第 5 章 图神经网络进阶

引言

图神经网络作为利用深度学习处理图结构数据的前沿工具，在捕捉节点关系、结构信息和非欧几里得特性方面展现了强大的能力。然而，由于图数据和应用场景的复杂化，传统图神经网络模型在性能和适用性上暴露出了一些瓶颈，往往受到数据质量、模型架构和训练策略的限制。随着图神经网络的发展，越来越多的先进模型被提出以突破传统图神经网络的瓶颈。本章从数据、架构和训练三个层面介绍了针对传统图神经网络的优化策略。在数据层面，介绍如何提升数据质量，挖掘图数据潜在的高质量特征，减少噪声干扰；在架构层面，介绍如何改进传统图神经网络主要部件，包括信息传递，采样和池化，以更好地捕捉图数据的复杂关系，提升模型性能；在训练层面，介绍如何改进图神经网络的训练策略，包括图自监督学习和图课程学习，从而缓解依赖大量标注进行训练的困境，并且实现更稳健的模型收敛和更高的泛化能力。

本章学习目标

(1) 理解不同图数据优化技术的目的，掌握从结构、特征和标签三个方面对图数据进行优化的主要方法；

(2) 理解传统图神经网络在消息传递方面存在的主要问题，掌握针对过平滑、长距离依赖，表达能力受限问题的主要解决方法；

(3) 掌握图上的采样和池化的主要改进方法，并能根据实际任务灵活选择；

(4) 掌握在缺乏标签情况下设计对比或生成式图自监督训练的方法；

(5) 了解图课程学习的主要思想，学会利用图课程学习来稳定训练和提升模型效果。

5.1 数据优化

在机器学习的发展历程中，丰富的实践经验表明，大量高质量数据是提升模型性能、推动模型进步的关键因素。例如，大规模视觉数据集 ImageNet 的引入，催生出了经典的卷积神经网络模型 AlexNet 和 ResNet，奠定了卷积神经网络在图像处理领域的主导地位。同样，在图学习领域，大量研究工作聚焦于如何改进图数据质量，以便图神经网络能够更好地捕捉图数据中的信息。这些研究主要从图结构优化、图特征优化和图标签优化三个方面入手，在提升图神经网络模型的准确性、鲁棒性和效率方面发挥了重要作用。

5.1.1 图结构优化

图结构是图数据中最核心的部分，它描述了节点之间的关联信息。在本节中，将探讨如何从图结构的角度优化图数据。首先介绍结构缩减，旨在减少图中的冗余节点和边，以降低计算复杂度并提高模型的可扩展性；其次讨论结构增强，通过较低的开销丰富图结构信息，从而缓解模型过拟合的问题；接下来，阐述结构生成，其目标是生成高质量且多样化的图样本；最后介绍结构学习，专注于从图数据中挖掘出有价值的图结构，进一步提升图模型的表达能力。

1. 结构缩减

近年来，图数据集的规模和复杂性呈现出指数级增长。对于大规模网络（如社交图和引文网络），现有图神经网络在可扩展性和效率方面面临着严峻挑战。结构缩减技术在保留关键信息的前提下，通过减少图数据集的规模，降低计算复杂度并提升模型可扩展性。结构缩减方法可以分为三类：

1) 图稀疏化

图稀疏化通过移除原始图G中的部分边，生成一个简化图$G_s = \{V, E_s\}$,其中$E_s \subseteq E$。通常G_s

需要保持原图G的某些关键性质，如图切割值（Cut Value）、最短路径（Shortest Path）等。割稀疏化（Cut Sparsification）是图稀疏化的一个典型例子。它的目标是在尽可能保持图切割值的前提下，通过减少图中的边来简化图结构。图的切割将图的节点分为两部分，可以表示为$C = (V', V - V')$。在此过程中，图切割值（Cut Value，记作$w(C)$）是指跨越该切割的所有边的权值和，这个值反映了两部分节点之间的连接强度。通常，ϵ-割稀疏化（ϵ-cut Sparsifier）是指一个简化图G_s，其图切割值在所有切割的情况下与原始图G的图切割值保持接近，表示如下：

$$(1-\epsilon)w_G(C) \le w_{G_s}(C) \le (1+\epsilon)w_G(C)$$

其中ϵ的取值范围为(0,1)，当$\epsilon = 0$ 时，G_s完全等同于原始图G，随着ϵ增大，允许的偏差更多，但简化图G_s在图切割值上依然与原始图G保持接近。割稀疏化也广泛应用于解决图的连通性问题、最大流问题、最小二等分问题等。

