymbol{A}_E\in\mathbb{R}^{|E|\times n}$ 表示 G 中边的属性矩阵，其中 n 表示边属性的维度. 当 G 为有权图时，$v_i,v_j\in V$，$e_{ij}\in E$，边的权值 w_{ij} 属于 $A_{e_{ij}}$ 中的元素，即 $w_{ij}\in A_{e_{ij}}$，$A_{e_{ij}}\in\boldsymbol{A}_E$，且 $w_{ij}\in[0,1]$. $L\in\mathbb{R}^{|V|\times|\gamma|}$ 表示网络中点的标签，其中 γ 表示 G 中节点的标签集合. 由于网络结构数据的多样性，在有些数据集中，点属性、边属性或标签不一定存在，分别用 $\boldsymbol{A}_E=\text{Null}$，$\boldsymbol{A}_V=\text{Null}$ 或 $L=\text{Null}$ 表示.

定义 2. 网络表示学习. 是对网络结构数据 $G=(V,E,\boldsymbol{A}_V,\boldsymbol{A}_E,L)$ 找到一个映射函数 M：$v_i\mapsto x_{v_i}\in\mathbb{R}^d$，其中 x_{v_i} 是点 v_i 通过学习得到的低维表示，d 为低维向量的维度，因此对于任意网络表示学习方法，可以表示为 $X=M_\theta(G)$，其中 θ 为映射函数 M 需要的超级参数，例如 $d\in\theta$，X 表示网络表示学习的结果.

定义 3. 性能. 使用 $P(M_\theta,G,A)$ 表示性能函数，即网络表示学习算法 M 的最终性能由 M 所使用的超级参数配置 θ，网络结构数据 G，下游数据分析应用 A 共同决定，其中 A 可以为节点分类或链接预测等具体应用.

3.2 问题描述

通常网络表示学习算法 M 的性能函数 $P(\cdot,\cdot,\cdot)$ 未知. 以基于随机游走的网络表示学习算法 Deepwalk 为例，我们使用 Arxiv[4] 数据集统计了随机游走序列长度和每个点生成的随机游走序列数目与链接预测应用性能之间的关系，得到性能曲面如图 1 所示. 从图 1 中可以看出，相同算法在不同参数配置条件

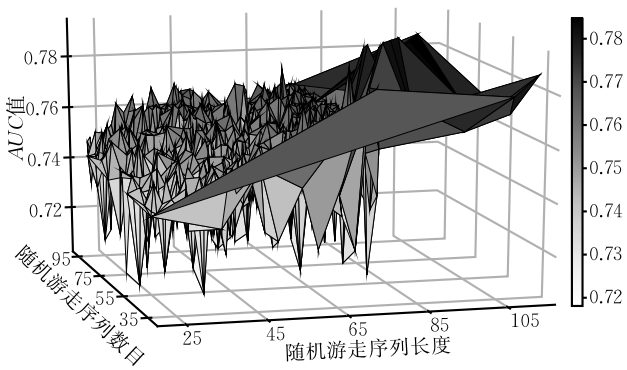


图 1 网络表示学习算法 Deepwalk 参数性能曲面

下使用相同数据集针对相同应用性能有明显差别，因此我们提出了如何对不同网络表示学习算法的性能进行公平比较的问题.

我们的目标是对于任意网络表示学习算法 M，针对其中的超级参数 θ，在提供的超级参数取值范围 dis_θ 和超级参数取值限制条件 lim_θ 下，在给定资源限制条件 N（例如最多允许的网络表示学习算法运行次数 N_t 或可使用的最大内存 N_m）下，最大化 $P(M_\theta,G,A)$，最终输出 $P(M_\theta,G,A)$ 的最优参数配置 $\theta_{M_{\text{best}}}$，最高性能值 $P_{M_{\text{best}}}$ 和对应的节点表示 $X_{M_{\text{best}}}$. 通过对不同 M 的 $P_{M_{\text{best}}}$ 进行比较实现不同算法之间的性能公平比较.

4　JITNREv 框架

为解决 3.2 节中所描述的网络表示学习算法性能公平比较问题，我们提出了基于超级参数优化的网络表示学习算法性能公平比较框架 JITNREv（如图 2 所示）. 在这一节中，我们首先对 JITNREv 框架的整体结构进行介绍，再对框架中主要涉及的超级参数处理部分和网络表示学习算法及性能评估这两部分内容分别进行介绍，之后根据网络结构数据通常规模较大的特点，我们对超大规模网络结构数据集的优化模块图粗化进行介绍，在这一节的最后我们对 JITNREv 框架整体的时间复杂度进行了分析.

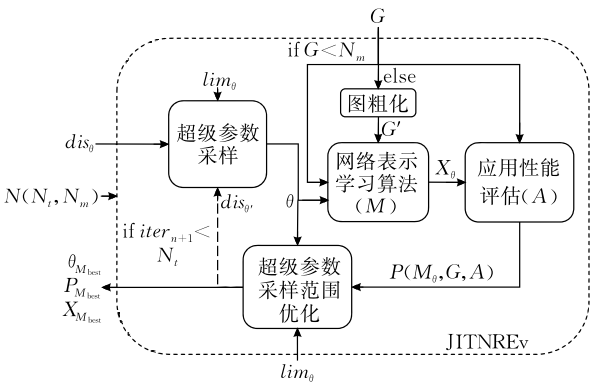


图 2　JITNREv 框架整体结构

4.1　整体设计

JITNREv 框架整体为闭环结构，可以自动在资源允许的条件下多次运行，找到使算法性能最优的超级参数配置集合. 框架输入信息中的资源限制条件 N 可以是最多允许算法运行的次数 N_t 和最大允许算法占用的内存空间 N_m 等，其他输入包括使用的网络结构数据集 G，需要调整的超级参数集合 θ，及参数的采样范围上下界（表示为 dis_θ）. 其中 lim_θ 表示超级参数采样时的限制条件（非必要输入），通常在有 lim_θ 的条件下更容易找到更优的配置（例如 Deepwalk 算法在节点的低维表示向量长度为 4 的倍数时，通常会有更好的性能[8]）. 整个框架的输出为在资源限制条件 N 下的网络表示学习算法 M，针对网络结构数据集 G 的最优参数配置集合 $\theta_{M_{\text{best}}}$，最优性能值 $P_{M_{\text{best}}}$ 和对应的节点表示结果 $X_{M_{\text{best}}}$.

JITNREv 以信息流的方式将框架中的不同模块以松耦合方式进行连接，便于框架的扩展和集成. 整个框架主要由超级参数采样，网络表示学习算法，应用性能评估和超级参数采样范围优化这 4 部分组成. 超级参数采样是在给出的资源限制条件和参数上下界范围内对需要调整的超级参数进行采样；网络表示学习算法模块是以采样得到的超级参数作为输入，运行需要调整的网络表示学习算法，生成节点的表示学习结果；应用性能评估模块是以常用的表示学习下游应用和评测指标对学习得到的结果进行衡量；超级参数采样范围优化模块是以现有的超级参数采样结果和对节点表示学习结果进行评估得到的性能值作为输入，通过剪枝的方式缩小超级参数的采样范围，在资源允许的条件下重新进行采样. 在数据集规模超过所允许使用的最大内存 N_m 时，JITNREv 框架引入了图粗化模块对超大规模数据集进行预处理. 在保留输入数据结构信息的基础上压缩数据规模，减少内存占用，缩短进行超级参数调整所需的运行时间. 整个框架通过闭环的方式多次自动运行，能够针对相同数据分析任务在资源限制条件下找到不同网络表示学习算法的最优性能，以实现网络表示学习算法之间性能的公平比较.

4.2　超级参数处理

整个超级参数处理部分被分为了超级参数采样和采样范围优化这两个模块. 在获取超级参数的采样范围 dis_θ 以及资源限制条件 N 后，超级参数采样模块需要生成一系列初始采样样本. 在对超级参数进行采样时，我们需要满足以下两个条件：(1) 对网络表示学习算法中不同类型的超级参数都适用，由于网络表示学习中的超级参数众多，不仅包括整型、浮点型参数，还有枚举、字符串等类型的参数，而现有的面向网络表示学习算法的超级参数调整框架 AutoNE[24] 并不能支持对字符串或枚举类型参数的调整，因此为了保证框架的通用性 JITNREv 的超级参数采样模块需要支持各类超级参数；(2) 能够

在有限样本集中收敛，由于 JITNREv 框架限定了在进行超级参数调整时使用的资源 N 即能够运行网络表示学习算法的次数等，因此需要超级参数采样和采样范围优化模块能够在有限的运行次数中收敛到当前最优.

基于超级参数调整的网络表示学习算法性能公平比较问题本质是一个以超级参数采样范围作为输入的面向性能目标的最优化问题. 对于性能的最优化问题可以通过专家知识进行建模实现白盒优化，但是这样的建模过程耗时很长且代价巨大，随着网络表示学习算法更新的日益加速，这种方式难以得到广泛应用. 因此，我们将网络表示学习算法的性能优化建模为黑盒优化问题. 黑盒优化的解决方法通常分为两大类：一类是基于时序模型的优化方法，例如贝叶斯优化；另一类是基于搜索的方法，例如随机采样或网格采样等，这类方法可以克服贝叶斯优化在冷启动阶段收集样本的偏差对性能模型构建造成的影响，但是并没有对之前的样本信息进行充分利用. 因此我们借鉴了文献[13]中提出的方法，解决面向网络表示学习算法的超级参数优化问题.

我们以拉丁超立方采样(LHS)作为 JITNREv 超级参数采样模块的基础方法. 具体实现方式是：为了保证能够在有限时间内收敛，我们仿照网格采样将算法中需要被调整的 n 个超级参数中的每一个都根据各自采样范围 dis_{θ_i}，$\forall i\in[1,n]$ 划分为 k 个间隔，在每一个间隔中采集一次样本，其中 k 值可以根据 N_t 的限制进行设置. 但是这种类似网格采样的方法在保证了收敛性的同时，会随着需要被调整的超级参数数目的增加，造成组合爆炸问题. 为了解决这个问题，我们观察到算法的性能通常由更有影响力的超级参数决定. 如图 1 中我们观察到“随机游走序列长度”对 Deepwalk 算法的性能有更明显影响. 因此，我们不需要对有相同“随机游走序列长度”值和不同“随机游走序列数目”的组合再进行测试. 所以在将 n 个超级参数都划分为 k 个间隔后，我们只需要采集 k 个样本进行测试，即可以表示为对于测试所用的样本集合 S_θ：

$$S_\theta=\{S_1,S_2,\cdots,S_k\} \tag{1}$$

$$S_{pi}\neq S_{qi},\ \forall p,q\in[1,k],\ \forall i\in[1,n] \tag{2}$$

例如图 3 中的超级参数 X 和 Y，我们分别将他们划分为 6 个间隔后，我们只需要按照图示随机采集 6 组样本进行测试，X 和 Y 的每一个小间隔，都只会有一组样本进行覆盖，极大减少了需要采样的样本数目和测试所需的时间.

