https://crad.ict.ac.cn
62(3): 694−708, 2025

融合图神经网络的高效子图匹配算法

薛 欣[1,3] 朱天晨[1,3] 孙庆赟[1,3] 周号益[2,3] 李建欣[1,3]
[1](北京航空航天大学计算机学院 北京 100191)
[2](北京航空航天大学软件学院 北京 100191)
[3](北京市大数据与脑机智能高精尖创新中心 北京 100191)
(xuexin@act.buaa.edu.cn)

Efficient Subgraph Matching Algorithm with Graph Neural Network

Xue Xin[1,3], Zhu Tianchen[1,3], Sun Qingyun[1,3], Zhou Haoyi[2,3], and Li Jianxin[1,3]
[1](*School of Computer Science and Engineering*, *Beihang University*, *Beijing* 100191)
[2](*School of Software*, *Beihang University*, *Beijing* 100191)
[3](*Beijing Advanced Innovation Center for Big Data and Brain Computing*, *Beijing* 100191)

Abstract Subgraph matching is an optimization problem in graph, which is to find all matching subgraphs of the query graph in a large target graph. Although subgraph matching is an NP-Hard problem, the problem is common in many fields such as social networks, biochemistry, and cognitive science. Backtracking searching algorithms for subgraph matching have high time complexity, and the pruning strategy is essential to reduce the operating time. However, complex expanding in the existing pruning strategy leads to high complexity of time and space. To balance the efficiency and the effectiveness, only limited neighborhood structure information can be used in conflicting judging, which lets lots of useless states pass the pruning judging and wastes time. An efficient, accurate, and adaptive subgraph matching algorithm is proposed. The algorithm captures the detail structure of the whole graph by graph neural network, builds structure connections, and generates pruning possibilities for all candidate searching states. It replaces complex expanding pruning method with inferring by the neural network model to rapidly estimate the probability of pruning during searching. A data sampling mechanism is designed to alleviate the problem of network training collapse. Experiments show that using our pruning method in traditional backtracking search can improve search efficiency.

Key words subgraph matching; graph neural network; backtracking algorithm; combinatorial optimization; attention mechanism

摘 要 子图匹配是在大型目标图中找出给定查询子图的全部匹配位置,在社交网络、生物化学和认知科学等多个领域都具有关键意义.基于回溯搜索的子图匹配算法时间复杂度高,需要有效的剪枝策略减少运行时间.然而,现有启发式剪枝算法只能依据当前状态的粗略邻域信息做出结构冲突判断,使得大量无效状态难以被筛出,导致子图匹配的性能不佳.提出了一种高效、准确、自适应的融合图神经网络的子图匹配算法,通过图神经网络捕获细粒度邻域结构信息,生成全局结构关联,利用模型推理代替传统剪枝策

收稿日期: 2023-09-12; **修回日期**: 2024-07-05
基金项目: 国家杰出青年科学基金项目(62225202); 国家自然科学基金青年科学基金项目(62202029); 中国科协青年人才托举工程项目(2023QNRC001); 中国人工智能学会-华为 MindSpore 学术奖励基金
This work was supported by the National Science Foundation for Distinguished Young Scholars (62225202), the National Natural Science Foundation of China for Young Scientists (62202029), the Young Elite Scientists Sponsorship Program by CAST (2023QNRC001), and the CAAI-Huawei MindSpore Open Fund.
通信作者: 周号益(haoyi@buaa.edu.cn)

略，估算剪枝概率. 该算法能够在单次查询中有效利用全局信息，显著提升对无效状态的筛选效率. 此外，还设计了一种数据采样机制，以缓解样本分布不均衡导致的网络训练崩溃问题. 实验证明，以基于图神经网络的算法替代回溯式算法的剪枝策略，能够显著提高其搜索效率.

关键词 子图匹配；图神经网络；回溯式算法；组合优化；注意力机制

中图法分类号 TP301.6

DOI: 10.7544/issn1000-1239.202330732 **CSTR:** 32373.14.issn1000-1239.202330732

近年来，图数据在社交网络分析[1]、道路分析[2]、生物蛋白质网络分析[3]、恶意代码监测[4]等众多领域的应用日益广泛. 各式各样的图数据库，例如Neo4j[5]，Janusgraph[6]等，也在学术界和工业界纷纷崭露头角. 子图匹配是查询、分析和处理大规模图数据的一个重要研究问题，用于确定给定查询图在大型目标图中是否存在及其全部匹配位置. 子图匹配算法在众多领域有着广泛应用. 例如：在生物蛋白研究领域，需要利用子图匹配算法来对蛋白质间相互作用网络图进行子结构分析[7]；在社交网络分析领域，需要依据子图匹配信息对社交网络中的社交群体进行结构分析[8]；在道路信息分析领域，需要利用子图信息对路网中的道路结构进行匹配分析[9]；在化学有机大分子研究领域，需要利用分子的子结构匹配信息对分子性质进行分析[10].

子图匹配问题的判定问题已被证实为NP-完全问题，找出所有匹配子图的位置是NP-难问题[11]，而单图数据量的增大使得传统子图匹配算法面临挑战. 在现实世界中，诸如社交网络、生物蛋白合成等应用领域的图数据的单图规模庞大，传统算法难以在有效时间内求得可行解. 例如，在生物学领域常用的Yeast数据集[12]中，目标图的节点数超过3 000，边数超过1万；Youtube提供的社交网络领域公开数据集[13]显示，目标图中的节点数超过100万，边数接近300万. 实验表明[14]，在Youtube数据集上对不超过32个节点的查询图进行询问，传统算法的单次询问运行时间达到几十秒甚至几百秒之久. 除了目标图的规模增大以外，查询图的规模也具有日益增长的趋势. 而在基于启发式搜索的传统算法中，查询图的规模增大，将使得搜索树的深度增大，搜索树的大小将随之快速增长，进而使得运行时间急剧增加，这是实际应用中所难以接受的. 此外，图数据库的大量应用也对子图匹配算法提出更高的要求. 在图数据库中，子图匹配问题对应着数据表的自然连接问题. 因此在面对海量的查询需求时，亟需更加高效的子图匹配算法.

传统子图匹配算法主要为基于启发式的迭代搜索. 近年来，为了提高传统子图匹配算法的搜索速度，现有工作依据查询图结构特点、节点或边的类别特性已提出了大量的剪枝算法. 例如，VF2[15]算法提出了基于匹配节点集合向外进行2层拓展后的节点度数、标签等粗粒度信息的剪枝策略. VF2+[16]，VF2++[17]等算法通过更严苛的剪枝条件减少搜索次数，以达到降低时间复杂度的目的.

然而，这些剪枝算法受时间和空间限制，只能依据匹配节点集合及其有限的粗粒度扩展信息进行局部冲突判断，剪枝能力有限，导致大量无效节点未被筛出. 目前子图匹配算法的计算速度难以满足含有复杂结构的海量图数据的计算需求，面对大量查询请求时无法及时响应. 因而，如何提高子图匹配过程中剪枝算法对全图细粒度信息的使用效率，减少无效节点的漏筛，提高剪枝效率，进而提高子图匹配算法的搜索效率，以适应规模化单图数据的批量查询需求，是目前图数据领域研究中的一个重要问题.

有研究尝试利用图神经网络(graph neural network, GNN)求解图数据领域的NP-难问题，例如Vinyals等人[18]使用指针网络模型近似求解旅行商问题，其效果显著优于部分传统近似算法；Liu等人[19]使用神经网络算法近似求解子图统计问题，相比于传统算法，可在保证误差在可接受范围内，提高求解速度10~1 000倍. 也有研究尝试将图神经网络与传统算法相结合，以求解NP-难问题. 相比于传统算法，这类算法能够在大幅降低计算成本的情况下，求得高质量的解集. 因此，本文试图将图神经网络融入传统子图匹配算法的剪枝过程中，提出一种基于图神经网络的高效、准确、自适应的子图匹配算法. 该算法能够精准捕捉目标图的细粒度结构特征，并构建全局结构关联，在查询阶段快速估计结构冲突的概率，代替传统剪枝策略，以实现快速、自适应、高效的剪枝. 此外，本文针对图神经网络在子图匹配算法训练过程中的采样样本分布不均衡的挑战，设计了一种新的数据采样机制，有效缓解样本分布单一导

致的网络训练崩溃的问题.

本文的贡献总结为:

1)提出了一种融合图神经网络的剪枝概率预测模型,利用全局信息对结构冲突进行更为精准的判断,并将其嵌入子图匹配的回溯搜索算法中,以代替剪枝策略;

2)针对图神经网络子图匹配模型训练过程中,单状态采样机制存在的数据样本分布不均衡问题,设计了面向子分支集合的均衡数据采样机制,有效缓解图神经网络模型的训练崩溃问题;

3)在 Yeast, DBLP 等公共数据集上开展实验验证,实验结果证明神经网络模型能够较好地进行剪枝预测,有效提升子图匹配算法的搜索效率.

1 相关工作

本节主要介绍子图匹配算法以及与图神经网络相关的组合优化算法的研究与应用现状. 本节首先介绍现有的子图匹配算法,主要包含传统启发式搜索算法和基于神经网络的算法. 然后,将进一步介绍在图领域其他组合优化问题上神经网络和传统算法的融合与应用.