2) 图粗化

图粗化通过将一组紧密连接的节点合并成超级节点的方法来简化图结构。简化图G_s可以记为$G_s = (V_s, E_s)$，其中节点数$|V_s| < |V|$。近年来，谱保持的图粗化方法（Spectrum-preserving Coarsening）因为可以较好地保持原始图的重要结构信息受到了较多关注。受限谱相似性（Restricted Spectral Similarity, RSS）[1]是一种用来确保简化图能够学习到原始图谱特性的技术，具体可以定义为：

$$(1-\zeta_k)\lambda_k \le \boldsymbol{u}_k^T \tilde{\boldsymbol{L}} \boldsymbol{u}_k \le (1+\zeta_k)\lambda_k$$

其中，λ_k和$\boldsymbol{u}_k$分别表示原始图G的拉普拉斯矩阵$\boldsymbol{L}$的第k个特征值和特征向量，$\tilde{\boldsymbol{L}} \in \mathbb{R}^{N\times N}$是简化图$G_s$拉普拉斯矩阵的近似，$\zeta_k$是误差容忍度，通常取值范围为（0,1）。受限谱相似性方法通过确保简化图的拉普拉斯矩阵特征值和特征向量在一定误差范围内接近原始图，从而确保简化图能有效地学习和保留原始图的谱特性。

3) 图压缩

不同于图稀疏化和图粗化在原始图上进行结构上的缩减，图压缩通过合成一个新的更小的图来实现原始图的压缩。图压缩的目的是生成一个包含较少节点和边的简化图$G_s = \{V_s, E_s\}$，其中$|V_s| \ll |V|$，使得在简化图上训练的模型能表现出与原始图上训练的模型相似的性能。经典的图压缩框架是利用梯度匹配的方法对齐原始图和简化图的梯度[2]。具体来说，它通过最小化梯度之间的距离使得在简化图上训练模型时，模型的参数更新与在原始图上训练时相似，可以表示如下：

$$\boldsymbol{r}_s = \nabla_\theta L(f_\theta(\boldsymbol{A}_s, \boldsymbol{X}_s), \boldsymbol{Y}_s),$$

$$\boldsymbol{r} = \nabla_\theta L(f_\theta(\boldsymbol{A}, \boldsymbol{X}), \boldsymbol{Y}),$$

$$\min \mathrm{Dis}(\boldsymbol{r}_s, \boldsymbol{r}).$$

其中，f_θ表示图模型，θ是模型参数，模型的输入是图的邻接矩阵$\boldsymbol{A}$和节点特征矩阵$\boldsymbol{X}$，L是损失函数，$\boldsymbol{r}_s$是在简化图G_s上计算得到的梯度，$\boldsymbol{r}$是在原始图G上计算得到的梯度，$\mathrm{Dis}(\boldsymbol{r}_s, \boldsymbol{r})$是衡量两个梯度距离的函数，比如余弦相似度。通过最小化此距离，使得简化图保留了原始图的结构和特征信息。

2. 结构增强

模型通常需要大量数据才能有效地理解数据的特征和规律。然而，由于图数据的稀缺性和稀疏性，图神经网络在训练过程中往往无法充分拟合图数据的底层分布，容易陷入局部最优解，导致模型过拟合，严重削弱了模型在实际应用中的有效性和可靠性。为了缓解这一问题，结构增强方法在不改变图关键信息的前提下对图的拓扑结构进行适当的扰动，以一种低开销的方式增强了拓扑结构信息，有效提高了模型的泛化能力。结构增强可分为启发式和自适应增强方法。

启发式增强方法是一类通过预定义规则或经验策略对图结构进行修改或扩展的技术，其核

心思想是利用简单、高效的操作提升模型的鲁棒性和泛化能力。这些方法通常与具体任务无关，具有较强的普适性，同时对模型结构无显著依赖，因此容易集成到现有的图神经网络中。

1) 丢弃

丢弃是结构增强中一种基本而广泛应用的技术，旨在通过随机丢弃图中的边、节点和特征等来改善模型的训练效果。这种方法通常不需要对模型结构进行修改，而是在训练过程中动态地对图进行随机裁剪，因此非常易于集成到现有的图神经网络训练流程中。以随机丢弃边的方法为例，DropEdge[3]在每个训练轮次在图的边集合中随机选择$p|E|$条边进行丢弃，其中p表示去边率，经过丢弃后的邻接矩阵可以表示如下：

$$\boldsymbol{A}_{drop} \quad = \quad \boldsymbol{A} - \boldsymbol{A}^{'}$$

其中，$\boldsymbol{A}^{'}$是随机选取的$p|E|$条边组成的稀疏邻接矩阵。DropEdge 没有改变邻居聚合的期望，是一种无偏的图结构增强技术，类似于典型的图像增强操作（如旋转和裁剪），所以可以缓解过拟合问题，增强模型的泛化能力。