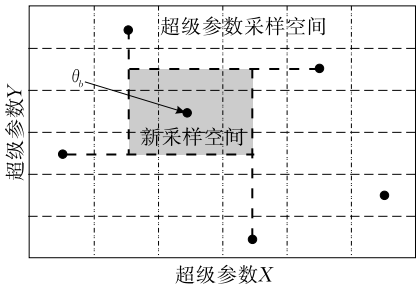


图 3 JITNREv 中超级参数采样方法[13]

对于需要不同类型超级参数都适用的需求，JITNREv 通过将这些变量转化到连续的数值空间中再进行切分采样解决. 我们以 GCN 中可以选择的超级参数‘模型’为例，GCN 在运行中可以选择三种模型类型，即

$$Model\in\{\text{‘}gcn\text{’},\text{‘}gcn\_chebyshev\text{’},\text{‘}dense\text{’}\} \tag{3}$$

我们以 mdl 作为指示变量，设置其取值范围为 $[0,3)$，当取值为 $[0,1)$ 时对应‘gcn’，在取值为 $[1,2)$ 时对应‘$gcn\_chebyshev$’，取值为 $[2,3)$ 时为‘$dense$’，即

$$Model=\begin{cases}\text{‘}gcn\text{’}, & mdl\in[0,1)\\ \text{‘}gcn\_chebyshev\text{’}, & mdl\in[1,2)\\ \text{‘}dense\text{’}, & mdl\in[2,3)\end{cases} \tag{4}$$

再使用上述方法进行采样，这样的映射方式同样可以应用到字符串或布尔类型的变量中以解决超级参数类型多样性的问题.

在超级参数采样范围优化模块中我们观察图 1 中网络表示学习算法的性能曲面，作出“在当前最优的超级参数配置附近，有性能更优的超级参数配置的概率更大”的假设. 因此我们使用如图 3 所示的方法进行剪枝.

既在初始超级参数采样的样本集合 S_θ 中，θ_b 拥有最好的性能值，则我们认为在‘新采样空间’内可能拥有更好性能的超级参数配置集合的概率更大，且更优性能的超级参数配置和已有的超级参数配置集合没有交集. 如图 3 所示，对于每一个需要调整的超级参数 θ_i，我们首先找到上一轮采集的所有样本中比 θ_{bi}（表示上一轮的最优超参配置集合 θ_b 中超级参数 θ_i 的取值）小的最大值，记为

$$\theta_i^h=\max\{S_{pi}\mid S_{pi}<\theta_{bi},p\in[1,k]\} \tag{5}$$

k 为上一轮采集的样本数，同时确定比 θ_{bi} 大的最小值 θ_i^l，记为

$$\theta_i^l=\min\{S_{pi}\mid S_{pi}>\theta_{bi},p\in[1,k]\} \tag{6}$$

这样在采样空间 dis_θ 中，n 个维度的每一个维度 i 都由一个超级参数 θ_i 确定了新的上下界 (θ_i^h,θ_i^l)，即

$$\theta_i \in [\theta_i^l, \theta_i^h],\ \forall i \in [1,n] \tag{7}$$

因此新一轮采样中超级参数样本空间表示为

$$\Omega = [\theta_1^l, \theta_1^h] \times [\theta_2^l, \theta_2^h] \times \cdots \times [\theta_n^l, \theta_n^h] \tag{8}$$

这样在迭代次数 $iter_{n+1} < N_t$ 的条件下，我们就得到了新的超级参数采样范围 $dis_{\theta'} = \Omega$，并将新的采样范围 $dis_{\theta'}$ 输入超级参数采样模块中，进行下一轮迭代.

上述假设和采样范围优化方式对离散型超级参数同样适用. 仍以 GCN 算法为例，假设运行初始样本集合 S_θ 后可知，‘*dense*’模型的性能最优，根据上述假设和采样范围优化的方式，*mdl* 变量新一轮的采样范围大概率仍在[2,3)内，因此根据式(4)可知‘*Model*’参数的取值仍为‘*dense*’，解决了离散型超级参数采样及采样范围优化问题.

整个框架的核心代码如算法 1 所示. 当算法收敛或不再满足资源限制条件后，输出当前最优的超级参数配置集 $\theta_{M_{\text{best}}}$，最优的性能值 $P_{M_{\text{best}}}$ 和学习得到的表示结果 $X_{M_{\text{best}}}$. JITNREv 通过设定相同的 N_t，使用相同的数据集 G 和数据分析应用 A，对不同网络表示学习算法 M 的 $P_{M_{\text{best}}}$ 进行比较，实现网络表示学习算法性能的公平比较.

算法 1. JITNREv 框架中的超级参数处理代码.

输入：资源限制条件 N_t；网络结构数据 G；网络表示学习算法 M；需要调整的超级参数集合 θ 和采样范围 dis_θ；超级参数限制条件 lim_θ；初始采集样本数 k

输出：最优参数配置集 $\theta_{M_{\text{best}}}$

1. According to θ, dis_θ and lim_θ, sample k hyperparameter configurations into set S_θ;
2. Run M on G with S_θ and select the best θ_b;
3. $iter_{n+1} = k+1$;
4. WHILE ($iter_{n+1} < N_t$) DO
5. Trim hyperparameter space around θ_b, and generate new sampling space $dis_{\theta'}$
6. $\lfloor k = k/2 \rfloor$
7. Sample k configurations into set $S_{\theta'}$, according to $dis_{\theta'}$ and lim_θ
8. Run M on G with $S_{\theta'}$ and select best θ_b'.
9. $\theta_b = \theta_b'$
10. $iter_{n+1} += k+1$
11. END WHILE;
12. $\theta_{M_{\text{best}}} = \theta_b$
13. RETURN $\theta_{M_{\text{best}}}$

4.3　网络表示学习方法及应用性能评估

由于 JITNREv 是基于超级参数调整的通用网络表示学习算法性能公平比较框架，网络表示学习算法模块的核心就是保证对各类算法都适用，与 AutoNE 相同，我们也以面向静态网络的节点表示学习算法为例. 该模块以超级参数采样模块获得的超级参数集合 S_θ 作为输入，通过命令行调用的方式运行被调整的网络表示学习算法，并得到学习结果 X_θ 传递给应用性能评估模块. 由于这一模块的调用仅通过命令行实现，因此易于与现有的网络表示学习算法进行集成，体现了 JITNREv 框架的通用性.

应用性能评估模块以当前网络表示学习算法 M 在当前参数配置下学习得到的结果 X_θ 作为输入，通过对 X_θ 在下游数据分析应用 A 中的表现进行评估，计算 M 在当前参数配置集合 S_θ 和数据集 G 中的性能 P，为之后的超级参数采样范围调整模块提供信息.

由于网络表示学习算法的性能与数据集、应用和应用的性能评测指标都有密切关系，为了能够通用的对算法进行性能评估，根据静态网络节点表示学习算法的实现方式，可以将现有的大量算法粗略分为基于矩阵分解，基于随机游走和基于神经网络这 3 类实现，我们在这 3 类算法中选择了最具代表性的工作融合进框架中(详细信息见 5.2.1 节)，并保留了易用的接口方便对其他网络表示学习算法进行集成. 同时，我们将网络表示学习算法代表性工作中所使用的数据集、评测应用和评测指标进行了整理(见表 1)，选择了覆盖性广的 6 种数据集、2 种应用和 3 种评测指标集成在 JITNREv 框架中，且在 JITNREv 框架中保留了接口便于更多性能评测应用、指标以及数据集的集成，关于 JITNREv 框架中现有数据集、评测应用及指标的详细信息，见 5.1 和 5.2.2 节.

表 1　静态网络节点表示学习算法中常用数据集、应用及性能评测指标统计

Datasets	Applications		Metric
	Classification	Link Prediction	
Arxiv		√	*AUC*
Blogcatalog	√	√	*AUC*, Micro-*F*1
Wikipedia	√	√	*AUC*, Micro-*F*1
Cora	√		*Accuracy*
Pubmed	√		*Accuracy*
Flickr	√	√	*AUC*, Micro-*F*1

4.4　超大规模数据集的预处理

JITNREv 能够在有资源限制的条件下，对各类网络表示学习算法进行超级参数调整. 其中的资源限制条件除了时间限制即最多允许算法运行的次数 N_t 外，还可以是允许使用的最大内存 N_m. 所以针对超大规模的网络结构数据集，我们提出了使用图粗

化的方式进行预处理，在保留数据结构的基础上缩小数据集规模，限制整个框架在进行超级参数调整时所使用的内存.此外，网络表示学习算法的运行时间通常与数据集的规模正相关，缩小数据集的规模可以进一步缩短进行超级参数调整所需的时间.

在JITNREv中我们采用了基于多层布局(Multi-level Layout Algorithm)的图粗化算法(Graph Coarsening)对大规模网络结构数据进行预处理，在不同的层次上保留数据的结构信息，对图数据整体规模进行压缩的条件下，保留图的全局结构信息.其中不同层次的图数据结构信息主要为节点间的一阶相似性(First-Order Proximity)和二阶相似性(Second-Order Proximity)[6]，保留一阶相似性的粗化是对图中已知的边进行压缩，保留二阶相似性的粗化是对节点的邻居结构进行处理.我们使用了HARP[57]中提出的边融合(Edge Collapsing)和星融合(Star Collapsing)方式对输入数据进行预处理.