1.1 传统子图匹配算法

子图匹配问题是在给定目标图中找出所有与查询图同构的子图. 子图匹配问题是 NP-难问题,目前尚未找到多项式时间复杂度内的求解算法. 随着图数据的广泛应用和图数据挖掘研究的发展,传统子图匹配算法由于时间复杂度过高,难以应对大数据时代海量图查询与计算上的新挑战.

传统子图匹配算法主要有 2 类[14]: 一类是基于深度优先搜索的回溯式搜索(backtracking search)算法,代表性算法有 VF2[15], QuickSI[20] 等,此类算法对图的结构特性依赖较小,具有一定的普适性,可以较好解决多个领域的子图匹配问题. 然而,此类算法运行过程中具有大量递归过程,产生较大的时间和空间开销. 另一类是基于广度优先搜索的多路连接(multi-way join)算法,代表算法有 Binary Join[21], Worst Case Optimal Join[22] 等,此类算法更易支持将图划分成不同子集进行存储,因而在图数据库领域具有广泛应用. 此类算法在某些特定图结构中具有较好的效果,然而其性能极大依赖于图的结构特征.

基于深度优先搜索的回溯式子图匹配算法的主要思想是递归查找匹配集合,试探性扩充匹配集合,并在以扩充后的匹配集合为基础继续进行搜索,直到达到终结状态. 算法在终结状态进行递归回溯,直到将所有分支搜索殆尽. VF2 算法是一种具有代表性的基于深度优先搜索的回溯式算法. 该算法提出了一种可行性法则(feasibility rules)剪枝判断策略,在将候选节点对加入匹配集合前,基于匹配集合向外进行 2 层邻域节点扩展,并依赖邻域中点的度数、标签等粗粒度信息,判断是否存在结构冲突,及时终止对不合法的状态的搜索.

其后的 VF3 算法[23] 通过增加对查询图搜索顺序的确定和对查询图结构的预处理 2 个优化策略,进一步提高搜索效率. 其中,对查询图搜索顺序的确定,是根据查询图中节点的粗粒度邻域扩展信息生成匹配概率值,并据此确定搜索顺序. 该优化策略使得算法优先考虑匹配可能性较小的节点,在早期缩减搜索树的宽度,减少无意义搜索. 查询图结构的预处理,是在查询图搜索顺序确定后,验证搜索顺序相邻的节点之间的前驱和后继结构关系,并在搜索过程中快速校验候选状态与父状态的结构冲突. 该算法是一种对局部结构冲突的快速的粗略检查,及时阻止不符合要求的搜索状态,以减少搜索次数.

基于广度优先搜索的多路连接式子图匹配算法,以边或节点为中心,对已有匹配集合进行试探性连接,在搜索树上进行逐层查找,直到达到终止状态. Binary Join[20] 是一种以边为中心搜索的多路连接子图匹配算法,核心思想是不断将 2 个相关边进行合并,直到找到可行解. Worst Case Optimal Join[21] 则是以节点为中心的广度优先搜索,并根据候选节点的邻居节点进行剪枝. Worst Case Optimal Join 算法保证在最差情况下优于 Binary Join 算法. 在图数据库领域,查询过程往往需要表的连接,而图数据往往以边为单位,分散存储于多个不同的表中,因而基于广度优先的多路连接算法更为合适. 连接式图数据库系统通常将 Worst Case Optimal Join 与 Binary Join 结合,例如 EmptyHeaded[24], LogicBlox[25] 等. 基于广度优先搜索的算法在一部分剪枝思路上,与基于深度优先搜索的算法是类似的. 因而,其剪枝算法也具有互相借鉴和迁移的可能性.

在本文研究中,主要考虑基于深度优先搜索的回溯式算法,在此基础上提出基于神经网络的更为精准的剪枝策略.

1.2 图神经网络与指针网络

本节中将首先介绍常用于图数据表征的图神经网络算法和图指针网络模型,并对比分析组合优化问题上现有的基于神经网络的求解算法.

1.2.1 图神经网络

图神经网络[26]是一种经典的机器学习算法，它通过网络中节点之间的信息传递来获取图中的依存关系，从中提取重要信息并进行推理. 随着图数据结构的普及，图神经网络已经成为许多重要应用的强大工具. 图卷积神经网络[27]（graph convolutional network, GCN）是一种基于离散卷积的图神经网络模型，可以有效提取空间特征，较好地处理点分类问题，并在不同场景下衍生出关系图卷积神经网络模型[28]、时空图卷积神经网络模型[29]等具有强大表达能力的模型. GraphSAGE[30]（graph sample and aggregate）则将改进的侧重点放在采样过程中，在每次迭代过程中计算一定深度内的邻居节点聚合信息，在大规模图数据中能够较好地降低训练成本. GIN[31]（graph isomorphism network）则在聚合过程中引入多层感知结构，使网络能够捕获全部标签信息，而不仅仅是标签的相对分布信息. GIN 模型在图同构问题和点分类问题上有着出色表现，但存在难以泛化到其他领域的问题. GAT[32]（graph attention network）充分考虑到同一个节点的不同邻居之间的重要性差异，为邻居节点之间设置注意力系数，用以量化一个邻居节点对当前节点的重要程度.

1.2.2 图指针网络

指针网络（pointer network, PN）模型[18]是 Vinyals 等人提出的一个序列到序列模型，主要目的是解决序列到序列模型中输出对输入的严重依赖性. 在序列到序列问题中，一般使用编码器-解码器（encoder-decoder）模型，并在编码器与解码器内部设计注意力机制，使得模型能够从编码器和解码器的隐式状态中提取信息. 指针网络模型修改了注意力机制，通过编码器和解码器信息整合后的权重直接计算每一个候选单元的概率. Wang 等人[33]在片段抽取任务中应用指针网络模型，并获得优异表现.

图指针网络（graph pointer network, GPN）[34]是指针网络模型在图上的应用. 图指针网络依然采用编码器-解码器结构，通过图神经网络对图信息进行嵌入，然后利用长短期记忆网络作为编码器，利用指针网络模型中的指针网络层作为解码器，最终生成节点的分类信息. 图指针网络能够较好地依据给定输入序列，生成可变长度的答案序列，在凸包问题、旅行商问题等图上求解不定长序列问题中有着出色表现.

1.2.3 融合图神经网络的优化问题求解

近年来，一些研究将传统算法思路与神经网络算法相结合，用于解决一部分图领域中的 NP-难问题. Khalil 等人[35]于 2017 年提出了一种基于图神经网络的贪心式算法框架，适用于最小点集覆盖问题、最大割问题和旅行商问题 3 类图领域的经典 NP-难问题. 相较于传统算法，该框架下算法能够有效提高时间效率. 以最小点覆盖问题为例，先对图结构信息进行嵌入，然后贪心地选出当前局部最优解，加入解集，以局部解为基础继续进行嵌入和选取操作，直到找到完整的可行解. 然而，贪心式算法仅适用于寻找最优可行解的问题，不适用于找到全部可行解的问题. Wang 等人[36]于 2022 年提出一种基于图神经网络的搜索式子图匹配问题算法，该算法在传统搜索式算法执行前，利用图神经网络预测查询图中每个节点的匹配概率，据此确定节点的搜索顺序，以减少搜索状态数，在不影响解的质量的情况下减少时间开销. 然而，该算法的本质是对候选点进行排序，减少搜索树宽度，无法直接对不合法状态进行剪枝，优化效果有限. Ying 等人[37]提出了基于图神经网络进行匹配子图存在性判定的算法，利用图神经网络近似求解 NP-完全问题，但不能对目标图中的全部匹配位置进行检测. Roy 等人[38]提出了子图相似性比较算法，可以快速从目标图库中找出与查询图具有较高子结构相似性的目标图，但无法对目标图进一步拆分，无法找到目标图中的具体匹配位置.

本文利用神经网络的信息提取与预测能力，对传统回溯式搜索过程中的候选集合进行筛选，以加速传统子图匹配算法的搜索过程.

1.3 注意力机制

如何使模型聚焦于样本信息中的关键信息，是提升神经网络模型学习能力的要点. 注意力机制的引入可以帮助神经网络模型较好地选取和甄别有效信息，使模型能够聚焦于更有价值的信息，而非学习和记忆全部输入信息. 同时，注意力机制能够帮助模型快速获得长程依赖，减少计算开销.

Vaswani 等人[39]最早在 Transformer 模型中提出自注意力机制，其主要作用是减少对外界信息的干扰，捕捉数据或特征之间的内部相关性. 自注意力机制允许对输入输出序列的依赖项进行建模，而无需考虑它们在序列中的距离. 鉴于其在序列问题上的优异表现，自注意力机制被 BERT[40], Longformer[41], Transformer-XL[42], PVT[43]等模型所采纳，并被不断完善和改进.

2 融合图神经网络的高效子图匹配算法

本节介绍了融合图神经网络的高效子图匹配算

法的框架. 首先介绍子图匹配问题描述与整体框架, 然后从均衡采样机制、候选集预测模型和模型融合3个方面展开介绍. 其中面向子分支集合的均衡采样机制, 用以生成训练数据; 基于图神经网络的候选集预测模型, 用以估测剪枝概率, 替代传统剪枝策略; 模型融合则是将传统回溯式算法与本文提出的候选集预测模型相结合.

2.1 问题描述

在介绍本文提出的子图匹配算法前, 首先对子图匹配问题和传统回溯式搜索算法给予形式化描述.