2) 子图替换

子图替换是为了弥补基于丢弃的方法仅关注到节点或边等最基本层次的信息，而忽视了更高层次的信息的不足而提出的，它通过替换图中的特定子结构来实现图结构的增强。MoCL[4]是在生物医学领域运用子图替换的方法进行图增强的一个重要的工作，它指出大多数丢弃的方法在增强过程中可能会改变分子图的语义，因此通过注入领域知识来辅助增强过程。MoCL 引入了生物电子等排体（Bioisosteres）的概念来对分子中的特定有效子结构进行替换。生物电子等排体是一类具有相似物理或化学性质的分子片段，替换后不会显著改变分子的整体性质。这种替换能够保持分子的生物活性和物理化学性质，同时引入变化以增强数据多样性。

启发式增强方法还包括图扩散等，这里不做详细介绍。启发式增强方法对于在特定任务上需要增强模型鲁棒性和性能时可能存在不足。自适应增强方法在训练阶段基于具体任务优化自适应地进行结构增强，分为基于边的方法、基于子图的方法，以及自动化增强方法。

1) 基于边的方法

为了使可微损失函数指导边增强过程，一些研究将图上边的权值视为可被优化的连续变量，而不是具有固定的边连接。这些工作通过引入特定的约束（如平滑性和稀疏性）来构建损失函数，从而生成用于优化边权重的梯度。例如，Pro-GNN[5]提出了如下的损失函数：

$$L = \| \widetilde{\boldsymbol{A}} - \boldsymbol{A} \|_F^2 + \eta \| \widetilde{\boldsymbol{A}} \|_1 + \beta \| \widetilde{\boldsymbol{A}} \|_* + \rho\left(\boldsymbol{X}^T \hat{\boldsymbol{L}} \boldsymbol{X}\right) + \gamma L_{GNN},$$

其中，$\widetilde{\boldsymbol{A}}$是增强后的邻接矩阵，$\hat{\boldsymbol{L}}$是归一化的拉普拉斯矩阵。具体来说，$\| \widetilde{\boldsymbol{A}} - \boldsymbol{A} \|_F^2$（$\| \cdot \|_F$代表 Frobenius 范数）旨在让$\widetilde{\boldsymbol{A}}$ 接近原始邻接矩阵A。$\eta \| \widetilde{\boldsymbol{A}} \|_1$ ($\| \cdot \|_1$ 表示 L_1 范数)和$\| \widetilde{\boldsymbol{A}} \|_*$ ($\| \cdot \|_*$ 表示核范数) 分别确保图的稀疏性和低秩特性。此外，$\rho\left(\boldsymbol{X}^T \hat{\boldsymbol{L}} \boldsymbol{X}\right)$控制特征的平滑性。$\gamma$ 控制针对具体任务的图神经网络损失函数L_{GNN}的比重。

2) 基于子图的方法

基于子图的自适应增强方法旨在找到最具代表性和信息量的子图，类似于分子中的官能团。然后基于这些子图来进行数据增强，以提高模型的性能、可解释性和鲁棒性等。GREA[6] 是一个典型的基于子图的增强过程，它定义了核心子图和环境子图的概念，利用核心子图与不同的环境子图的组合生成新的数据样本，以让模型感知到核心子图的重要性，帮助模型学习更丰富的特征和拓扑结构。具体而言，如图 5-1 所示，核心子图（Rationale Subgraph）是指图结构中最能解释或支持模型预测的子结构。环境子图（Environment Subgraph）是指在核心子图被识别并分离后，图中剩余的部分。GREA 首先利用图神经网络生成潜在节点表示，然后通过多层感知机（MLP）计算掩码向量，通过优化属性预测损失来指示哪些节点属于核心子图。接着，它将核心子图与其他样本的环境子图的表示结合，以产生新的增强数据。通过这种方式，图的全局结构得到了丰富，同时保留了与任务相关的关键信息。在模型训练过程中，GREA 的增强数据可以被视为新的输入，并参与优化模型参数。