边融合[58]是对图数据中的一阶相似性进行保留.它首先在图$G=(V,E)$中选择一个匹配$E_p\subseteq E$，即E_p中的任意两条边不关联同一个节点，之后将$(u_i,v_i)\in E_p$中的点u_i,v_i融合为新的点w_i，并将与u_i,v_i相连的边都迁移到与w_i相连.边融合后的图表示为$G'=(V',E')$，即

$$c(u_i,v_i)=w_i,\ (u_i,v_i)\in E_p \tag{9}$$

$$V'=(V\backslash\{u_1,v_1\}\backslash\{u_2,v_2\}\cdots)\cup\{w_1,w_2,\cdots\} \tag{10}$$

$$E'=E\backslash E_p \tag{11}$$

通过边融合的方式对网络结构数据进行预处理，网络中节点的数目会有一半压缩，大幅度降低了数据的规模.此外，由于G'中完成针对算法的超级参数调整后，我们还会在G中使用选出的最优配置参数运行生成$X_{M_{\text{best}}}$和$P_{M_{\text{best}}}$，所以对E_p进行采集和融合的顺序并不会对整个框架的最终输出结果造成明显影响.

星融合 现实中的网络结构数据通常具有无标度(Scale-free)特性因此存在许多星型结构，即一个中心节点与多个周围节点相连.虽然边融合能够很好的保留一阶相似性，但是对于星型的数据结构无法很好进行结构保留，所以我们使用了与HARP中相同的星融合方式对星型结构进行保留.

星融合的本质是对拥有相同邻居信息的节点进行融合，因此整个融合过程表示为对中心节点v_c，其邻居节点定义为

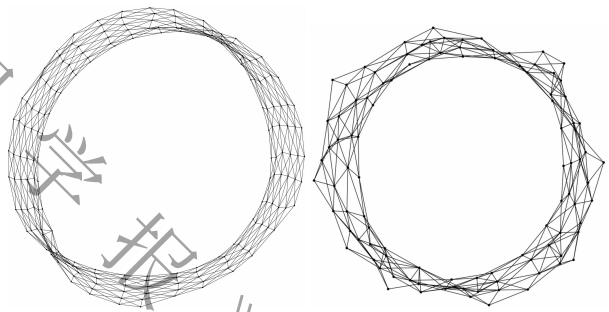
$$\mathcal{N}_{v_c}=\{v_i\,|\,(v_i,v_c)\in E\} \tag{12}$$

则进行星融合后生成的新邻居节点集合$\mathcal{N}'_{v_c}$定义为

$$\mathcal{N}'_{v_c}=\begin{cases}\{c(v_1,v_2),\cdots,c(v_{|\mathcal{N}_{v_c}|-1},v_{|\mathcal{N}_{v_c}|})\}, & |\mathcal{N}_{v_c}|\%2=0\\ \{c(v_1,v_2),\cdots,v_{|\mathcal{N}_{v_c}|}\}, & |\mathcal{N}_{v_c}|\%2=1\end{cases} \tag{13}$$

其中$|\mathcal{N}_{v_c}|$表示在进行星融合之前中心节点v_c的邻居节点数目，由于星融合是对拥有相同邻居的节点进行两两融合与二阶相似性的定义相似，因此星融合能够在粗化过程中很好保留图的二阶结构信息，且由于进行两两融合，生成的G'极大压缩了数据的规模.

因此在对超大规模数据集进行预处理时，我们首先使用星融合的方式对图数据进行粗化，再使用边融合的方式进行进一步压缩，最终生成预处理后的结果G'，我们以can_187数据集[59]为例，对预处理模块的数据结构保留有效性进行展示(见图4).JITNREv框架在预处理后的数据G'上进行超级参数调整可以降低调整过程中的内存占用并缩短网络表示学习算法的运行时间，在获得$\theta_{M_{\text{best}}}$后，再在G上运行一遍算法，生成$\theta_{M_{\text{best}}}$对应的$X_{M_{\text{best}}}$和$P_{M_{\text{best}}}$作为框架的最终输出结果.



(a) can_187原数据 (b) 预处理后结果

图4 原始数据和经过图粗化处理后数据的比较

AutoNE[24]中也有针对超大规模数据集的处理，与我们方法的主要区别在于以下2点：(1)他使用随机游走的方式对原图进行采样生成多个小规模子图，破坏了图的整体数据结构；(2)采样时需要预先设定子图的个数、子图的大小和采样起始点引入了额外信息.我们将在实验部分对AutoNE和JITNREv框架中的超大规模数据集子图处理及运算加速部分进行更详细的比较和说明(见5.3.5节).

4.5 JITNREv框架时间复杂度分析

JITNREv的核心代码在算法1中进行了展示，整个框架的时间复杂度为$O(N_tM)$，其中N_t为允许算法运行的次数，M为被调整网络表示学习算法的时间复杂度，例如GCN算法为$O(|E|)$，Deepwalk算法为$O(|V|\log|V|)$，AROPE算法为$O(|V|)$等.针对超大规模数据集，我们使用了HARP中提出的

图粗化方法对数据集规模进行压缩，算法的时间复杂为 $O(|V|)$，所用的时间不会长于额外运行一次网络表示学习算法的时间，且粗化后所生成的 G' 与原图 G 相比 $|V'| \ll |V|$，$|E'| \ll |E|$，因此在 G' 上进行超级参数调整的时间可以忽略，而最主要的运行时间还是在于将在 G' 上找到的最优超级参数配置集合 $\theta_{M_{\text{best}}}$ 应用到算法 M 中在原图 G 上进行细微调整并输出最终结果的运行时间.

5　实验与结果

我们将在这一节对 JITNREv 框架的性能进行评测. 首先我们对评测所使用的数据集和实验设置进行详细介绍，包括与 JITNREv 框架进行比较的基准算法和与框架进行融合的 3 种代表性静态网络节点表示学习算法. 我们还将对性能评测所使用的数据分析应用和评价指标进行介绍. 在实验部分我们应用 JITNREv 框架和 3 种基准算法分别对 3 种不同类型的网络表示学习算法超级参数进行调整，并展示其性能. 由于 JITNREv 使用图粗化方式对超大规模数据集进行了额外优化，因此我们将 JITNREv 在超大规模数据集上的性能也进行了展示，并与基准算法 AutoNE 中针对超大规模数据集的优化部分从性能稳定性和时间两方面进行了详细比较，展示了 JITNREv 在超大规模数据集节点表示学习上的有效性.

5.1　数据集

我们使用了静态网络节点表示学习算法中常用的 6 种数据集对 JITNREv 框架的性能进行评测，这 6 种数据集包含了有节点属性信息的网络，无节点属性信息的网络 2 大类，又可以根据网络的规模分为中等规模网络和大规模网络 2 大类，数据集的具体信息见表 2.

表 2　网络结构数据集具体信息

Datasets	Category	#Nodes	#Edges	#Labels	w/attr
Blogcatalog[1]	Social	10312	333983	39	F
Wikipedia[60]	Word	4777	184812	40	F
Pubmed[3]	Citation	19717	44338	3	T
Arxiv[4]	Collaboration	5242	28980	—	F
Cora[2]	Citation	2708	5429	7	T
Flickr[1]	Social	80513	11799764	195	F

(1) Blogcatalog[1]. 该数据集是常用的社交网络数据集，其中点表示社交网络中的用户，边表示用户之间的关注关系，根据用户的兴趣设定了 39 个标签对用户进行分类，每个用户至少拥有其中 1 个标签；

(2) Wikipedia[60]. 该数据集中的点表示一个单词，如果在一个滑动窗口中两个单词共同出现，则这两个点之间拥有共现关系存在一条边，这个数据集将点分为 40 类；

(3) Pubmed[3]. 该数据集包含了 Pubmed 数据库中的 19717 篇文章作为点，每一个点都有 500 维的实数向量作为点的属性，文章之间的引用关系作为边，根据文章的内容将文章分为 3 类；

(4) Arxiv[4]. 该数据集为 Arxiv 网站的作者合作关系数据集，数据集中的点表示 Arxiv 网站上的作者，如果两个作者之间有合作则构成一条边；

(5) Cora[2]. 该数据集包含了 Cora 数据库中的 2708 篇文章作为点，每个点都由 0/1 构成的向量作为点属性，文章之间的引用关系作为边，根据文章的内容被分为 7 个不同类别；

(6) Flickr[1]. 根据数据集规模，Flickr 数据集与上述 5 种不同，属于大规模数据集，该数据集反应了 Flickr 网站的用户关系，节点和边分别表示用户和用户之间的关注关系，以用户的兴趣作为标签，每个用户至少拥有 195 个标签中的一个.

5.2　实验设置

本文首次提出了在有资源限制条件下的网络表示学习算法性能公平比较问题，并构建了基于超级参数调整的性能公平比较框架 JITNREv. JITNREv 主要针对面向静态网络的节点表示学习算法，这些算法根据所使用基础模型的不同，可以粗略的被分为基于矩阵分解的网络表示学习算法、基于随机游走的网络表示学习算法和基于神经网络的网络表示学习算法 3 大类. 为了衡量 JITNREv 框架的通用性和有效性，我们将 JITNREv 和这 3 类中的代表性算法相结合进行评测，并使用默认参数、随机搜索和最新相关工作 AutoNE 作为基准，对 JITNREv 框架的性能优越性进行展示.

5.2.1　基准算法

为了验证框架的有效性，我们将算法通过 JITNREv 框架获取的性能与通过 3 种基准算法获取的性能进行比较：

(1) 算法默认超级参数配置下的性能(默认配置). 这组性能是使用算法对应论文中推荐的超级参数配置运行算法获得，针对不同数据集和应用同一算法都采取相同的超级参数配置；

(2) 通过随机搜索调整超级参数(随机搜索)[35]. 随机搜索是现在最常用的针对超级参数进行优化的方法之一. 和 JITNREv 中提出的方法相似，也是

使用基于搜索的方法进行超级参数调整，在假设有充足时间的条件下，随机搜索总能找到最优的参数配置.但是这个方法相较于 JITNREv 中使用的超级参数调整方法的缺点是并没有使用之前运行过程中反馈的信息.根据[35]中的结论，相比网格搜索，由于随机搜索能够在更短时间内更有效的搜索更广泛的采样空间，随机搜索在多数情况下拥有更好的效果.所以我们选择了随机搜索作为基准算法.