Carletti 等人[23]对子图匹配问题做出了形式化的定义. 假设存在目标图 $G_T=(V_T,E_T)$ 和查询图 $G_Q=(V_Q,E_Q)$, 1个匹配 $m\subset V_T\times V_Q$ 需要满足:

$$\forall u\in V_Q,\ \exists\mu(u)\in V_T:(u,\mu(u))\in m, \tag{1}$$

$$\forall u,u'\in V_Q,\ u\neq u'\Rightarrow\mu(u)\neq\mu(u'), \tag{2}$$

$$\forall(u,u')\in E_Q,\ \exists(\mu(u),\mu(u'))\in E_T, \tag{3}$$

$$\forall u,u'\in V_Q,(\mu(u),\mu(u'))\in E_T\Rightarrow(u,u')\in E_Q, \tag{4}$$

其中 $\mu(u)$ 表示 G_Q 中的点 u 在 G_T 中的关联匹配点. (u,v) 表示以点 u 与点 v 为端点的边. 对于点或边上包含标签的图, 还需要满足:

$$\forall u\in V_Q,\ \lambda(u)=\lambda(\mu(u)), \tag{5}$$

$$\forall(u,u')\in E_Q,\ \lambda(u,u')=\lambda(\mu(u),\mu(u')), \tag{6}$$

其中 λ 表示点或边的标签信息. 子图匹配问题即是从目标图中找出所有的匹配 m 的集合 M.

为了形式化说明传统回溯式算法, 本文给出其伪代码, 如算法1所示. 其中, 本文提出的融合图神经网络的高效子图匹配算法, 主要对传统回溯式算法的行⑧处生成候选集的过程中进行剪枝, 通过图神经网络的预测, 筛选和删除不可能产生解的节点, 以达到剪枝的目的, 减少不必要的搜索过程.

算法 1. 回溯式子图匹配算法 *search*(*s*).

输入: 当前状态 *s*;

输出: 所有匹配集合 *M*.

① $M\leftarrow\varnothing$;
② if 当前状态达成完成匹配子图 then
③ return {状态 *s* 中的匹配关系 *m*};
④ end if
⑤ if 当前状态不符合匹配规则 then
⑥ return ∅;
⑦ end if
⑧ 生成候选集 *C* 和当前查询节点 *v*;
⑨ for *u* in *C* then
⑩ $s'\leftarrow\{(u,v),s\}$;
⑪ $M'\leftarrow search(s')$;
⑫ $M\leftarrow M\cup M'$;
⑬ end for
⑭ return *M*.

2.2 整体框架

本文提出的融合图神经网络的高效子图匹配算法整体框架如图1所示.

图1中以搜索树的形式展现了回溯搜索的过程, 搜索树每个状态中包含已经匹配的子图结构, 其中

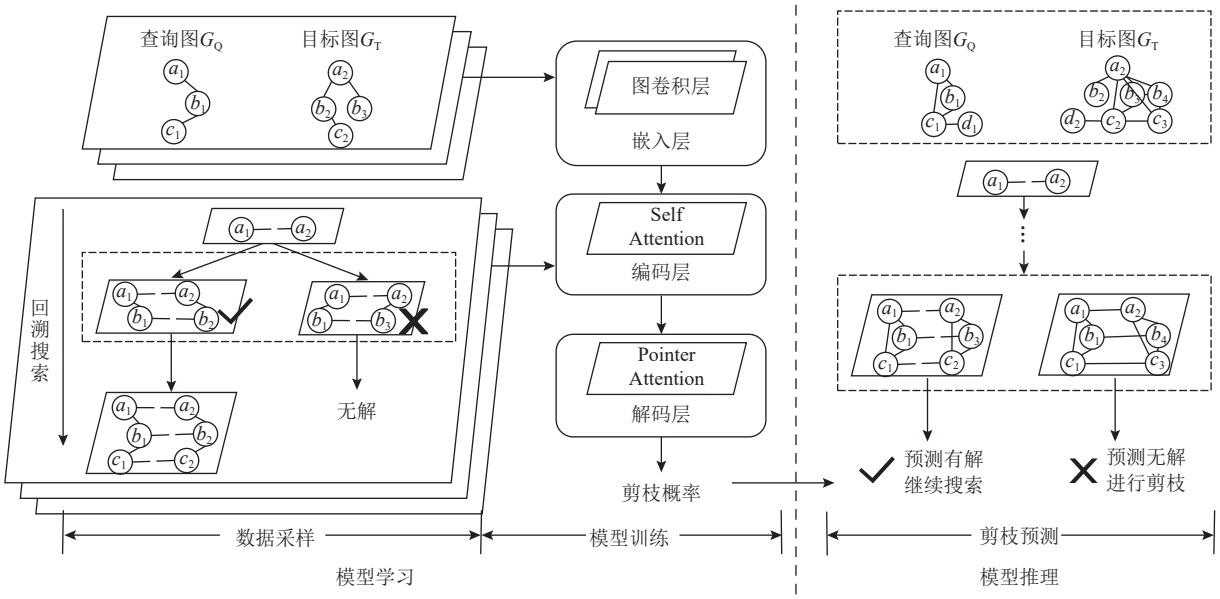


Fig. 1 Overall framework diagram of subgraph matching algorithm with graph neural network

图1 融合图神经网络的子图匹配算法整体框架图

实线为同一图中实际存在的边，虚线表示不同图中关联点的匹配关系. 在本算法中，学习过程离线进行，预先对大规模的查询图进行学习；其推理过程在线进行，即子图匹配搜索过程中进行候选节点的剪枝预测.

本文设计了基于传统回溯式搜索的采样算法，用以采集训练和测试神经网络模型所需的样本数据. 然而，以单搜索状态为单位进行采样容易产生样本分布不均衡问题，导致网络训练崩溃. 因而，本文提出以同一搜索状态的子状态集合为单位进行采样的算法，缓解样本分布差异问题.

本文设计了候选集预测模型，以提高剪枝准确率，并替代传统剪枝策略. 传统剪枝策略受时间和空间限制，只能利用匹配节点集合及其度数、标签等粗粒度扩展信息进行局部冲突判断，信息利用率低. 本文提出利用神经网络模型捕获目标图局部结构信息并生成全局信息关联，在搜索过程中快速验证结构冲突，从而实现更加精准的剪枝优化.

本文将候选集预测模型与传统回溯式子图匹配搜索算法相结合，构成融合图神经网络的子图匹配算法，利用更为精准的剪枝策略，减少回溯式算法的搜索状态.

2.3 面向子状态集合的均衡采样机制

候选集预测模型需利用当前搜索状态信息，估测剪枝概率. 模型训练所需数据采样自回溯式算法执行过程. 所采集的样本数据包含搜索状态信息和判别标签信息 2 个部分. 其中搜索状态信息为包括已经构成匹配的子图节点和匹配关系；判别标签信息则表示搜索状态是否会在未来产生可行解，即搜索状态是否需要剪枝.

本节中首先阐述样本均衡问题和面向子状态集合采样的必要性，然后介绍基于回溯式搜索算法进行子状态集合采样的算法.

2.3.1 样本均衡问题

依据单搜索状态为单位进行采样容易产生样本分布不均衡问题，进而导致模型在训练过程中容易出现过拟合或无法拟合的问题. 本文列举出一组样本示例来进行说明，如表 1 所示.

Table 1 Comparison Table of a Set of Examples

表 1 一组样本示例对照表

样本编号	当前搜索状态	可能的最终深度
1	{(0, 1 721)}	1，2，3，4
2	{(0, 1 721)，(1, 515)，(2, 30)}	3，4

对于预测模型而言，不同搜索层级的样本的学习难度存在不均衡问题. 表 1 中 2 个样本均取自 1 个大小为 4 的查询图示例，但分别来自搜索树的第 1 层和第 3 层. 在样本 1 中，匹配节点对只有 1 个，模型做出预测所依据的输入信息更少，但其搜索的深度可能在 1~4 之间变化，答案变化区间更大. 因而，相比于层级高的样本，层级低的样本更难以被模型学习.

单搜索状态采样后，不同搜索层级的样本独立存在，无差别的训练将使模型难以注意到样本之间的难易程度差异，可能导致模型训练困难.

为解决不均衡问题，本文提出以子状态集合替代单状态进行采样的算法，将同一搜索状态下的子状态视为整体进行采样. 该算法保证同一集合中的状态来自同一层级，不仅有利于模型横向对比同一层级的子样本进行学习，还可以使模型在 1 次预测中对多个子状态进行判断，提高时间效率.

2.3.2 基于搜索的子状态集合采样算法

面向子状态集合的采样算法思路如图 2 所示，图 2 中以搜索树的形式展现了回溯搜索的过程，搜索树每个状态中包含已经构成匹配的子图结构. 首先进行回溯式搜索，对搜索树中所有状态标记是否可以在未来产生可行解. 然后对各个层级分别进行随机抽样，每层级内选取若干组样本，每组样本为某一状态下的所有子状态构成的集合.