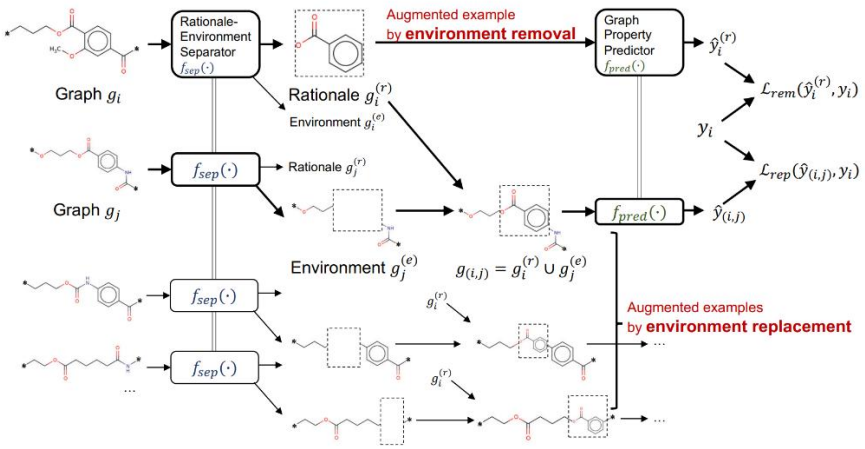


图 5-1 GREA 整体框架图[6]

3) 自动化增强

不同的数据集可能需要不同的增强策略，同一数据集的不同训练阶段也可能需要不同的增强策略。与上面两种固定的增强策略不同，自动化增强是一种在模型训练过程中动态学习最佳增强策略的方法，可以根据数据的特性和任务的需求调整其策略，以提高模型自适应数据分布和任务的能力。这种方法的主要思路是通过双层优化算法或强化学习来选择最合适的增强方式。JOAO[7]方法通过双层优化的框架来同时学习模型的编码器和增强策略，以实现更有效的数据增强。在内层优化过程中，首先使用当前的增强策略对训练数据进行增强，生成新的训练样本。这些增强的数据被用于训练模型的编码器，以提高其性能。在外层优化中，评估编码器性能并根据评估结果调整增强策略。JOAO 从具有可学习参数的分布中采样增强策略（比如基于边和基于子图的方法），通过优化过程自动更新这些参数，以充分发挥每个增强策略的性能。

3. 结构生成

尽管图增强可以初步丰富拓扑信息，但不可避免地会引入噪声，从而影响模型性能。作为一种更高级的方法，图生成旨在生成高质量和多样化的图样本。在这一部分，将针对不同的生成的图数据形式介绍相应的代表性生成方法，包括节点序列的生成、邻接矩阵的生成和节点嵌入的生成。

1) 节点序列生成

将图简化为序列是图生成的一个初步思路，催生了自回归图生成方法。通常，自回归方法旨在基于预先采样的节点顺序逐个生成图的节点。然而，由于图的非唯一性和高维特性，使用节点顺序作为输入时需要考虑置换不变性的问题。为了解决这一挑战，GraphRNN[8] 提出使用广度优先搜索（BFS）或深度优先搜索（DFS）来保证节点顺序的一致性，它的核心思想是将图的生成过程分为节点生成和边生成两个步骤。具体来说，GraphRNN 首先通过 BFS 或 DFS 遍历图，得到节点的序列。这一步使得模型可以根据节点的访问顺序逐步生成图结构，确保生成过程具有唯一性。GraphRNN 在节点生成阶段逐个生成图中的节点。生成的节点数量可以根据图的规模动态调整。每个节点通过一个循环神经网络（Recurrent Neural Network, RNN）生成，该 RNN 根据当前生成的节点序列预测下一个节点。在生成了一个新节点之后，边生成器会预测该节点与已经生成的节点之间的连接情况。GraphRNN 使用另一个 RNN 模型对边的存在性进行逐一预测，判断是否存在边连接到当前已生成的节点。通过这种方式，GraphRNN 将图的生成转化为一个序列化的任务，避免了直接处理图中所有节点和边带来的高计算成本。

2) 邻接矩阵生成

除了序列生成，另一种自然的思路是直接生成图的邻接矩阵。与序列生成的方法不同，这种方法一次性生成整个图的结构信息，尤其适用于小型图的生成。属于这一类别的方法，如 EDGE[9]，它的核心思想是基于离散扩散模型（Discrete Diffusion Model）。扩散模型最早用于连续空间数据的生成（如图像生成），而 EDGE 方法将其扩展到离散空间，应用于图的生成。

它的工作原理主要分为两个阶段：第一阶段是正向扩散过程，逐步向图结构中加入噪声，直到数据接近于均匀噪声；第二阶段是反向去噪过程，从噪声数据开始，通过逐步去噪恢复出图结构信息。另外，EDGE 还为每个节点设计一个目标度向量，通过优化度匹配损失函数来确保生成的图符合预定的节点度分布。