(3) AutoNE[24]. AutoNE 是最新提出的专门针对静态网络节点表示学习算法的超级参数调整框架，与我们工作的相关度最高，所以我们也将这个算法列作基准算法进行比较. AutoNE 通过随机游走的方式采样得到多个子图，并在子图上进行超级参数调整，最终根据转化学习的方式，选择与原图最相似子图上的最优超级参数配置在原图上运行，输出最终的结果.与 JITNREv 相比，针对大规模数据集 AutoNE 通过随机游走的方式采样生成多个子图，引入了额外信息(例如需要设置采样的子图数目或子图规模)，且随机游走采样方式破坏了原始数据的整体结构信息；此外 AutoNE 在子图上采用的是传统随机搜索的方式进行超级参数调整，与 JITNREv 中使用的拉丁超立方采样相比对样本的经验信息利用不足，降低了找到算法最优配置参数的概率.

为了验证 JITNREv 框架的通用性，我们从静态网络节点表示学习算法最常用的三类基础方法即基于矩阵分解、基于随机游走和基于神经网络 3 类中，分别选取代表性工作进行超级参数调整，我们将这些算法的被调整参数集合和采样范围进行介绍：

(1)基于矩阵分解的网络表示学习算法 AROPE[34]. 基于矩阵分解的网络表示学习算法通常使用输入数据的邻接矩阵构建对应网络的高阶近似性矩阵(Proximity Matrix)，通过矩阵分解的方式获得每个节点的表示学习结果[61]. 由于 AROPE 能够构建任意阶数的近似性矩阵，我们选择不同阶数近似性矩阵的权重作为被调整的超级参数.分别为一阶权重(w_1)、二阶权重(w_2)和三阶权重(w_3)，取值范围为$[0.0001, 3.0]$，对于默认配置 w_1, w_2, w_3 都取定值 0.1；

(2)基于随机游走的网络表示学习算法 Deepwalk[8]. 基于随机游走的网络表示学习算法主要是受到了自然语言处理领域 Word2vec[52] 算法的启发，将网络中的节点作为单词，随机游走生成的路径作为句子进行节点的表示学习，对网络中节点的一阶相似性和二阶相似性有很好的保留效果.我们使用这类算法中的代表工作 Deepwalk[8] 进行测试，其中对 Deepwalk 算法性能有明显影响的 4 个超级参数分别是生成节点低维表示向量的大小(取值范围为$[40, 256]$)，每个节点进行的随机游走的数目(取值范围为$[40, 100]$)，每一个随机游走序列的长度(取值范围为$[20, 80]$)和进行计算时滑动窗口的大小(取值范围为$[5, 30]$)，对于默认配置 4 个参数都取固定值，即节点低维表示向量大小为 128，在每个节点上进行随机游走的次数为 40，随机游走生成序列长度为 40，滑动窗口的大小为 10；此外 Deepwalk 文章中提出，在节点低维表示向量的大小为 4 的倍数时算法会有更好的性能，因此我们将 JITNREv 中的 lim_θ 设置为只能取 4 的倍数；

(3)基于神经网络的网络表示学习算法 GCN[7]. 网络表示学习的核心就是将高维空间中的网络结构映射到低维的向量空间，作为在其他领域有很多成功应用的非线性学习模型，神经网络在网络表示学习领域受到了越来越多的关注.我们在这类算法中选择了基于卷积神经网络的 GCN[7] 作为代表，他不仅能够对网络的结构信息进行学习，还能够融合节点的属性特征.针对 GCN 算法，我们选择了对他性能有影响的 5 个参数进行调整，分别为训练迭代(Training Epoch)的数目(取值范围为$[10, 300]$)，隐藏层中神经元的数目(取值范围为$[2, 64]$)，学习速率(取值范围为(1e-5, 0.1))，Dropout Rate(取值范围为(0.1, 0.9))和使用 L2 正则化的权值衰退率(取值范围为(1e-5, 1e-3))，默认配置的取值都为固定数值，训练迭代的数目为 200，隐藏层中神经元的数目为 16，学习速率为 1e-3，Dropout Rate 为 0.5，使用 L2 进行正则化的权值衰退率为 5e-4. 因为 JITNREv 与 AutoNE 相比，能够支持对枚举等类型的参数进行调整，所以我们在 Pubmed 数据集中，对 GCN 使用的模型参数也进行了调整，其中模型参数的取值为['*gcn*', '*gcn_chebyshev*', '*dense*'].

在每一组实验中，我们都将展示 JITNREv、默认配置，随机搜索和 AutoNE 四组基准算法的性能.除了上述具体参数设置外，对于 AutoNE 我们与论文中保持一致，设定子图数目为 5，子图规模为 $5\%|V|-20\%|V|$，其中 $|V|$ 为输入数据中的节点数.

JITNREv，随机搜索和 AutoNE 对于中等规模数据集，针对每种应用我们都分别进行 50 次迭代，对于大规模数据集 Flickr 我们则进行 10 次迭代，既对于 JITNREv 我们将资源限制条件设置为对中等规模数据集 $N_t=50$，对于 Flickr 数据集 $N_t=10$，初始采集样本数 $k=\lfloor N_t/2 \rfloor$. 我们将每组实验都在相

同的配置条件下分别运行 5 次，最终将 5 次实验的平均值和方差进行展示.

5.2.2　评测应用和指标

我们与 AutoNE[24] 文章中相同，使用链接预测和节点分类 2 个静态网络节点表示学习算法中常见的下游数据分析应用对 JITNREv 框架的性能进行评测.

对于链接预测应用，针对 Arxiv、Blogcatalog 和 Wikipedia 这 3 个中等规模数据集我们隐藏 20% 的边进行测试，对于大规模数据集 Flickr 隐藏 10% 的边进行测试. 首先将隐藏边之后的网络输入需要被调整的算法中进行节点表示学习，再通过学习得到的两个节点表示向量之间的内积衡量两个点之间的相似性，即式(14)，我们也与 AutoNE 中相同使用 AUC[62] 作为链接预测应用的性能衡量指标.

$$Sim(v_i,v_j)=\boldsymbol{y}_{v_i}^{\mathrm{T}}\cdot\boldsymbol{y}_{v_j} \tag{14}$$

对于节点分类应用，我们分别使用有节点属性的数据集 Pubmed 和 Cora 对 GCN 算法进行测试，使用没有额外节点属性的 Blogcatalog 和 Wikipedia 数据集对 Deepwalk 和 AROPE 进行测试.

针对 Pubmed 和 Cora 数据集，我们使用基于神经网络的网络表示学习算法中常用的准确率(Accuracy)作为性能衡量指标[7,37]. 准确率也是在节点分类任务中常见的性能衡量指标，对于有 $|V|$ 个节点的网络，l_i 表示节点 i 原本的标签，而 c_i 表示通过节点分类应用预测得到的标签，准确率(Accuracy)的定义表示为式(15)：

$$Accuracy=\frac{\sum_{i=1}^{|V|}\delta(l_i,c_i)}{|V|} \tag{15}$$

其中 $\delta(l_i,c_i)$ 为指示函数，当 $l_i=c_i$ 时 $\delta(l_i,c_i)=1$，否则 $\delta(l_i,c_i)=0$.

针对 Blogcatalog 和 Wikipedia 数据集，使用学习得到全部节点数的 80% 作为训练集，对逻辑回归分类器(Logistic Regression Classifier)进行训练，并与 AutoNE 框架相同，也使用 Micro-$F1$ 作为性能评价指标[24]. Micro-$F1$ 是在分类应用中常见的性能评测指标，为了给出 Micro-$F1$ 的定义，我们首先定义 3 个基础概念：

(1) 真正类(True Positive). 这一类数据表示真实数据是正类，在分类结果中也为正类的样本，缩写为 TP.

(2) 假正类(False Positive). 这一类数据表示真实数据是负类，在分类中被分为正类的样本，缩写为 FP.

(3) 假负类(False Negative). 这一类数据表示真实数据是正类，在分类中被归为负类的样本，缩写为 FN.

由于 Micro-$F1$ 是精确率(Precision)和召回率(Recall)的调和值，所以我们将分别给出这 3 种指标的计算公式. 针对有 L 类标签的样本，l 表示为样本中被预测标签为 l 的样本，则令 $TP(l)$、$FP(l)$ 和 $FN(l)$ 分别表示标签被预测为 l 的真正类、假正类和假负类样本数目. 精确率(Pr)、召回率(R)和 Micro-$F1$ 的定义为式(16)：

$$Pr=\frac{\sum_{l\in L}TP(l)}{\sum_{l\in L}(TP(l)+FP(l))},$$

$$R=\frac{\sum_{l\in L}TP(l)}{\sum_{l\in L}(TP(l)+FN(l))},$$

$$\text{Micro-}F1=2\times\frac{Pr\cdot R}{Pr+R} \tag{16}$$

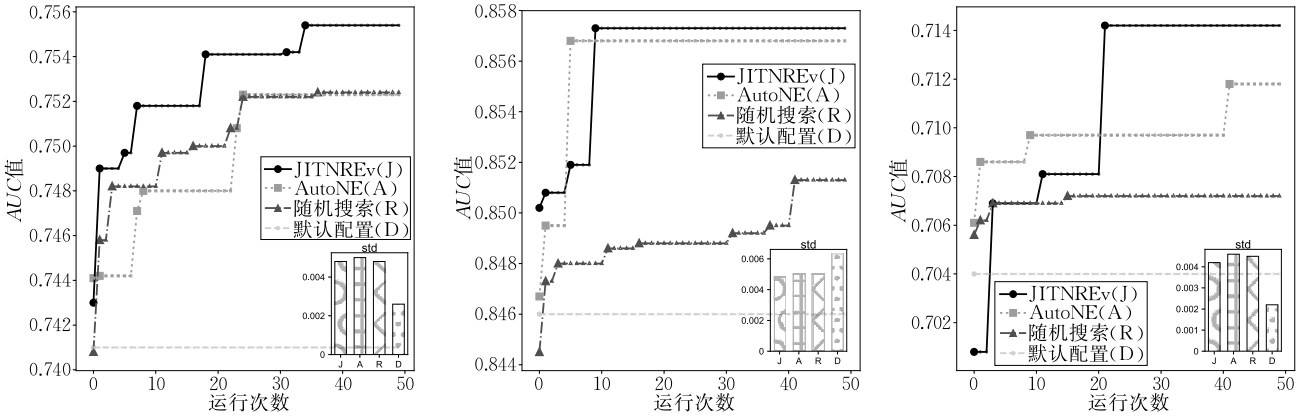
5.3　实验结果与分析

在这一节中我们将对上述实验设置下的 JITNREv 与 3 种基准算法在多种数据集和应用下的性能进行展示.