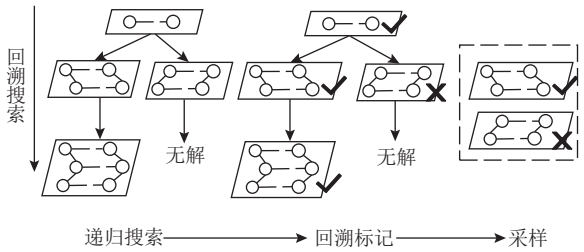


Fig. 2 Illustration of sampling algorithm for substate set

图 2 面向子状态集合的采样算法示意图

所抽取的样本信息 x 包括已经构成匹配的节点对集合 N、来自查询图中的待匹配节点 v、来自目标图中的候选节点集合 C、能在未来形成可行解的候选节点集合 P. 由于 VF3 算法在搜索过程中，固定了查询图的搜索顺序，因而每一层级的待匹配节点 v 有且只有 1 个节点. 上述变量的形式化表述为：

$$N = \{(u,v) | u \in V_T, v \in V_Q\}, \tag{7}$$

$$v = order(|N|+1), \tag{8}$$

$$C = \{u | u \in V_T, label(u) = label(v), feasible(u) = \text{True}\}, \tag{9}$$

$$P=\{u|u\in V_{\mathrm{T}},label(u)=label(v),\ feasible(u)=\mathrm{True},\exists\mu(\exists\mu(u,v)\in\mu)\},\tag{10}$$

其中 $order(i)$ 表示 VF3 搜索过程中第 i 层状态的查询图待匹配的节点，$feasible(u)$ 表示 VF3 的可行性判定函数，μ 表示 1 个完整的匹配. 可行节点集合 P 是候选节点集合 C 的子集.

基于搜索的子状态集合采样算法代码思路如算法 2 所示，在所有子状态回溯后即可获得当前状态的判别标签，并进行采样.

算法 2. 基于搜索的子状态采样算法 $sample(s)$.

输入：状态 s；

输出：有解标记 f，样本集合 X.

① 初始化样本集合 $X\leftarrow\varnothing$

② if 当前状态已产生完整匹配子图 then

③ return True, X;

④ end if

⑤ if 当前状态不符合匹配规则 then

⑥ return False, X;

⑦ end if

⑧ 生成候选集 C 和当前查询节点 v;

⑨ 初始化有解候选节点集合 $P\leftarrow\varnothing$;

⑩ 初始化有解标记 $f\leftarrow$ False;

⑪ for u in C then

⑫ $s'\leftarrow\{u,v,s\}$;

⑬ $f',X'\leftarrow sample(s')$;

⑭ $X\leftarrow X\cup X'$;

⑮ if f' then

⑯ $P\leftarrow P\cup u$;

⑰ end if

⑱ $f\leftarrow(f\mid f')$;

⑲ end for

⑳ $x\leftarrow(s,v,C,P)$;

㉑ $X\leftarrow X\cup x$;

㉒ return f,X.

2.4 基于图神经网络的剪枝预测模型设计

候选解预测模型利用图神经网络模型捕获目标图局部结构信息，并利用注意力机制生成全局结构关联，以便于快速验证全局结构冲突，实现精准剪枝判断，减少冗余搜索，提高时间效率. 候选集预测模型的整体架构如图 3 所示，分为嵌入层、编码层和解码层 3 个部分.

嵌入层根据目标图中的度数排名和类别统计信息来确定每个节点的特征信息，并利用图卷积神经网络提取图细粒度结构信息. 编码层将查询图和目标图的信息进行整合，编码生成解码层所需的候选信息和查询信息. 解码层利用候选信息与查询信息进行分析，根据查询信息确定候选集合中每一个节点的剪枝概率. 模型最终输出的剪枝概率即是嵌合子图匹配算法对无效状态进行剪枝的依据，当一个状态的剪枝概率达到某一阈值时，则对该状态进行剪枝操作.

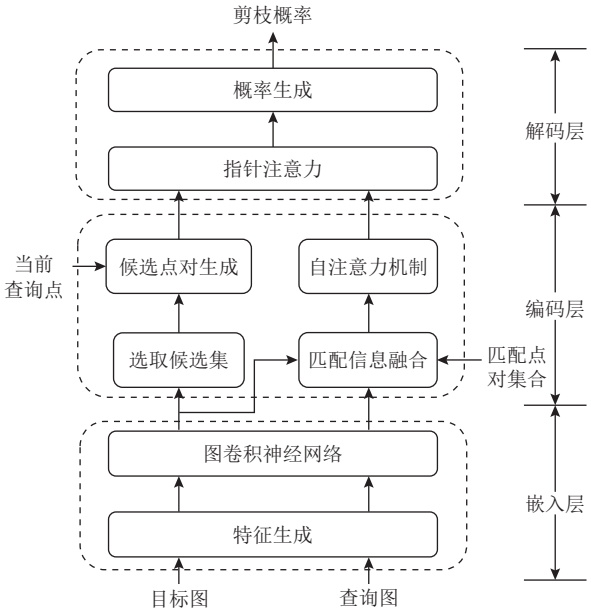


Fig. 3 Illustration of pruning predicting model

图 3 剪枝预测模型整体结构图

2.4.1 结构特征生成与嵌入

嵌入层主要结构如图 4 所示，其主要任务是进行节点特征生成，以便于模型更好地学习和关注重要信息.

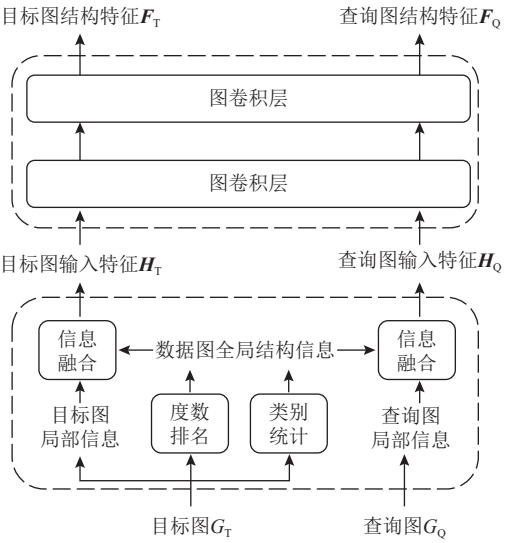


Fig. 4 Structure diagram of embedding layer

图 4 嵌入层结构图

在 VF3 的剪枝策略中，主要利用节点的度数和标签信息进行甄别，这 2 类信息可以较好地体现节点之间的差异性，因而本文也将使用这 2 类信息作为节点输入特征. 同时为了适应不同图中特征的全局变化尺度，将度数在全局中的排名和标签在全局中的数量也作为输入特征.

本文研究中的输入特征的提取思路与 VF3 大致相同. 将节点输入信息划分为 4 个维度，其中前 2 个维度为节点的自身基本信息，包括节点的度数和标签编号；后 2 个维度为节点在图中的重要性信息，包括当前节点的度数排名和标签在全局中的数量. 前 2 个维度用于嵌入节点自身及其周围的局部信息，后 2 个维度用于嵌入节点参照全局排名和统计的全局相对信息，以适应全局变化尺度. 通过融合局部基本信息与全局尺度信息，模型能够更好地提取图结构特征.

不同于 VF3，为了适应图神经网络模型，本文不仅需要对目标图进行特征生成，还需要对查询图进行特征生成. 然而，由于查询图大小和结构差异巨大，查询图自身的度数排名和标签数量随查询图的变化而产生巨大的分布差异，不利于模型的学习. 因而，对于查询图的度数排名、标签数量的生成等全局尺度信息，本文采用目标图的全局度数和标签作为映射依据，以保证训练过程的稳定性. 即查询图的度数排名是目标图中同等度数的节点的排名，查询图的标签数量是目标图中相同标签的数量统计信息.

然后，对归一化处理后的特征信息进行特征传递和局部结构特征聚合. 图卷积神经网络模型可以较好地掌握图的局部结构信息，因而，本文采用图卷积神经网络分别对目标图和查询图的细粒度结构信息进行提取，生成每个节点特征. 在图卷积层中，使用 2 层图卷积层以便于充分掌握图的邻域细粒度结构信息，生成目标图特征信息 $\boldsymbol{F}_{\mathrm{T}}$ 和查询图特征信息 $\boldsymbol{F}_{\mathrm{Q}}$.

在图卷积层后，利用激活函数来捕获非线性因素，增强模型的表达能力.

2.4.2　候选信息生成与匹配节点对信息融合

编码层主要负责候选节点对信息的融合和查询信息的生成，其结构如图 5 所示. 左侧部分为候选集合信息融合，右侧部分为查询信息融合.

候选信息包含目标图中的待匹配节点 v 和在目标图中等待与之匹配的候选节点集合 C. 查询信息包含已经构成匹配的子图结构信息 s. 若将待匹配节点 v 作为查询信息中的一部分，由于在目标图中缺少与之匹配的数据节点，导致 2 个图节点对无法一一对应，同时需要增加额外信息来表示该节点与其他查询图节点的区别，增大模型参数量，提升训练难度. 故而，本文研究中将待匹配节点与候选节点集合 C 中的每一个点逐一结合，构成候选节点对集合，作为候选信息.