3) 节点嵌入生成

生成图的邻接矩阵通常耗时较长，且无法扩展到大型图。一个可能的解决方案是间接生成图。例如，可以通过节点嵌入来表示邻接矩阵：$\boldsymbol{A} = \boldsymbol{H} \cdot \boldsymbol{H}^{\mathrm{T}}$。通过这种方式，只需生成一个较小的张量$\boldsymbol{H} \in \mathbb{R}^{N \times d}$,而不是大型邻接矩阵$\boldsymbol{A} \in \mathbb{R}^{N \times N}$,其中$d \ll N$。基于这一思想，变分图自编码器（VGAE）[10]首先利用编码器（通常是图神经网络）学习节点表示，之后与标准的变分自编码器（VGA）类似，VGAE 假设节点的潜在表示为服从高斯分布的随机变量，并通过参数矩阵的变换输出其均值和方差，然后利用重参数技巧重新采样潜在表示。解码器从潜在表示中重构图的邻接矩阵,这一步通常通过计算节点潜在表示的内积来预测边的存在概率。在训练过程中，通过优化链接预测损失函数重建图结构。

4. 结构学习

现实世界中的图结构往往噪声较多或不完整，导致模型的学习效果下降，甚至产生错误的结果。这种图结构在社交网络、交通系统、生物网络等多个领域中普遍存在。图结构学习旨在从图数据中发现和优化有价值的结构，以增强图表示学习。根据是否考虑边的权重信息，现有的图结构学习方法大致可以分为两类：离散图结构学习和加权图结构学习。

1) 离散图结构学习

离散图结构学习将图结构视为随机变量，进而可以从概率邻接矩阵中进行采样。这种方法的核心在于利用概率模型捕捉图中节点之间的关系，并通过采样得到多样化的图结构，以建模节点连接关系的潜在不确定性和多样性。在这一框架下，研究人员使用多种技术来联合优化概率邻接矩阵和图神经网络的参数。通过同时优化这两个部分，模型不仅能够更好地学习图中的复杂关系，还能提升其泛化能力。接下来，将介绍一种经典的基于蒙特卡洛方法的离散图结构学习方法。蒙特卡洛方法是一类基于随机抽样的计算方法，常用于数值积分、概率分布的近似、优化等问题。它的核心思想是通过大量随机样本来估计一个期望值或者积分结果，这种方法特别适合高维复杂问题，尤其是当解析解难以获得时。

在离散图结构学习中，VGCNs[11]通过参数化的随机图模型纳入不确定的图信息，并利用蒙特卡洛方法进行近似。具体而言，VGCNs 的目标是通过已知的信息（包括部分已知标签、特征和观察到的图结构）推断节点或图的标签的后验概率，公式如下：

$$p(\boldsymbol{Z}|\boldsymbol{Y}_L, \boldsymbol{X}, G_{obs}) = \int p(\boldsymbol{Z}|\boldsymbol{W}, G, \boldsymbol{X}) p(\boldsymbol{W}|\boldsymbol{Y}_L, \boldsymbol{X}, G) p(G|\lambda) p(\lambda|G_{obs}) d\boldsymbol{W} dG d\lambda$$

其中，$p(\boldsymbol{Z}|\boldsymbol{W}, G, \boldsymbol{X})$表示在给定神经网络权重$\boldsymbol{W}$、图$G$和特征$\boldsymbol{X}$的情况下，模型输出$\boldsymbol{Z}$的条件概率。这个概率可以通过图卷积神经网络（GCN）来建模；$p(\boldsymbol{W}|\boldsymbol{Y}_L, \boldsymbol{X}, G)$表示在已知部分标签$\boldsymbol{Y}_L$，特征$\boldsymbol{X}$和图结构$G$的情况下，GCN 权重$\boldsymbol{W}$的后验概率，这描述了在图神经网络上进行训练时，权重的更新过程；$p(G|\lambda)$表示在给定参数λ的情况下，随机图G的生成概率；$p(\lambda|G_{obs})$表示在给定观察到的图G_{obs}的情况下，参数λ的后验概率，它表示从观察到的图中推断出随机图生成模型的参数。上述积分通常是不可解析的，因此通常需要采用近似方法来进行计算。VGCNs 使用蒙特卡洛方法近似上述积分：

$$p(\boldsymbol{Z}|\boldsymbol{Y}_L, \boldsymbol{X}, G_{obs}) \approx \frac{1}{V} \sum_{v} \frac{V}{N_G S} \sum_{i=1}^{N_G} \sum_{s=1}^{S} p\left(\boldsymbol{Z}|\boldsymbol{W}_{s,i,v}, G_{i,v}, \boldsymbol{X}\right)$$