5.3.1　基于矩阵分解的网络表示学习算法

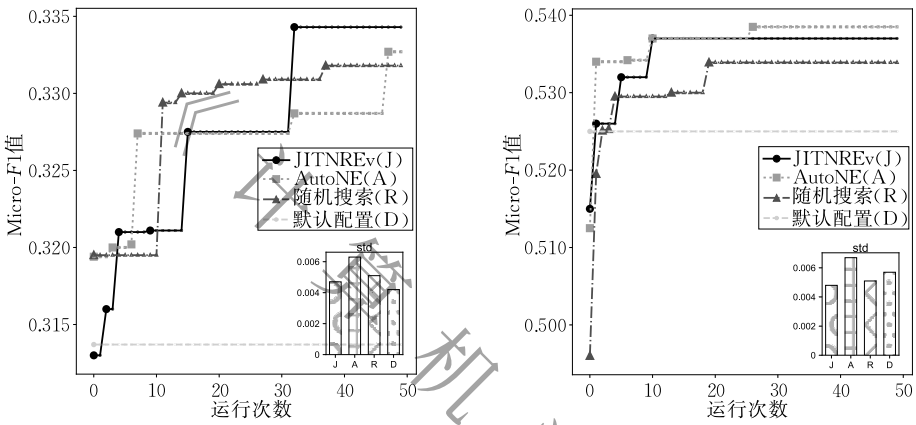
我们使用 AROPE 作为这类方法的代表，通过 Arxiv、Blogcatalog 和 Wikipedia 三个数据集对 JITNREv 和 3 种基准算法的性能进行评测. 我们首先展示在链接预测应用中的性能，见图 5. 我们将每种算法根据上述的实验条件分别运行 5 次，将 5 次结果的平均值进行展示，每张图中的直方图表示 5 次运行结果之间的标准差. JITNREv、AutoNE 和随机搜索性能曲线的起始点都是在超级参数搜索范围内，通过不同采样方法获取参数配置后运行算法得到的性能. 从图中可以看出 JITNREv 能够稳定的通过较少的运行次数获得比基准算法更好的性能，且 JITNREv 的标准差小于使用随机搜索方法和 AutoNE 进行参数调整的标准差. 我们还发现，无论使用什么方式对算法的超级参数进行调整，都会获得好于默认参数配置下的性能. 印证了超级参数对网络表示学习算法性能有明显影响的事实.

由于 Arxiv 数据集中的点没有标签无法进行节点分类的评测，所以我们使用 Blogcatalog 和 Wikipedia 数据集，针对节点分类应用对 AROPE 算法在不同基准算法中的性能进行展示，结果见图 6.



(a) Arxiv数据集中的性能 (b) Blogcatalog数据集中的性能 (c) Wikipedia数据集中的性能

图 5 在资源限制条件下 JITNREv 框架在链接预测应用中的性能(被调整的网络表示学习方法为 AROPE)



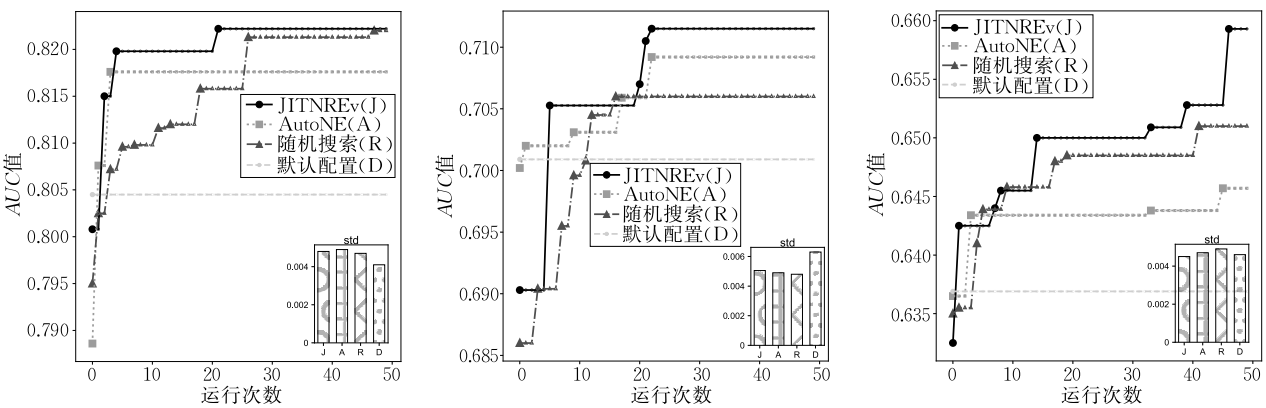
(a) Blogcatalog数据集中的性能 (b) Wikipedia数据集中的性能

图 6 在资源限制条件下 JITNREv 框架在节点分类应用中的性能(被调整的网络表示学习方法为 AROPE)

从图 6 中我们可以得出和针对链接预测应用相同的结论,且即使在 JITNREv 框架中配置参数起始点的性能稍差,也能通过参数调整在较短的运行次数下稳定获得好于基准算法的性能. 在 Wikipedia 数据集上,JITNREv 最终获得的平均性能稍差于 AutoNE 算法,但是相比 AutoNE,JITNREv 框架的性能更快到达稳定状态,且与 AutoNE 相比 JITNREv 在 5 次运行中最终结果的标准差更小,证明 JITNREv 能够通过对参数进行调整获得更稳定的算法性能.

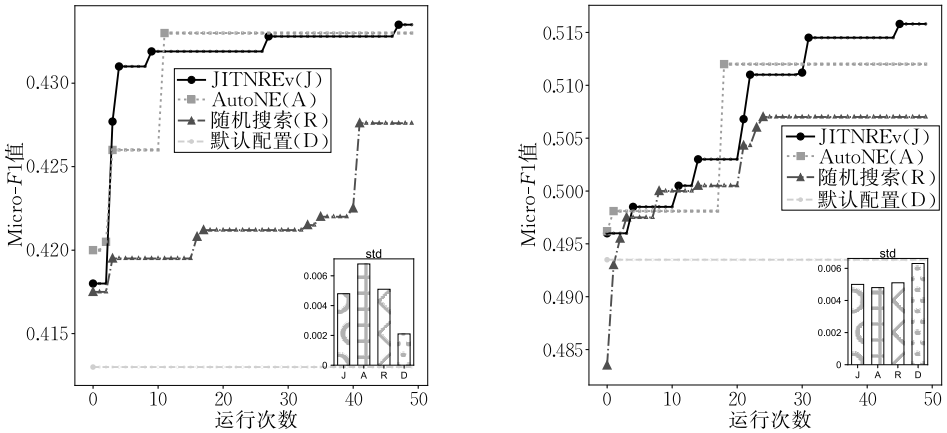
5.3.2 基于随机游走的网络表示学习算法

我们选择了基于随机游走的静态网络节点表示学习算法中最具代表性的工作 Deepwalk[8] 作为这一节中进行超级参数调整的算法. 使用和 AROPE 相同的数据集和实验设置,针对链接预测和节点分类应用的实验结果性能图分别为图 7 和图 8.



(a) Arxiv数据集中的性能 (b) Blogcatalog数据集中的性能 (c) Wikipedia数据集中的性能

图 7 在资源限制条件下 JITNREv 框架在链接预测应用中的性能(被调整的网络表示学习方法为 Deepwalk)



(a) Blogcatalog数据集中的性能　　(b) Wikipedia数据集中的性能

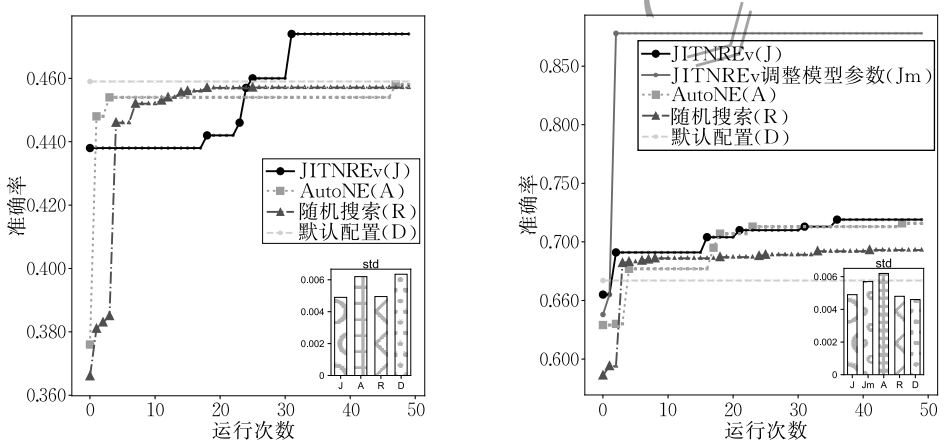
图 8　在资源限制条件下 JITNREv 框架在节点分类应用中的性能(被调整的网络表示学习方法为 Deepwalk)

与基于矩阵分解的网络表示学习算法结论相似,JITNREv 在绝大多数场景下,无论起始参数配置下的性能如何,都能稳定的以较少的运行次数获得高于基准算法的最终性能.此外值得注意的是,在使用 Blogcatalog 数据集进行链接预测应用评测时,AutoNE 算法的性能起始点和推荐默认配置的起始点要高于 JITNREv 和随机搜索的性能起始点,侧面说明了 AutoNE 框架中转化学习模块的作用.

5.3.3　基于神经网络的网络表示学习算法

基于神经网络的表示学习算法通常有大量的超级参数对算法的性能造成影响,在这一节我们选择了这类方法中的代表工作 GCN[7] 进行超级参数调整.由于 GCN 是针对有节点属性信息的网络结构数据集进行节点分类的半监督学习方法,而前面使用的数据集都不包括节点属性信息,所以在这部分我们使用了 Pubmed 和 Cora 两个数据集.