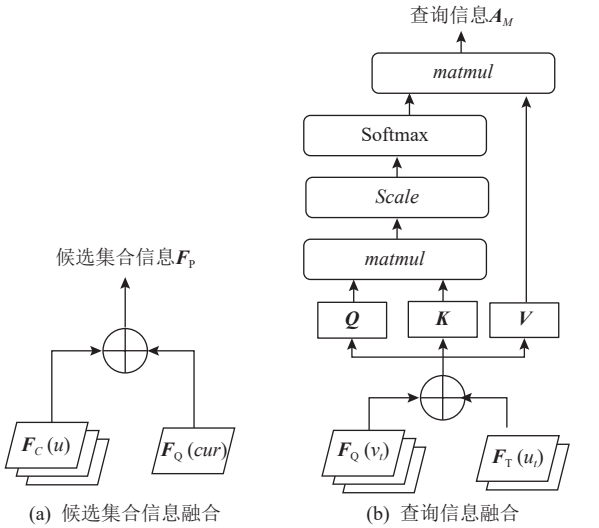


Fig. 5　Structure diagram of encoding layer

图 5　编码层结构图

由于本文采集的同一组样本拥有共同的父状态，因而同组样本中已经构成的匹配结构 N 相同且待匹配节点 v 唯一，这使得本文可以一次性对 C 中全部节点进行统一分析和预测，而非逐个状态预测. 候选节点对集合信息生成方法如下.

首先，从目标图特征信息 $\boldsymbol{F}_{\mathrm{T}}$ 中选出当前状态全部候选节点的特征信息，候选节点特征为

$$\boldsymbol{F}_C = \boldsymbol{F}_{\mathrm{T}}(u),\ u \in C. \tag{11}$$

目标图候选集合中的每一个节点的特征分别与查询图候选节点进行组合，构成候选节点对集合，形式化表述为

$$\boldsymbol{F}_{\mathrm{P}} = \boldsymbol{F}_C(u) + \boldsymbol{F}_{\mathrm{Q}}(v), u \in C, \tag{12}$$

其中 $\boldsymbol{F}_{\mathrm{Q}}$ 为查询图待匹配节点的特征向量. 组合后的候选节点对特征信息 $\boldsymbol{F}_{\mathrm{P}}$ 将作为输入解码层的候选信息.

而输入解码层的查询信息包含来自 2 个图中已经构成匹配的节点对集合信息，模型将利用已有匹配信息预测未来匹配节点对. 设 l 为当前搜索层数，对匹配节点对集合 M 的定义为

$$M = \{(v_t, u_t) | v_t \in G_{\mathrm{Q}}, u_t \in G_{\mathrm{T}}, t = 1, 2, \cdots, l\}. \tag{13}$$

生成查询信息首先需要按照匹配节点对信息，将来自目标图的节点特征和来自查询图节点的特征进行一一匹配组合，得到匹配节点对特征，形式化表述为

$$\boldsymbol{F}_M = \boldsymbol{F}_{\mathrm{Q}}(v_t) + \boldsymbol{F}_{\mathrm{T}}(u_t), v_t \in G_{\mathrm{Q}}, u_t \in G_{\mathrm{T}}, (v_t, u_t) \in M \ . \quad (14)$$

$\boldsymbol{F}_M$为所有匹配节点对特征向量组成的矩阵. 然后, 需要将匹配节点对中的不同节点对进行特征融合, 以建立全局结构关联, 引入自注意力模块, 使得模型更好地关注查询图内部的结构特征, 注意力计算方法为

$$\boldsymbol{A}_M = softmax\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d_K}}\right)\boldsymbol{V}, \quad (15)$$

其中

$$\boldsymbol{Q} = \boldsymbol{W}_{\boldsymbol{Q}}\boldsymbol{F}_M, \quad (16)$$

$$\boldsymbol{K} = \boldsymbol{W}_{\boldsymbol{K}}\boldsymbol{F}_M, \quad (17)$$

$$\boldsymbol{V} = \boldsymbol{W}_{\boldsymbol{V}}\boldsymbol{F}_M. \quad (18)$$

自注意力模块计算得出的结果$\boldsymbol{A}_M$将作为输入解码层的查询信息.

2.4.3 交互注意力与剪枝概率生成

解码层计算查询信息和候选信息的交互注意力, 并依据注意力向量计算候选集中每个节点的剪枝概率, 主要结构如图 6 所示.

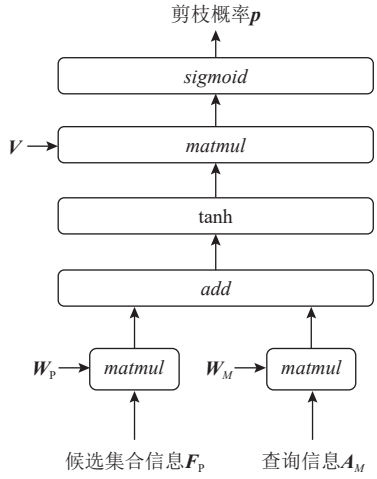


Fig. 6 Structure diagram of decoding layer

图 6 解码层结构图

在编码层中, 已经将图结构信息转化成了查询信息和候选信息, 在本层中按照图指针网络中计算交互注意力的方式, 计算查询信息与候选信息之间的交互注意力, 进而求出剪枝概率. 对候选信息与查询信息之间的交互注意力计算方式为

$$\boldsymbol{s}_i = \tanh(\boldsymbol{W}_{\mathrm{P}} \times \boldsymbol{F}_{\mathrm{P}}(i) + \boldsymbol{W}_M \times \boldsymbol{A}_M) \times \boldsymbol{V}, \quad (19)$$

其中$\boldsymbol{W}_{\mathrm{P}}$, $\boldsymbol{W}_M$, $\boldsymbol{V}$为可训练参数.

根据交互注意力, 计算出每个节点的剪枝概率, 计算方法为

$$\boldsymbol{p} = sigmoid(\boldsymbol{s}). \quad (20)$$

在候选集预测模型中产生的剪枝概率, 将作为融合图神经网络的子图匹配算法中判断搜索分支是否需要进行剪枝操作的依据. 最终生成的概率在 0~1 之间, 以 0.5 作为是否剪枝的分界.

2.4.4 损失函数

在本任务中, 预测每一个候选点是否被选中, 是一个二分类问题, 交叉熵函数是常用的分类问题损失函数. 然而, 在本任务中, 正样本往往出现概率较低, 容易出现正负样本不均衡的问题. 为了平衡样本分布的差异性, 我们选用交叉熵的变体 Focal Loss 作为损失函数, Focal Loss 通过设置平衡系数缓解样本分布不均衡问题. Focal Loss 形式化表述如式(21)所示:

$$\boldsymbol{l} = \begin{cases} -\alpha_1(1-\boldsymbol{p}_i)^{\gamma}\mathrm{lb}(\boldsymbol{p}_i), & \boldsymbol{p}_i = 1, \\ -\alpha_0\boldsymbol{p}_i^{\gamma}\mathrm{lb}(1-\boldsymbol{p}_i), & \boldsymbol{p}_i = 0. \end{cases} \quad (21)$$

其中, α_i为分类样本类别平衡系数, γ为样本难易程度平衡指数.

2.5 嵌合神经网络的优化搜索算法设计

本文将经过改造的传统回溯式算法与基于神经网络的候选集预测模型进行嵌合, 组成一种融合神经网络的回溯式搜索算法.

在算法进行子图匹配搜索过程中, 基于神经网络的候选集预测模型只进行推理、不进行训练, 需要固定模型中的参数, 以防止模型出现抖动. 同时, 不再如训练过程中一样采用批数据形式, 而是将单组数据直接输入模型中.

对传统回溯式算法进行改造, 使得算法可以在递归过程中依据神经网络模型的预测结果进行剪枝, 算法主要包含 3 个模块: 预计算模块、VF3 搜索模块和神经网络预测模块. 3 个模块的交互关系如图 7 所示.

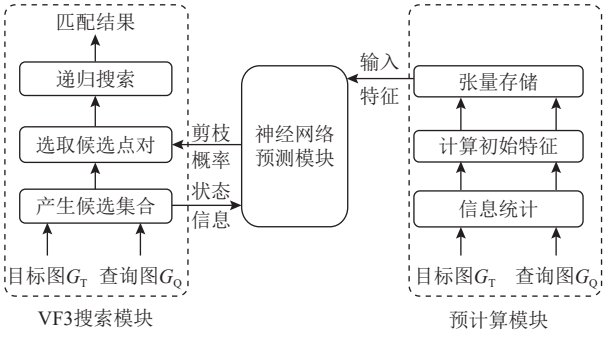


Fig. 7 Illustration of hybrid model

图 7 嵌合模块示意图

预计算模块生成和存储推理所需的张量信息, 主要包括节点的输入特征张量和边的稀疏张量, 张

量信息需存储全局共享，避免重复计算，减少时间和空间开销.

搜索模块将当前搜索状态的匹配节点对集合信息、候选子状态信息进行提取，输入至神经网络预测模块，并根据预测结果，从候选子状态中选出符合要求的子状态继续搜索. 搜索模块根据 VF3 算法的搜索模块改造而来，改造后的搜索模块伪代码如算法 3 所示. 其中 *predict* 为神经网络预测模块.

算法 3. 嵌合神经网络的子图匹配算法 *ml_search*(*s*).

输入：状态 *s*；

输出：匹配集合 *M*.

① $M\leftarrow\varnothing$；

② if 当前状态达成完成匹配子图 then

③ return {状态 *s* 中的匹配关系 *m*}；

④ end if

⑤ if 当前状态不符合匹配规则 then

⑥ return ∅；

⑦ end if

⑧ 生成候选集合 *C* 和当前查询节点 *v*；

⑨ 有效候选集合 $P\leftarrow predict(s, C, v)$；

⑩ for *u* in *P* then

⑪ $s'\leftarrow\{v, u, s\}$；

⑫ $M'\leftarrow ml_search(s')$；

⑬ $M\leftarrow M\cup M'$；

⑭ end for

⑮ return *M*.