蒙特卡罗近似通过采样的方法来估计标签$\boldsymbol{Z}$的后验分布$p(\boldsymbol{Z}|\boldsymbol{Y}_L, \boldsymbol{X}, G_{obs})$。首先，从图生成模型的参数分布$p(\lambda|G_{obs})$中生成$V$个样本$\lambda_v$,这些样本代表不同的模型参数；然后，对于每个$\lambda_v$,从条件分布$p(G|\lambda_v)$中生成$N_G$个图样本$G_{i,v}$,反映出在这些参数下可能的图结构；接着，对于每

个图样本$G_{i,v}$，从贝叶斯图卷积神经网络中采样权重矩阵$\boldsymbol{W}_{s,i,v}$；最后，通过对所有样本的加权平均，近似计算出 $p(\boldsymbol{Z}|\boldsymbol{Y}_L, \boldsymbol{X}, G_{obs})$，实现了综合考虑多种可能的图结构和神经网络权重来估计标签的后验概率。

2) 加权图结构学习

离散图结构学习过于依赖已知的的图结构和节点连接模式，缺乏对新节点的适应能力，因此在处理未知或未见节点时表现不佳，在推理阶段面对未见节点时，往往无法有效进行归纳学习。另外，与二元邻接矩阵相比，加权邻接矩阵能够编码更丰富的边的信息，有利于后续图表示学习。加权图结构学习往往假设节点属性或多或少包含推断图的隐性拓扑结构的有用信息，因此将图结构学习作为定义在节点嵌入空间上的相似度量学习，使得学到的相似性度量函数以后可以应用于未见过的节点嵌入集来推断图结构，从而实现归纳图结构学习。

加权图结构学习的核心思想是基于节点嵌入学习节点对的相似性度量函数，得到边的权值，从而实现加权的图结构学习。最简单的度量函数是计算任意一对节点嵌入之间的点积，可以表示为：$\boldsymbol{S}_{i,j} = \vec{\boldsymbol{v}}_i^{\top}\vec{\boldsymbol{v}}_j$ 其中，$\boldsymbol{S} \in \mathbb{R}^{n\times n}$是一个节点相似性矩阵,$\vec{\boldsymbol{v}}_i$和$\vec{\boldsymbol{v}}_i$是节点的向量表示。 为了提高点积的学习能力，引入具有可学习参数的点积：$\boldsymbol{S}_{i,j} = (\vec{\boldsymbol{v}}_i \odot \vec{\boldsymbol{u}})^{\top}\vec{\boldsymbol{v}}_j$， 其中⊙表示逐元素相乘，$\vec{\boldsymbol{u}}$是一个非负的可训练权重向量，用于强调节点嵌入的不同维度。 需要注意的是，输出的相似性矩阵$\boldsymbol{S}$是不对称的。为了进一步提高表达能力，出现了引入权重矩阵的度量方法， $S_{i,j} = \mathrm{ReLU}(\boldsymbol{W}\vec{\boldsymbol{v}}_i)^{\top}\mathrm{ReLU}(\boldsymbol{W}\vec{\boldsymbol{v}}_j)$， 其中$\boldsymbol{W}$是一个$d\times d$的权重矩阵，$\mathrm{ReLU}(\boldsymbol{x}) = \max(0, \boldsymbol{x})$是一种激活函数，在这里用于保证相似性矩阵的稀疏性。之后的方法进一步对两个节点嵌入应用了不同的线性变换，并引入了归一化操作： $\boldsymbol{S}_{i,j} = \mathrm{softmax}\left((\boldsymbol{W}_1\vec{\boldsymbol{v}}_i)^{\top}\boldsymbol{W}_2\vec{\boldsymbol{v}}_j\right)$ ，其中$\boldsymbol{W}_1$和$\boldsymbol{W}_2$是$d\times d$的权重矩阵，softmax 函数定义为 $\mathrm{softmax}(\vec{\boldsymbol{z}})_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$用于获得行归一化的相似性矩阵。

5.1.2 图特征优化

图特征是用于描述图中节点、边或整个图的属性和信息的特征表示。在本节中，将探讨如何从图特征的角度优化图数据，首先介绍特征增强，通过扩展或修改原始特征，避免模型训练时的过拟合；其次，讨论特征选择，旨在识别和提取与标签高度相关的特征，避免维度灾难；最后，介绍特征补全，解决图数据中特征不完整的问题。