实验结果如图 9 所示,从实验结果中可以看出 JITNREv 框架能够稳定的在较少运行次数下获得高于对比算法的性能,再一次证明我们框架的有效性.此外,由于 JITNREv 框架能够对非数值类型的超级参数进行调整,我们在用 Pubmed 数据集进行评测时,对 GCN 的'*model*'参数进行了调整,由结果可知在运行时间足够长的条件下'*dense*'模型下的性能最高达到 87.45%的准确率,与使用'*gcn*'作为'*model*'参数的默认配置相比性能提升了 31%,与 GCN 论文中的结论保持一致.



(a) Cora数据集中的性能　　(b) Pubmed数据集中的性能

图 9　在资源限制条件下 JITNREv 框架在节点分类应用中的性能(被调整的网络表示学习方法为 GCN)

5.3.4　针对大规模数据集的性能测试

上述的实验都是使用中等规模或者较小规模的数据集进行测试,为了证明 JITNREv 框架在面向超大规模数据集时同样有效,在这一节中我们使用 Flickr[1] 数据集对网络表示学习算法进行超级参数调整.由于 Flickr 数据集中不包含节点属性信息且数据规模较大,因此我们使用运行速度最快的 AROPE 算法和链接预测应用进行性能评测,由于运行超大

规模数据集需要的时间更长，我们将每种算法都只运行 10 次.此外，由于 JITNREv 框架针对超大规模数据集做了额外优化即先进行图粗化且在粗化后的数据上进行超级参数调整，并将最优解应用到原图输出最终结果.为了证明在粗化后的数据上进行超级参数调整的结果与直接在原图上调整同样有效，我们将 JITNREv 的两种运行方式的最终结果都进行展示，分别为直接在输入数据上进行 10 次超级参数调整的结果和在一轮粗化后的数据上进行超级参数调整并直接将最优解应用到原图中这两种，最终的实验结果如图 10.

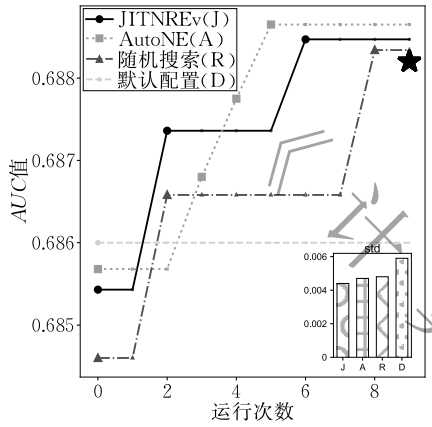


图 10 在大规模网络结构数据集 Flickr 中针对链接预测应用的性能(被调整的网络表示学习方法为 AROPE)

由图我们可以看出，JITNREv 相比随机搜索和默认配置，在 AROPE 的链接预测应用中仍能通过较少的运行次数获得较好的性能.当直接在原图上进行超级参数调整时，JITNREv 的性能略差于 AutoNE，但 AutoNE 在图 10 中的结果是将子图上的最优解在原图上进行再次细微调整的结果，与 JITNREv 相比花费了额外的时间.图 10 中表示 JITNREv 对输入数据进行粗化后，在粗化数据上进行 10 次超级参数调整，并直接将粗化图中的最优参数配置应用到原图上所产生的性能值，我们可以观察到将结果直接应用到原图上能够获得与直接在原图中进行调整相近的结果，且与 AutoNE 的起始点(即第 0 次数据点)相比有更大可能获得更优性能的可能.证明了 JITNREv 框架充分保留了数据集原始的结构特征.

5.3.5 大规模数据集处理及运行时间分析

由于网络表示学习算法的运行时间通常与数据规模正相关，在面对超大规模数据集时为了缩短算法进行超级参数调整的时间，JITNREv 框架对输入数据进行了图粗化处理，通过在保留了数据结构的小规模图上进行超级参数调整，再应用到原图中，降低了总体运行的时间成本并限制了内存占用.除了 JITNREv 框架外，AutoNE 也对超大规模网络结构数据进行了额外处理，他通过随机游走的方式对超大规模数据集进行采样生成多个子图，并在不同子图上分别运行网络表示学习算法，通过随机搜索的方式进行超级参数调整，再通过转化学习将可能在原图上有最好性能的超级参数配置集合应用到原图上输出最终结果.

在这一节中，我们将 JITNREv 和 AutoNE 两种子图处理方式进行比较，分别统计了 AutoNE 的子图点数和所有子图运行时间之和，与 JITNREv 中经过图粗化处理后获得的小规模图的点数和运行时间进行比较，以 Deepwalk 和 AROPE 两种算法为例.对于 AutoNE 框架，我们将采样的子图数设置为 5，子图规模为 $5\%|V|-20\%|V|$；对于 JITNREv 框架，我们进行一轮图粗化.得到的子图点数见表 3，我们可以观察得到，AutoNE 通过随机采样获得的子图规模即使对于相同数据集也存在波动，但是 JITNREv 的粗化结果很稳定，避免了引入额外性能波动.此外，我们将 AutoNE 在所有子图上的超级参数调整时间之和与在 JITNREv 的粗化图上进行超级参数调整的时间进行统计结果如图 11，根据观察我们可知虽然 AutoNE 采样得到的子图中点数都小于 JITNREv 对原图进行粗化后获得的小规模图中节点的数目，但是由于需要采样的子图数目较多，所以运行时间之和依旧要长于 JITNREv.由此证明了 JITNREv 框架针对超大规模数据集进行超级参数调整时进行额外优化的有效性.

表 3 子图点数统计

Datasets	Deepwalk		AROPE	
	AutoNE	JITNREv	AutoNE	JITNREv
Wikipedia	1510,1815,1150,880,1690	2822	1070,1170,1100,1000,675	2822
Blogcatalog	1667,1374,1429,972,705	6838	1470,1270,1425,620,1925	6838
Arxiv	1407,1162,1512,1187,610	3706	865,690,1160,1900,1950	3706

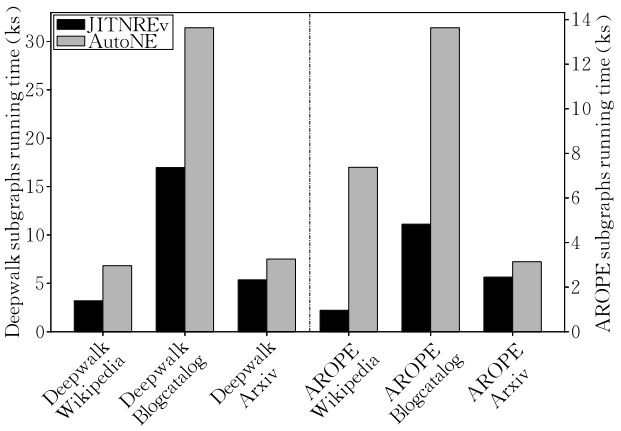


图 11　在不同数据集的子图上超级参数调整框架的运行时间比较

5.3.6　限制最大占用内存的性能分析

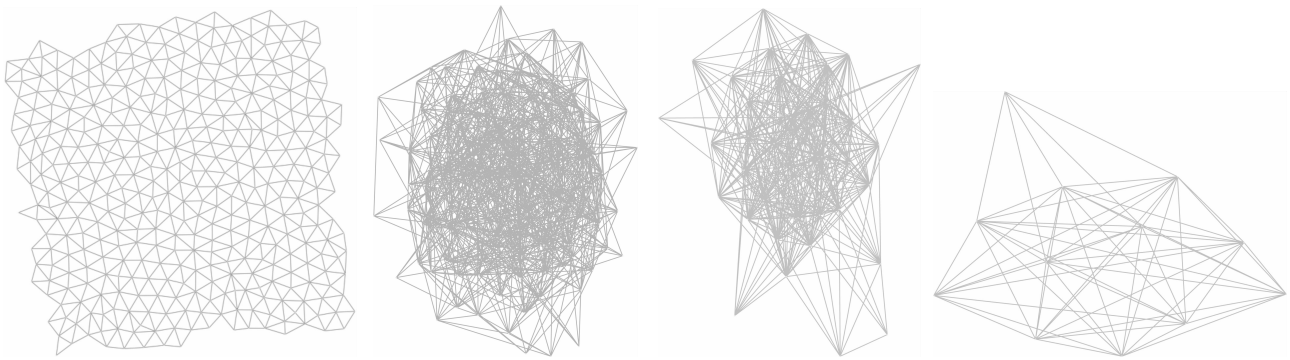
这一节将对 JITNREv 框架在有最大占用内存空间 N_m 限制时的性能进行分析. 由于网络表示学习算法的内存占用与所处理数据集的规模正相关，结合 JITNREv 中对大规模数据集进行预处理时所使用的图粗化算法，我们将对不同粒度的图粗化结果(即粗化后小规模图的点数和边数)、在对应图上进行超级参数调整并将获得的最优超级参数配置结果直接应用在原图上的性能进行展示.

我们以 Wikipedia 数据集为例，使用 JITNREv 框架对 AROPE 和 Deepwalk 算法针对链接预测应用进行超级参数调整. 当 N_m 越小时，图粗化的粒度越大(即粗化层数越高)，当粗化后的数据集点数小于原图的 1/10 时停止进行粗化. 我们在小规模图上分别进行 10 次超级参数调整并直接将获得的最优超级参数配置在原图上运行，最终获得的性能如表 4 所示.

表 4　限制最大内存占用时 JITNREv 框架的性能

粗化层数	图规模		性能(*AUC* 值%)	
	点数	边数	JITNREv(AROPE)	JITNREv(Deepwalk)
0(原图)	4777	184812	71.02	64.51
1	2822	163740	69.80	64.10
2	1699	140706	70.14	62.88
3	1021	115814	69.52	63.76
4	601	85206	69.36	62.61
5	351	51120	67.88	61.92

根据上述实验结果我们可知，当所允许的最大占用内存空间 N_m 越小时，所获得的最优超级参数配置直接应用到原图上的性能越差. 我们对这一现象也进行了分析，以 Possion2D 数据集为例，我们将原数据集和使用图粗化算法逐层进行粗化后的结果通过可视化展示(如图 12 所示). 由图我们可知，当图粗化的粒度越大即 N_m 越小时，粗化后的数据与原始数据相比结构信息的损失会越多，因此在粗粒度的小规模图上进行超级参数调整后所获得的最优超级参数配置直接应用到原图中的性能会越差. 而如何找到内存占用与性能之间的平衡，我们将在之后的工作中进行研究.