神经网络预测模块根据预计算模块提供的全图信息和搜索模块提供的搜索状态信息进行分析和预测，在嵌合算法运行的过程中，模型仅执行推理任务，不进行梯度计算和权重更新，以降低时间成本.

3 实验验证

本文实验主要探究本文所提算法的合理性，并与传统回溯式算法进行对比分析，主要分为 4 个部分：

1）候选集预测模型准确性分析. 与基准算法比较准确率等指标，分析本文预测模型的预测效果.

2）图卷积神经网络的层数分析. 对比不同层数的图卷积神经网络的预测效果，分析层数对预测能力的影响.

3）嵌合算法的效果分析. 通过在真实目标图上的运行实例，对本文嵌合算法的剪枝效果进行量化和可视化分析.

4）采样数量分析. 对比不同数据集样本数量需求，分析所采集的样本数量与图结构的关系.

3.1 实验设置

本研究的实验测试使用 Ubuntu 20 系统，算法设计与实现基于 PyTorch 框架，图的存储和查询等操作的实现基于 Networkx 图数据处理辅助软件包. 神经网络模型的训练过程中使用内存为 32 GB 的 NVIDIA V100 GPU.

神经网络模型在 Yeast 数据集和 DBLP 数据集上单次训练分别迭代 200 次和 100 次.

模型训练过程中，学习率变化策略采用预热策略和指数递降策略相结合的方式. 具体而言，前 10 个迭代轮次为预热启动轮次，学习率由 0 上升至最大学习率 0.000 4，以避免模型陷入局部最优解. 后续迭代轮次为指数递降轮次，在每个轮次中，学习率将递减以防止模型训练后期发生强烈抖动.

3.2 数据集

本研究采用 DBLP 人物关系数据集和酵母蛋白关系数据集 Yeast 两个数据集开展实验. 本文利用 Sun 等人[14] 整理的部分数据生成本文所需数据集，每个数据集包含节点数量为 4~32 的不同查询图. 2 个数据集中图数据的基本信息如表 2 所示.

Table 2 Information Table of Datasets

表 2 数据集信息表

数据集	领域	点数	边数	类别数	点平均度数
DBLP	社交网络	317 080	1 049 866	15	6.6
Yeast	生物蛋白	3 112	12 519	71	8.0

3.3 预测模型训练效果

在 VF3 算法中，节点匹配概率是节点的搜索顺序安排的依据，该算法对根据查询图中节点的度数、标签信息生成每一个节点的匹配概率值，并按照匹配概率值从小到大对查询图中的节点展开搜索. 这使得算法可以优先考虑匹配可能性较小的节点，在搜索过程的早期缩减搜索树的宽度，减少大量无意义的搜索. 为衡量本文剪枝概率预测效果，本文利用 VF3 算法中的匹配概率生成算法，对目标图的每一个节点生成匹配概率，然后将所有节点的概率进行归一化操作作为评价基准. 匹配概率值的计算为：

$$P_{\mathrm{fea}}(u)=P_{\mathrm{lab}}(\lambda(u))\sum_{d'\geqslant d^{\mathrm{in}}(u)}P_{\mathrm{deg}}^{\mathrm{in}}(d')\times\sum_{d'\geqslant d^{\mathrm{out}}(u)}P_{\mathrm{deg}}^{\mathrm{out}}(d'),\quad(22)$$

其中 u 是查询图中的节点，$d^{\mathrm{in}}(u)$, $d^{\mathrm{out}}(u)$, $\lambda(u)$ 分别为节点 u 的入度、出度和标签，$P_{\mathrm{lab}}(\lambda(u))$为目标图中出现与 u 相同标签的概率，$P_{\mathrm{deg}}^{\mathrm{in}}(d')$, $P_{\mathrm{deg}}^{\mathrm{out}}(d')$分别为目标图中出现入度、出度为 d 的节点的概率.

表 3 是本研究中对基准算法和本文算法进行准确率和 *F*1 分数测试的测试结果对照表. 其中，基准算法属于传统算法，重复运行结果不变；本文算法结合了神经网络模型，在训练过程中由于随机种子的变化，其结果有一定变化，本文中取不同随机种子进行 5 次实验后分别计算准确率与 *F*1 分数的平均值.

Table 3 Comparison of Prediction Results

表 3 预测结果对比

数据集	VF3 匹配概率算法		本文算法	
	准确率	*F*1 分数	准确率	*F*1 分数
Yeast	0.539 7	**0.699 7**	**0.657 8**	0.655 4
DBLP	0.589 6	0.738 2	**0.702 8**	**0.772 9**

注：黑体数值表示该数据集下该项评分的最优值.

由表 3 可知，本文算法在 2 个数据集上的准确率均明显高于基准算法，这表明本文模型的判断更加准确. 在 DBLP 数据集上，*F*1 分数有明显提高；在 Yeast 数据集上，由于基准算法的召回率接近 1，本文算法相比之下较低.

3.4 图卷积神经网络层数分析

图神经网络的层数对预测效果具有一定影响，本文使用 2~4 层图卷积神经网络对 Yeast 数据集进行测试，预测准确率与 *F*1 分数如表 4 所示. 除图卷积神经网络层数外，其他实验设置细节与 3.3 节中实验相同.

Table 4 Prediction Results under Different GCN Layers

表 4 不同图卷积神经网络层数下的预测结果

GCN 层数	准确率	*F*1 分数
2	**0.657 8**	**0.655 4**
3	0.631 2	0.653 7
4	0.607 4	0.646 4

注：黑体数值表示该层数下该项评分的最优值.

表 4 表明，使用 2 层图卷积神经网络效果最优，更多的图卷积神经网络层使得预测效果下降. 虽然更多的图卷积神经网络层使得模型可以捕捉更大范围的细粒度局部结构信息，但由于图卷积神经网络自身的结构限制，过多的堆叠层数将带来过平滑问题，使得模型对点之间的区分能力和点的特征表达能力进一步下降，最终导致模型预测能力下降.

3.5 嵌合算法效果分析

本文对 VF3 算法与本文算法的搜索过程进行对比，通过量化和可视化分析，指出 VF3 算法的局限性和本文算法的优越性. 具体而言，在搜索过程中，通过对回溯部分的改造，标记每一个状态是否真正可以得到可行解. 统计总状态数和具有可行解的有效状态数，计算总状态数量与有效状态数之差作为无效状态数. 在本文的剪枝效果测试与讨论中，以无效状态数作为剪枝效果的衡量标准，无效状态数越少，则算法剪枝效果越好.

在一些标签和度数区分度不高、单节点候选集较大的查询图中，VF3 的搜索过程仍有较大改进空间. 1 组无效状态多的查询图示例如图 8 所示，在这组示例中，VF3 算法的搜索过程中产生了大量无效状态，使得 VF3 算法在这些示例中性能较差.

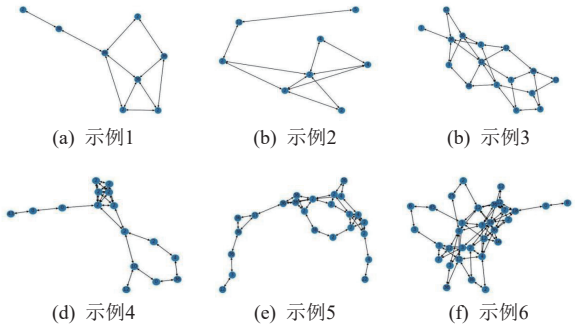


Fig. 8 Query graphs examples with large amount of useless search space

图 8 无效状态较多的查询图示例

这是由于 VF3 算法在筛选候选集时只对候选节点 2 层邻域中的点度数信息进行粗略校验，并不能对邻域内边的连接状态进行细致甄别. 在点度数相对平均的查询图中，2 层粗略邻域信息不足以使算法做出有效判断.

图 8 所示的本组查询图示例选自 Yeast 数据集，使用 VF3 算法和本文算法分别进行子图匹配搜索，搜索结果如表 5 所示. VF3 算法是传统算法，其找到的可行解数即为全部可行解数. 在该组查询图中，VF3 算法在搜索过程中总状态数与有效状态数相差较大，即无效状态数量较大，表明 VF3 算法在这组查

Table 5 Comparison Table of Pruning Effect

表 5 剪枝效果对比表

示例编号	有效状态数	VF3 算法			本文算法		
		解数	总状态数	无效状态数	解数	总状态数	无效状态数
1	68	44	1 682	1 614	44	910	842
2	39	11	763	724	11	596	557
3	217	34	786	569	34	342	125
4	6 321	1 536	101 882	95 561	1 536	32 272	25 951
5	358	86	5 290	4 932	86	4 127	3 769
6	34	2	211	177	2	140	106

询图中性能有较大上升空间. 而本文提出的嵌合算法能找到的可行解数与 VF3 算法相同，即也找到了全部可行解，但本文算法的总状态数和无效状态数显著少于 VF3 算法，表明本文算法在这些查询图中具有较好的剪枝效果.