1. 特征增强

特征增强通过对节点、边或图的原始特征进行扩展或修改，为图数据引入额外的、相关的信息，不仅能够缓解模型的过拟合，还可以提高模型的泛化性能，特别是在处理复杂图结构或稀疏图数据时表现尤为显著。本小节首先介绍一些通用的特征增强方法，然后介绍一种提高图神经网络表达能力的重要特征增强方法，即位置编码。

1) 通用特征增强

通用特征增强一般用于特征预处理，用于提高特征的效用和多样性，使特征更好捕捉图信息。对于节点本身具有特征的情况，特征损坏（Feature Corruption）通过向原始节点特征中加入可控噪声来产生增广数据，可以表示为$\widetilde{\boldsymbol{X}} = \boldsymbol{X} + \boldsymbol{R}$，其中 X 表示原始节点特征矩阵，$\boldsymbol{R}$表示添加的噪声矩阵；特征重排（Feature Shuffling）通过随机切换特征矩阵中的行和列来改变原始节点特征的上下文信息，产生增广数据，形式化表示为$\widetilde{\boldsymbol{X}} = \boldsymbol{P}_r\boldsymbol{X}\boldsymbol{P}_c$，其中，$\boldsymbol{P}_r$和$\boldsymbol{P}_c$分别是行排列矩阵和列排列矩阵，它们在每一行和每一列中恰好有一个元素为 1，其他位置均为 0；特征掩码（Feature Masking）的核心操作是将节点特征矩阵中的一部分置零，通过与一个 0-1 掩码矩阵$\boldsymbol{M}$逐元素相乘来实现：$\widetilde{\boldsymbol{X}} = \boldsymbol{X} \odot \boldsymbol{M}$；特征添加（Feature Addition）通过将节点特征中缺少的节点属性编码成特征向量并与原始节点特征拼接实现，一般来说，可以表示为$\widetilde{\boldsymbol{x}}_i = [\boldsymbol{x}_i \,\|\, \boldsymbol{x}_j]$，其中‖表示拼接操作，$\boldsymbol{x}_j$可以是一个空向量；特征传播（Feature Propagation）通过在图中扩散特征，结合来自不同节点的特征信息，表达式为：$\widetilde{\boldsymbol{X}} = \widetilde{\boldsymbol{A}}\boldsymbol{X}$，其中$\widetilde{\boldsymbol{A}}$是不同图传播方法对应的邻接矩阵。对于节点本身没有特征的情况，往往通过将图自身的结构信息编码成节点特征，比如直接使用节点的度做为节点特征。更加复杂的方法包括利用随机游走算法来捕获结构信息，并

仿照 word2vec 技术来生成节点特征等。

2) 位置编码

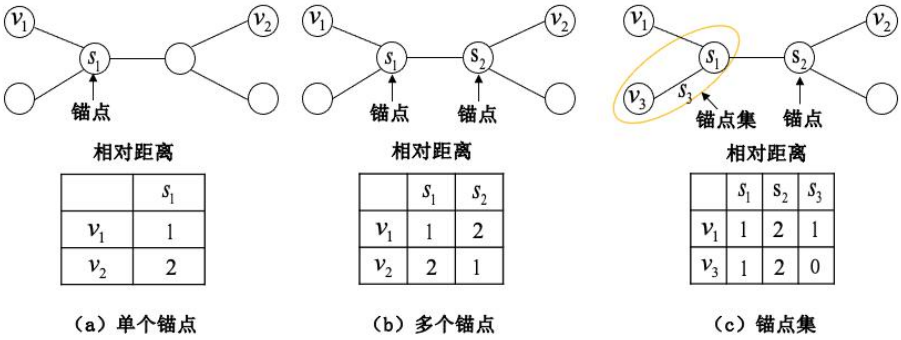
众所周知，图神经网络的表达能力受到 1-WL 测试的限制，其在区分图同构性方面存在一定局限性，仅能够区分部分同构图。为了打破这一限制，一种常见的策略是通过引入位置信息来增强节点特征，即位置编码。这里将介绍两类位置编码方法：绝对位置编码和相对位置编码。

绝对位置编码（Absolute Position Encoding，APE）的目标是为每个节点分配一个位置表示，以指示其在整个图中的唯一位置。一种流行的 APE 方法是利用图拉普拉斯矩阵的特征向量，具体来说，通过对拉普拉斯矩阵$\boldsymbol{\Delta}$进行特征分解，可以得到：

$$\boldsymbol{\Delta} = \boldsymbol{I} - \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2} = \boldsymbol{U}^{T}\boldsymbol{\Lambda}\boldsymbol{U}$$