(a) Poisson2D原数据　(b) 图粗化第2层结果　(c) 图粗化第4层结果　(d) 图粗化第6层结果

图 12　*Poisson2D* 原始数据和经过图粗化处理后数据的比较

5.3.7　JITNREv 初始值敏感度测试

在这一节我们将对 JITNREv 初始设置的敏感度进行测试. 对于 JITNREv 框架，可变的初始设置只有初始采集样本数 k. 因此我们使用 Arxiv 数据集针对 AROPE 算法的链接预测应用，在相同资源限制条件 $N_t=50$，相同超级参数集合 θ，相同参数采样范围 dis_θ 的情况下，对不同初始采集样本数 k 的取值对框架性能的最终影响进行测试. 实验结论(如

图 13 所示)说明在相同资源限制条件下,不同初始采集样本数 k 的取值对性能不会有明显影响,证明了 JITNREv 框架的稳定性.

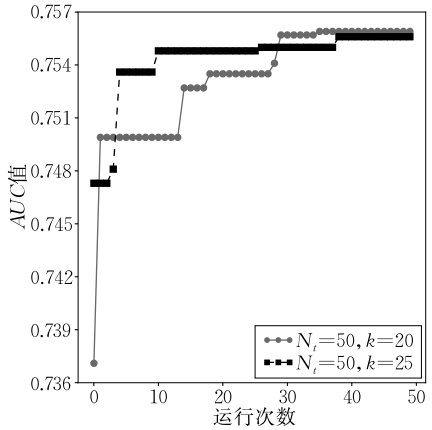


图 13 JITNREv 框架对于 k 值的初始值敏感度测试

5.3.8 JITNREv 框架通用性分析

现有的静态网络表示学习算法根据所使用基本模型的不同可以粗略分为基于矩阵分解、基于随机游走和基于神经网络这三大类.与基于矩阵分解类的算法相比,基于随机游走和基于神经网络这类算法数量较多且超级参数规模较大,因此在这一节中,为更进一步对 JITNREv 框架的通用性和有效性进行验证,我们将再分别选取其中的代表性算法对他们进行超级参数调整.

基于随机游走的方法我们选择了 Node2vec[38] 进行超级参数调整,除了与 Deepwalk 相同,我们还对控制随机游走方式的回退参数(Return Parameter) p 和前进参数(In-out Parameter) q 进行了调整,取值范围均为(0,5](根据 Node2vec 设置默认配置为 $p=4, q=1$).我们针对节点分类应用进行性能测试、实验细节、基准方法、数据集、评测指标等都与 Deepwalk 设置相同,实验结果如表 5 所示.由此我们可知,通过对 Node2vec 中的超级参数进行调整可以获得高于默认超级参数配置下的性能,且 JITNREv 框架相比 AutoNE 和随机搜索算法能够较为稳定的获得最优性能,充分证明了 JITNREv 框架的有效性.

表 5 在资源限制条件下 JITNREv 在节点分类应用中的性能(被调整的网络表示学习方法为基于随机游走的算法 Node2vec)

基准算法	Node2vec(Micro-$F1$%±标准差%)	
	Wikipedia	Blogcatalog
JITNREv	**57.80**±3.58	**43.03**±3.82
AutoNE	57.56±4.66	42.93±4.74
随机搜索	57.67±4.75	42.56±6.01
默认配置	56.80±6.14	41.97±1.28

基于图神经网络的方法我们选择了 GAT[54]、GraphSAGE-Mean[37] 和 Mixhop[55] 进行超级参数调整,除了与 GCN 相同,对训练迭代次数、隐藏层中神经元数目、学习速率、Dropout Rate 和使用 L2 正则化的权值衰退率这 5 个超级参数进行调整外,我们还对 GAT 中的 Batch Size(取值范围[1,64])(默认配置为 8),GraphSAGE 中的第一/二层样本数(取值范围[5,25])(默认配置为 10/5),Mixhop 中的 *adj_pows*(取值范围['0','1','2'])(默认配置为'1')进行了调整.实验针对节点分类应用进行,实验设置及所使用的基准方法,数据集,评测指标等都与之前相同,结果如表 6 所示.从结果中可知,基于神经网络的网络表示学习算法,通过对超级参数进行调整普遍可以获得高于默认超级参数配置下的性能,JITNREv 框架能够较为稳定的获得最优性能,充分证明了 JITNREv 框架的有效性.

表 6 在资源限制条件下 JITNREv 在节点分类应用中的性能(被调整的网络表示学习方法为基于神经网络的一系列算法)

基准算法	Cora 数据集中性能(准确率%±标准差%)			Pubmed 数据集中性能(准确率%±标准差%)		
	GraphSAGE	GAT	Mixhop	GraphSAGE	GAT	Mixhop
JITNREv	87.93±1.93	**84.16**±1.42	**86.27**±2.47	**88.82**±0.58	84.79±0.25	**83.87**±0.51
AutoNE	**88.07**±2.14	83.08±0.92	84.94±0.99	88.42±0.36	**84.81**±0.46	83.57±0.64
随机搜索	87.24±1.98	82.72±0.97	85.19±3.09	88.27±0.47	84.60±0.51	86.27±0.78
默认配置	86.90±0.36	82.09±1.87	81.40±1.02	87.31±0.30	84.53±0.26	80.32±0.80

综上我们可知,JITNREv 框架普遍适用于各类网络表示学习算法的超级参数调整.

6 总 结

本文提出了如何对网络表示学习算法的性能进行公平比较的问题,并针对该问题提出了基于超级参数调整的网络表示学习算法性能公平比较框架 JITNREv.框架通过闭环的整体结构,结合超级参数采样及采样空间剪枝算法,实现了在资源限制条件下的网络表示学习算法性能自动优化,并利用图粗化方法,进一步降低了对超大规模数据集进行超级参数调整时的时间成本和内存占用.此外框架不同模块之间仅通过信息流的方式进行松耦合连接,

将静态网络节点表示学习算法中常用的数据集、数据分析应用及评测指标结合到框架中，实现了框架通用、易用及易于扩展的设计目的.根据实验结果，再次验证了 JITNREv 框架的有效性.

基于超级参数调整的算法性能公平比较问题近期受到了学术界较高的关注，将来我们也计划面向更多的算法特性开展研究工作；此外网络表示学习算法超级参数优化过程中的优化顺序，参数间的互相影响，如何进一步降低网络表示学习算法进行超级参数调整的运行时间也是我们之后的重点研究方向.

参　考　文　献

[1] Tang Lei, Liu Huan. Relational learning via latent social dimensions//Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA, 2009: 817-826

[2] Mccallum A K, Nigam K, Rennie J, et al. Automating the construction of internet portals with machine learning. Information Retrieval, 2000, 3(2): 127-163

[3] Sen P, Namata G, Bilgic M, et al. Collective classification in network data. AI Magazine, 2008, 29(3): 93-106

[4] Leskovec J, Kleinberg J, Faloutsos C. Graph evolution: Densification and shrinking diameters. ACM Transactions on Knowledge Discovery from Data (TKDD), 2007, 1(1): 2-es

[5] Klymko C, Gleich D, Kolda T G. Using triangles to improve community detection in directed networks. arXiv preprint arXiv:1404.5874, 2014

[6] Tand Jian, Qu Meng, Wang Mingzhe, et al. LINE: Large-scale information network embedding//Proceedings of the 24th International Conference on World Wide Web. Republic and Canton of Geneva, Switzerland, 2015: 1067-1077

[7] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks//Proceedings of the 5th International Conference on Learning Representations. Toulon, France, 2017

[8] Perozzi B, Al-Rfou R, Skiena S. Deepwalk: Online learning of social representations//Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA, 2014: 701-710

[9] Wang Daixin, Cui Peng, Zhu Wenwu. Structural deep network embedding//Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA, 2016: 1225-1234

[10] Cruz A F, Saleiro P, Belém C, et al. Promoting Fairness through Hyperparameter Optimization. arXiv preprint arXiv: 2103.12715, 2021

[11] Zoph B, Le Q V. Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578, 2016

[12] Fang Meng, Li Yuan, Cohn T. Learning how to active learn: A deep reinforcement learning approach. arXiv preprint arXiv: 1708.02383, 2017

[13] Zhu Yuqing, Liu Jianxun, Guo Mengying, et al. BestConfig: Tapping the performance potential of systems via automatic configuration tuning//Proceedings of the 2017 Symposium on Cloud Computing. New York, USA, 2017: 338-350

[14] Van Aken D, Pavlo A, Gordon G J, et al. Automatic database management system tuning through large-scale machine learning//Proceedings of the 2017 ACM International Conference on Management of Data. New York, USA, 2017: 1009-1024

[15] Koch P, Golovidov O, Gardner S, et al. Autotune: A derivative-free optimization framework for hyperparameter tuning//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. New York, USA, 2018: 443-452

[16] Jin Haifeng, Song Qingquan, Hu Xia. Auto-Keras: An efficient neural architecture search system//Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. New York, USA, 2019: 1946-1956

[17] Akiba T, Sano S, Yanase T, et al. Optuna: A next-generation hyperparameter optimization framework//Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. New York, USA, 2019: 2623-2631

[18] Du X, Xu H, Zhu F. Understanding the effect of hyperparameter optimization on machine learning models for structure design problems. Computer-Aided Design, 2021, 135: 103013

[19] OtterTune. https://ottertune.cs.cmu.edu, 2021, 6, 24

[20] Zhang J, Liu Y, Zhou K, et al. An end-to-end automatic cloud database tuning system using deep reinforcement learning//Proceedings of the 2019 International Conference on Management of Data. Amsterdam, The Netherlands, 2019, 415-432

[21] Snoek J, Larochelle H, Adams R P. Practical Bayesian optimization of machine learning algorithms//Proceedings of the Advances in Neural Information Processing Systems. New York, USA, 2012: 2951-2959

[22] Bergstra J, Bengio Y. Random search for hyper-parameter optimization. Journal of Machine Learning Research, 2012, 13(2): 281-305

[23] Hutter F, Hoos H H, Leyton-Brown K. Sequential model-based optimization for general algorithm configuration//Proceedings of the International Conference on Learning and Intelligent Optimization. Berlin, Germany, 2011: 507-523