从具体例子来看，在图 8 的示例 1 中，VF3 算法搜索过程中共搜索得到 44 个解，总搜索状态达 1 682 个，但其中有效搜索状态仅为 68 个，无效状态数为 1 614 个，VF3 算法在大量不产生可行解的状态上进行搜索，效率极低. 本文算法也可以得到 44 个解，但全过程仅需经过 910 个状态，其中 68 个为有效搜索状态，842 个为无效状态. 可见本文算法能较好地减少无效状态的产生，提高搜索效率. 在该组其余 5 个查询图示例中，也能够发现类似现象. 可见 VF3 算法中经过大量无效状态，其搜索过程具有较大提升空间，而本文算法可以显著减少无效搜索状态，具有较强的剪枝能力.

为了更加直观地展示本文设计的嵌合算法的剪枝效果，本文对搜索过程进行了可视化，图 9 展示了对图 8 所示的示例中所有查询图进行搜索过程中的候选节点分布图(局部). 图 9 中的节点均为 VF3 搜索过程中所到达过的目标图中的节点，其中红色节点为有效节点，绿色和黄色节点为无效节点. 黄色节点为使用本文算法成功进行剪枝而跳过的无效节点，绿色节点为使用本文算法但依然未能跳过的无效节点，是本文算法真正经过的无效节点. 黄色节点越多说明本文算法较 VF3 算法相比，节约的搜索状态越多，本文算法越具有优势. 可以直观地看出，VF3 算法在这一组查询图中表现不佳，经过大量无效节点，造成效率下降. 而本文算法对大量无效状态进行了剪枝，有效提高了搜索效率.

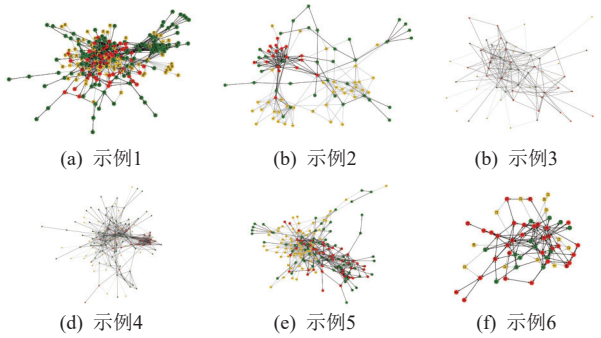


Fig. 9 Comparison of search nodes

图 9 搜索节点对比

从具体例子来看，图 8 中所展示的示例 1，其搜索状态图(局部)对应为图 9 中示例 1，图中红色节点少而稀疏，绿色和黄色节点数量之和明显多于红色节点，这说明 VF3 算法在搜索过程中到达了大量无效节点，产生了大量无意义的搜索. 而其中大量无效状态为黄色，表明本文算法可以对搜索过程中大量的无效节点进行剪枝，最终本文算法真正经过的无效节点(绿色)明显减少. 在该组其余 5 个示例中，也表现出了相似的情况，表明相较于 VF3 算法，本文算法可以裁剪掉大量无效节点

同时，我们还对本文算法产生解质量不高的查询图做出了分析和探究. 在一些匹配数量较大的查询图中，虽然本文算法所找到的解的数量显著低于 VF3 算法，但仍具有一定现实意义. VF3 算法难以在短时间内运行结束，而本文算法则可以在较短时间内，以牺牲一部分可行解为代价完成运行. 在实际生产环境下，可行解数量可能十分庞大，但往往并不需要精确找出全部可行解，而是希望获得达到一定数量的解即可. 因而在这种情况下，耗费大量时间找出全部可行解性价比不高，而能够在短时间内找出相当大一部分可行解也是具有应用价值的.

图 10 展示了 1 组运行时间长的查询图，其可行解过多，VF3 算法运行并找出所有的解需要的时间过长，而利用本文算法可以在相对较短的时间内找出数量较为可观的可行解. 图 10 中的 3 个示例分别选自 VF3 算法的运行时间在 3 个不同数量级时的情况，表 6 是对图 10 中查询图的测试结果统计.

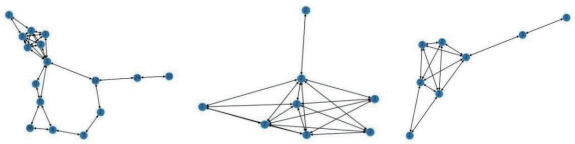


Fig. 10 Query graph examples of long running time

图 10 运行时间较长的查询图示例

Table 6 Comparison Table of Pruning Effect

表 6 剪枝效果对比表

示例编号	点数	边数	VF3 算法			本文算法		
			解数	总状态数	运行时间/s	解数	总状态数	运行时间/s
1	16	28	15 744	1 882 440	1 130.85	672	52 680	213.78
2	8	19	709 138	1 136 551	188. 21	450	9 766	35.86
3	8	14	228 176	431 467	44.78	10	1 435	5.48

具体而言，在图 10 的示例 1 是一张具有 16 个节点、27 条边的查询图，目标图为 Yeast 数据集中的目标图，其总可行解数量为 15 744 个. VF3 算法需要 1 130.85 s 完成运行，而加入剪枝优化后的算法可以

在213.78 s完成运行,搜索得到672个解.虽然损失大量可行解,但在一部分现实问题中,该数量级的部分可行解亦可以满足所需,同时运行时间下降了一个数量级.其他2组示例也展示出相似情况,在一些可行解数量巨大的查询图中,虽然嵌合算法的使用使得求出的可行解数量减少,但可以在求得一部分解的情况下极大降低运行时间,这在一些对匹配精确度要求不高的场景中具有一定应用价值.

3.6 采样数量分析

综合考虑训练时间和训练效果,本文在Yeast和DBLP数据集中分别选取200张和100张查询图用于样本数据的生成.

在候选集预测模型中,将样本按照7∶1∶2的数量比划分为训练集、测试集和验证集.过少的样本数据易产生过拟合现象;过多的样本数据训练时间、空间成本高.经过实验测试,平衡训练效果和训练成本,最终采用的生成样本数量的详细数据如表7所示,其中查询图节点数量分为4, 8, 16, 24, 32这5种.

Table 7 Amount of Samples

表7 样本数量

数据集	样本状态数量	训练样本数	测试样本数	验证样本数
Yeast	6 174	4 322	618	1 235
DBLP	2 154	1 508	215	431

观察表7中的样本数量,结合表2中的数据集基本信息,可以发现虽然DBLP目标图的大小大于Yeast目标图,但在DBLP数据集上所需总样本数量比Yeast数据集更少.产生该现象的原因是,DBLP数据集中节点的标签类别数量较多,节点之间的差异性更为显著,更利于神经网络模型分辨和学习不同标签信息以及图的结构信息,因而在DBLP数据集上训练所需的样本数量并未随着其节点数的大幅增加而增加.

4 结论与展望

本文研究将传统子图匹配算法与神经网络模型相结合,提出了一种高效、准确、自适应的融合图神经网络的高效子图匹配算法.该算法通过图神经网络捕获目标图的细粒度结构特征,并构建全局结构关联,利用模型推理代替复杂扩展以快速估测搜索状态的剪枝概率,为回溯式子图匹配算法提供更加精准的剪枝预测.同时,设计了一套面向回溯式子图匹配算法的采样机制,能够较好地缓解样本不均衡问题,以保证模型稳定学习.最终,本文研究将候选集预测模型与传统回溯式算法相结合,构成嵌合优化算法.实验表明,相比于原回溯式算法,本文算法能够在具有广泛复杂结构的图上进行高效搜索,展现出良好的剪枝效果.

未来的工作可以从3个方面展开:首先,在采样机制的探究中,还需尝试从理论层面归纳和总结样本设计理论,以及样本规模与模型训练效果之间的规律.其次,在剪枝模型设计中,准确率仍有提升空间,可以通过提取回溯式搜索算法中的搜索顺序信息加以改善.也就是,引入时序信息,并利用长短期记忆网络帮助模型获得前驱搜索状态信息,使得模型可以更为准确地对子状态剪枝情况进行预测.最后,在预测模型和传统算法的嵌合上,可以进一步考虑传统算法的数据结构特点和搜索顺序,依据搜索顺序对部分需要多次计算或存在包含关系的数据使用合理的数据结构进行存储,减少重复计算、重复存储等问题,进一步提高时间和空间效率.

作者贡献声明:薛欣完成了论文整体算法设计、实验验证和论文撰写;朱天晨对研究思路进行优化并撰写论文;孙庆赟对研究思路做出了调整与优化;周号益对研究方法提出了具体建议;李建欣总体把控研究方向并提供指导性建议.