[24] Tu Ke, Ma Jianxin, Cui Peng, et al. AutoNE: Hyperparameter optimization for massive network embedding//Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA, 2019: 216-225

[25] Cui Peng, Wang Xiao, Pei Jian, et al. A survey on network embedding. IEEE Transactions on Knowledge and Data Engineering, 2018, 31(5): 833-852

[26] Zhang Daokun, Yin Jie, Zhu Xingquan, et al. Network representation learning: A survey. IEEE Transactions on Big Data, 2018, 6(1): 3-28

[27] Hamilton W L, Ying R, Leskovec J. Representation learning on graphs: Methods and applications. arXiv preprint arXiv: 1709.05584, 2017

[28] Wang Jizhe, Huang Pipei, Zhao Huan, et al. Billion-scale commodity embedding for e-commerce recommendation in Alibaba//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. New York, USA, 2018: 839-848

[29] Zuo Yuan, Liu Guannan, Lin Hao, et al. Embedding temporal network via neighborhood formation//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. London, UK, 2018: 2857-2866

[30] Liu Xi, Hsien P C, Duffield N, et al. Real-time streaming graph embedding through local actions//Proceedings of the 2019 World Wide Web Conference. New York, USA, 2019: 285-293

[31] Zhu Dingyuan, Cui Peng, Zhang Ziwei, et al. High-order proximity preserved embedding for dynamic networks. IEEE Transactions on Knowledge and Data Engineering, 2018, 30(11): 2134-2144

[32] Du Lun, Wang Yun, Song Guojie, et al. Dynamic network embedding: An extended approach for skip-gram based network embedding//Proceedings of the 27th International Joint Conference on Artificial Intelligence. Stockholm, Sweden, 2018: 2086-2092

[33] Sajjad H P, Docherty A, Tyshetskiy Y. Efficient representation learning using random walks for dynamic graphs. arXiv preprint arXiv:1901.01346, 2019

[34] Zhang Ziwei, Cui Peng, Wang Xiao, et al. Arbitrary-order proximity preserved network embedding//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. New York, USA, 2018: 2778-2786

[35] Ou Mingdong, Cui Peng, Pei Jian, et al. Asymmetric transitivity preserving graph embedding//Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA, 2016: 1105-1114

[36] Wang Xiao, Cui Peng, Wang Jing, et al. Community preserving network embedding//Proceedings of the 31st AAAI Conference on Artificial Intelligence. San Francisco, USA, 2017: 203-209

[37] Hamilton W, Ying R, Leskovec J. Inductive representation learning on large graphs//Proceedings of the Advances in Neural Information Processing Systems. New York, USA, 2017: 1024-1034

[38] Grover A, Leskovec J. Node2vec: Scalable feature learning for networks//Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA, 2016: 855-864

[39] Cao Shaosheng, Lu Wei, Xu Qiongkai. Deep neural networks for learning graph representations//Proceedings of the 30th AAAI Conference on Artificial Intelligence. California, USA, 2016: 1145-1152

[40] Wang Suhang, Tang Jiliang, Aggarwal C, et al. Signed network embedding in social media//Proceedings of the 2017 SIAM International Conference on Data Mining. Philadelphia, USA, 2017: 327-335

[41] Ivanov S, Burnave E. Anonymous walk embeddings//Proceedings of the 35th International Conference on Machine Learning. Stockholmsmassan, Sweden, 2018: 2186-2195

[42] Dutta A, Sahbi H. High order stochastic graphlet embedding for graph-based pattern recognition. arXiv preprint. arXiv: 1702.00156, 2017

[43] Bai Yunsheng, Ding Hao, Qiao Yang, et al. Unsupervised inductive whole-graph embedding by preserving graph proximity//Proceedings of the 25th International Joint Conference on Artificial Intelligence. Macao, China, 2019: 1988-1994

[44] Mousavi S F, Safayani M, Mirzaei A, et al. Hierarchical graph embedding in vector space by graph pyramid. Pattern Recognition, 2017, 61: 245-254

[45] Qu Meng, Tang Jian, Shang Jingbo, et al. An attention-based collaboration framework for multi-view network representation learning//Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. Singapore, 2017: 1767-1776

[46] Zhang Hongming, Qiu Liwei, Yi Lingling, et al. Scalable multiplex network embedding.//Proceedings of the 27th International Joint Conference on Artificial Intelligence. Stockholm, Sweden, 2018: 3082-3088

[47] Ma Yao, Ren Zhaochun, Jiang Ziheng, et al. Multi-dimensional network embedding with hierarchical structure//Proceedings of the 11th ACM International Conference on Web Search and Data Mining. Los Angeles, USA, 2018: 387-395

[48] Liu Weiyi, Chen P Y, Yeung S, et al. Principled multilayer network embedding//Proceedings of the 2017 IEEE International Conference on Data Mining Workshop. New Orleans, USA, 2017: 134-141

[49] Zhao N, Zhang H, Wang M, et al. Learning content—social influential features for influence analysis. International Journal of Multimedia Information Retrieval, 2016, 5(3): 137-149

[50] Bordes A, Weston J, Collobert R, et al. Learning structured embeddings of knowledge bases//Proceedings of 25th AAAI Conference on Artificial Intelligence. San Francisco, USA, 2011: 301-306

[51] Cai H, Zheng V W, Chang K C C. A comprehensive survey of graph embedding: Problems, techniques, and applications. IEEE Transactions on Knowledge and Data Engineering, 2018, 30(9): 1616-1637

[52] Mikolov T, Sutskever I, Chen Kai, et al. Distributed representations of words and phrases and their compositionality//Proceedings of the Advances in Neural Information Processing Systems. New York, USA, 2013: 3111-3119

[53] Pan Shirui, Wu Jia, Zhu Xingquan, et al. Tri-party deep network representation. Network, 2016, 11(9): 12

[54] Veličković P, Cucurull G, Casanova A, et al. Graph attention networks. arXiv preprint, arXiv:1710.10903, 2017

[55] Abu-El-Haija S, Perozzi B, Kapoor A, et al. Mixhop: Higher-order graph convolutional architectures via sparsified neigh-

borhood mixing//Proceedings of International Conference on Machine Learning(PMLR). Long Beach, USA, 2019: 21-29

[56] Abu-El-Haija S, Perozzi B, Al-Rfou R, et al. Watch your step: Learning graph embeddings through attention//Proceedings of the 14th International Workshop on Mining and Learning with Graphs(MLG). London, UK, 2017

[57] Chen Haochen, Perozzi B, Hu Yifan, et al. HARP: Hierarchical representation learning for networks//Proceedings of the 22nd AAAI Conference on Artificial Intelligence. New Orleans, USA, 2018: 2127-2134

[58] Hu Yifan. Efficient, high-qualityforce-directed graph drawing. Mathematica Journal, 2005, 10(1): 37-71

[59] Rossi R A, Ahmed N K. The network data repository with interactive graph analytics and visualization//Proceedings of the 29th AAAI Conference on Artificial Intelligence. Austin, USA, 2015: 4292-4293

[60] Mahoney M. Large text compression benchmark. http://mattmahoney.net/dc/text.html 2020-09-09

[61] Qiu Jiezhong, Dong Yuxiao, Ma Hao, et al. Network embedding as matrix factorization: Unifying Deepwalk, LINE, Pte, and node2vec//Proceedings of the 11th ACM International Conference on Web Search and Data Mining. New York, USA, 2018: 459-467

[62] Hanley J A, Mcneil B J. The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology, 1982, 143(1): 29-36

GUO Meng-Ying, Ph. D. candidate. Her main research interests include network representation learning, hyperparameter optimization.

SUN Zhen-Yu, Ph. D. candidate. His main research interests include network representation learning, combinatorial algorithms and computational complexity.

ZHU Yu-Qing, Ph. D., assistant research professor. Her research interests include distributed systems, data management, system benchmarking and performance optimization.

BAO Yun-Gang, Ph. D., professor, Ph. D. supervisor. His main research interests include computer architecture, operating system and system performance modeling and evaluation.

Background

Nowadays, networks are widely used to represent complex relationships of objects. To analyze network data effectively and efficiently, network representation learning (NRL) for preserving the node similarity in a low-dimensional vector space has attracted considerable research attention in the past few years. Despite the popularity of NRL algorithms, the existing related works heavily rely on manual hyperparameter tuning or network architecture design to achieve the best performance, resulting in costly human efforts when a vast number of models emerge for various graph analysis tasks such as node classification or link prediction. Automated machine learning (AutoML) has been extensively studied to reduce human efforts in hyperparameter tuning for machine learning methods that focus on image or text data, such as Autotune, AutoKeras, and Optuna. However, these methods cannot easily handle large-scale networks due to the coupled relationships among nodes. The recent work AutoNE is the most related work close to ours, targeting NRL algorithms hyperparameter tuning, but it cannot guarantee the process to be completed under the given resource constraints.

In this work, we propose an easy-to-use framework named JITNREv, to compare NRL algorithms fairly within resource constraints based on hyperparameters tuning. The framework has loosely coupled components and adopts a sample-test-optimize process in a closed-loop. It has four main components named hyperparameter sampler, NRL algorithm manipulator, performance evaluator, and hyperparameter sampling space optimizer. All components only interact with each other through the data flow. We use the divide-and-diverge sampling method based on Latin Hypercube Sampling to sample a set of hyperparameters, and trim the sample space around the previous best configuration according to the assumption that "around the point with the best performance in the sample set we will be more likely to find other points with similar or better performance". Large scale of network data also poses great challenges to hyperparameters tuning, since the computational cost of an NRL algorithm increases in proportion to the size of the network. So we use a graph coarsening model to generate a hierarchical graph synopsis for large-scale network data. This model can both reduce the data size and preserve graph structural information. Therefore, JITNREv can more easily meet the resource limitation given by users.

This work is supported in part by the National Key R&D Program of China (No. 2016YFB1000201), and the National Natural Science Foundation of China (No. 61420106013). This work focuses on NRL performance fair comparison within resource limitations based on hyperparameters tuning. Our group has been studying performance modeling, configuration parameters or hyperparameters tuning and network representation learning for years and has proposed a series of works such as JITuNE, ClassyTune and BestConfig.