参 考 文 献

[1] Snijders T, Pattison P, Robins G, et al. New specifications for exponential random graph models[J]. Sociological Methodology, 2006, 36(1): 99−153

[2] Xue Jiawei, Jiang Nan, Liang Senwei, et al. Quantifying spatial homogeneity of urban road networks via graph neural networks[J]. Nature Machine Intelligence, 2022, 4(3): 246−257

[3] Przulj N, Corneil D, Jurisica I. Efficient estimation of graphlet frequency distributions in protein-protein interaction networks[J]. Bioinformatics, 2006, 22(8): 974−980

[4] Guo Fangfang, Wang Xinyue, Wang Huiqiang, et al. A dynamic stain analysis method on maximal frequent sub graph minin[J]. Journal of Computer Research and Development, 2020, 57(3): 631−638 (in Chinese)
(郭方方, 王欣悦, 王慧强, 等. 基于最大频繁子图挖掘的动态污点分析方法[J]. 计算机研究与发展, 2020, 57(3): 631−638)

[5] Webber J. A programmatic introduction to Neo4j[C]//Proc of the 2012 ACM Conf on Systems, Programming, and Applications: Software for Humanity. New York: ACM, 2018: 217−218

[6] Anikin D, Borisenko O, Nedumov Y. Labeled property graphs: SQL or NoSQL[C]//Proc of the 2019 Ivannikov Memorial Workshop. Piscataway, NJ: IEEE, 2019: 7−13

[7] Singh R, Xu Jinbo, Berger B. Pairwise global alignment of protein

interaction networks by matching neighborhood topology[C]//Proc of the 11th Annual Int Conf on Research in Computational Molecular Biology. New York: ACM, 2007: 16−31

[8] Zhang Xiaolin, Yuan Haochen, Li Zhuolin, et al. Distributed subgraph matching privacy preserving method for dynamic social network[C]// Proc of the 7th CCF Conf of BigData. Berlin: Springer, 2019: 159−173

[9] Nguyen H, Jeong H. An area partitioning and subgraph growing (APSG) approach to the conflation of road networks[J]. Sensors, 2022, 22(4): 1501−1526

[10] Pavlov D, Shturts I. Chemical substructure search screening with fingerprints built with subGraph enumeration[J]. Reviews on Advanced Materials Science, 2009, 20: 37−41

[11] Conte D, Foggia P, Sansone C, et al. Thirty years of graph matching in pattern recognition[J]. International Journal of Pattern Recognition and Artificial Intelligence, 2004, 18(3): 265−298

[12] Nakai, Kenta. Yeast : UCI Machine learning repository[DB/OL]. 1996[2023-03-01]. https://archive.ics.uci.edu/ml/datasets/Yeast

[13] Yang J, Leskovec J. Defining and evaluating network communities based on ground-truth[C]//Proc of the 12th IEEE Int Conf on Data Mining. Piscataway, NJ: IEEE, 2012: 745−754

[14] Sun Shixuan, Luo Qiong. In-memory subgraph matching: An in-depth study[C]//Proc of the 2020 Int Conf on Management of Data. New York: ACM, 2020: 1083−1098

[15] Cordella L, Foggia P, Sansone C, et al. A (sub)graph isomorphism algorithm for matching large graphs[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004, 26(10): 1367−1372

[16] Carletti V, Foggia P, Vento M. VF2 plus: An improved version of VF2 for biological graphs[C]//Proc of the 10th Int Workshop on Graph Based Representations in Pattern Recognition. Berlin: Springer, 2015: 168−177

[17] Juttner A, Madarasi P. VF2++ —An improved subgraph isomorphism algorithm[J]. Discrete Applied Mathematics, 2018, 242: 69−81

[18] Vinyals O, Fortunato M, Jaitly N. Pointer networks[C]//Proc of the 28th Annual Conf on Neural Information Processing Systems. New York: ACM, 2015: 2692−2700

[19] Liu Xin, Pan Haojie, He Mutian, et al. Neural subgraph isomorphism counting[C]//Proc of the 26th ACM Int Conf on Knowledge Discovery Data Mining. New York: ACM, 2019: 1959−1969

[20] Shang Haichuan, Zhang Ying, Lin Xuemin, et al. Taming verification hardness: An efficient algorithm for testing subgraph isomorphism[J]. Very Large Data Base Endowment, 2008, 1(1): 364−375

[21] Abdelaziz I, Al-Harbi R, Salihoglu S, et al. SPARTex: A vertex-centric framework for RDF data analytics[J]. Very Large Data Base Endowment, 2015, 8(12): 1880−1883

[22] Ngo H, Porat E, Re C, et al. Worst-case optimal join algorithms[J]. Journal of ACM, 2018, 65(3): 16: 1−16: 40

[23] Carletti V, Foggia P, Saggese A, et al. Challenging the time complexity of exact subgraph isomorphism for huge and dense graphs with VF3[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2018, 40(4): 804−818

[24] Aberger C, Lamb A, Tu S, et al. EmptyHeaded: A relational engine for graph processing[C]//Proc of the 2016 Int Conf on Management of Data. New York: ACM, 2015: 431−446

[25] Aref M, Cate B, Green T, et al. Design and implementation of the LogicBlox system[C]//Proc of the 2015 ACM Int Conf on Management of Data. New York: ACM, 2015: 1371−1382

[26] Scarselli F, Gori M, Tsoi A, et al. The graph neural network mode[J]. IEEE Transactions on Neural Networks, 2009, 20(1): 61−80

[27] Kipf T, Welling M. Semi-supervised classifification with graph convolutional networks[J]. arXiv preprint, arXiv: 1609.02907, 2016

[28] Schlichtkrull M, Kipf T, Bloem P, et al. Modeling relational data with graph convolutional networks[C]//Proc of the 15th Int Conf of ESWC. Berlin: Springer, 2018: 593−607

[29] Yan Sijie, Xiong Yuanjun, Lin Dahua. Spatial temporal graph convolutional networks for skeleton-based action recognition[C]// Proc of the 8th AAAI Conf on Artificial Intelligence. New York: ACM, 2018: 7444−7452

[30] Hamilton W, Ying R, Leskovec J. Inductive representation learning on large graphs[C]//Proc of the 31st Int Conf on Neural Information Processing Systems. New York: ACM, 2017: 1025−1035

[31] Xu Keyulu, Hu Weihua, Leskovec J, et al. How powerful are graph neural networks[J]. arXiv preprint, arXiv: 1810.00826, 2018

[32] Velickovic P, Cucurull G, Casanova A, et al. Graph attention networks[J]. arXiv preprint, arXiv: 1710.10903, 2017

[33] Wang Shuohang, Jiang Jing. Machine comprehension using Match-LSTM and answer pointer[J]. arXiv preprint, arXiv: 1608.0790, 2016

[34] Ma Qiang, Ge Suwen, He Danyang, et al. Combinatorial optimization by graph pointer networks and hierarchical reinforcement learning[J]. arXiv preprint, arXiv: 1911.04936, 2019

[35] Khalil E, Dai H, Zhang Y, et al. Learning combinatorial optimization algorithms over graphs[C]//Proc of the 30th Annual Conf on Neural Information Processing System. La Jolla, CA: NIPS, 2017, 6348−6358

[36] Wang Hanchen, Zhang Ying, Qin Lu, et al. Reinforcement learning based query vertex ordering model for subgraph matching[C]//Proc of the 38th Int Conf on Data Engineering. Piscataway, NJ: IEEE, 2022: 245−258

[37] Ying R, Lou Zhaoyu, You Jiaxuan, et al. Neural subgraph matching[J]. arXiv preprint, arXiv: 2007.03092, 2020

[38] Roy I, Velugoti V, Chakrabarti S, et al. Interpretable neural subgraph matching for graph retrieval[C]//Proc of the 36th AAAI Conf on Artificial Intelligence. New York: ACM, 2022: 8115−8123

[39] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Proc of the 30th Annual Conf on Neural Information Processing Systems. New York: ACM, 2017: 5998−6008

[40] Devlin J, Chang Mingwei, Lee K, et al. BERT: Pre-training of deep bidirectional transformers for language understanding[C]//Proc of the 2019 Conf of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Stroudsburg, PA: ACL, 2019: 4171−4186

[41] Beltagy I, Peters M, Cohan A. Longformer: The long-document transformer[J]. arXiv preprint, arXiv: 2004.05150, 2020

[42] Dai Zihang, Yang Zhilin, Yang Yiming, et al. Transformer-XL: Attentive language models beyond a fixed-length context[C]//Proc of the 57th Annual Meeting of the Association for Computational Linguistics. Stroudsburg, PA: ACL, 2019: 2978−2988

[43] Wang Wenhai, Xie Enze, Li Xiang, et al. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions[C]// Proc of the 2021 IEEE/CVF Int Conf on Computer Vision. Piscataway, NJ: IEEE, 2021: 548−558



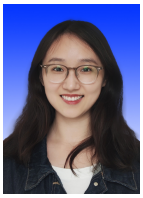
Xue Xin, born in 2001. PhD candidate. Student member of CCF. Her main research interests include graph neural network and combinatorial optimization problem.

薛　欣, 2001 年生. 博士研究生. CCF 学生会员. 主要研究方向为图神经网络、组合优化问题.



Zhu Tianchen, born in 1996. PhD. Member of CCF. His main research interest includes decision-making in time series.

朱天晨, 1996 年生. 博士. CCF 会员. 主要研究方向为时序决策.



Sun Qingyun, born in 1996. PhD, assistant professor. Member of CCF. Her main research interests include graph machine learning, network big data analysis, and anomaly detection.

孙庆赟, 1996 年生. 博士, 助理教授. CCF 会员. 主要研究方向为图机器学习、网络大数据分析、异常检测.



Zhou Haoyi, born in 1991. PhD, assistant professor. Member of CCF. His main research interests include time-series forecasting, task decision-making, and AI safety.

周号益, 1991 年生. 博士, 助理教授. CCF 会员. 主要研究方向为时间序列预测、任务决策、人工智能安全.



Li Jianxin, born in 1979. PhD, professor. Distinguished member of CCF. His main research interests include big data analysis and processing, network big data computing, and scientific computing.

李建欣, 1979 年生. 博士, 教授.CCF 杰出会员. 主要研究方向为大数据分析与处理、网络大数据计算、科学计算.