其中，$\boldsymbol{U}$是特征向量矩阵，$\boldsymbol{\Lambda}$是特征值矩阵，一般来说，选择前k个最小的非零特征值对应的特征向量，形成一个$N \times k$的矩阵，每一行可看作一个节点的位置编码。通过将节点的位置编码与节点的原始特征组合，实现对原始特征的增强。

另外一种位置编码方式是相对位置编码(Relative Position Encoding，RPE)，它通过将节点之间的距离作为位置编码来捕捉它们之间的关系信息。首先随机选择一个节点作为锚点节点，然后通过目标节点与锚点节点之间的相对距离来定位目标节点。如图 5-2(a)所示，使用图神经网络往往无法区分节点v_1和v_2，因为两个节点虽然位置不同，但是所处位置图结构相同。而通过选择$s1$ 作为锚点节点，便可以通过计算和$s1$ 的相对距离（跳数）来区分v_1和v_2。为了更精确捕捉节点的位置信息，可以选择多个锚点节点，如图 5-2(b)所示。PGNN[12]进一步引入锚点集的概念，如图 5-2(c)所示，通过锚点s_1和s_2无法区分节点v_1和v_3，而通过将s_1和v_3组合为锚点集，把目标节点到锚点集中所有节点的最近距离作为相对距离，就可以正确区分v_1和v_3。



	s_1
v_1	1
v_2	2

	s_1	s_2
v_1	1	2
v_2	2	1

	s_1	s_2	s_3
v_1	1	2	1
v_3	1	2	0

图 5-2 相对位置编码

2. 特征选择

当机器学习算法中使用的数据特征维度过高，就会在高维特征空间中呈现出稀疏性，导致需要指数级增长的数据量去维持泛化性，使得模型训练的成本显著增加，这种现象被称为维度灾难。因此，特征选择旨在识别与标签高度相关的特征，并在模型训练过程中优先考虑这些特征，从而缓解维度灾难。特征选择不仅有助于降低与高维数据相关的计算成本，还通过拟合有意义的特征来提高模型泛化性能。在图学习中，常用的特征选择方法可以根据其与下游任务的关系分为两类：任务无关的特征选择和任务特定的特征选择。

1) 任务无关的特征选择

这类方法专注于选择能够适用于任何图神经网络模型或下游任务的特征，主要围绕引入正则化目标函数进行特征选择。例如，AsGNNs[13]将正则化方法引入 GCN 和 GAT 中，将特征选择与 GNN 结合在一起，以提取有意义的特征并消除噪声特征。具体来说，以 GCN 为例，引入$\ell_{2,1}$范数来对每一个图卷积层的参数进行约束，优化目标可以表示为：

$$\min \mathcal{L}_{gcn}(\boldsymbol{A}, \boldsymbol{X}; \boldsymbol{\Theta}) + \sum_{k=1}^{K} \lambda_k \| \boldsymbol{\Theta}^{(k)} \|_{2,1}$$

其中，$\| \boldsymbol{\Theta}^{(k)} \|_{2,1} = \sum_{i=1}^{N} \sqrt{\sum_{j=1}^{M} \left| \boldsymbol{\Theta}_{ij}^{(k)} \right|^2}$表示参数矩阵每一行的 2-范数总和，$\lambda_k > 0$ 是权重参数，为了简化计算，AsGNNs 将所有的λ_k设为λ。这种约束确保了学习的参数矩阵$\boldsymbol{\Theta}^{(k)}$具备行稀疏性，从而可以自然地进行特征选择。

2) 任务特定的特征选择

与之相对，任务特定的特征选择在进行特征选择时考虑了下游的具体任务。以 Dual-Net GNN[14]为例，注意到在节点分类任务中，选择性聚合的效果优于全部聚合，因此它提出利用输入节点特征的子集训练一个分类器，以预测节点标签，并且设计了一个选择器模型，学习最佳输入特征子集以实现性能提升。具体来说，分类器是一个两层的 MLP，它可以表示为$f_c(\boldsymbol{\theta}; \boldsymbol{X}, m)$，其中，$\boldsymbol{\theta}$是网络的参数，$\boldsymbol{X}$是节点特征矩阵，$m$是指示节点特征矩阵子集的一个掩码向量；选择器也是一个两层的 MLP，可以表示为$f_s(\boldsymbol{\phi}, m)$，其中$\boldsymbol{\phi}$是选择器的参数，$m$同样是一个掩码向量，选择器的输出是一个标量值，表示输入掩码向量在分类器上的预测性能。

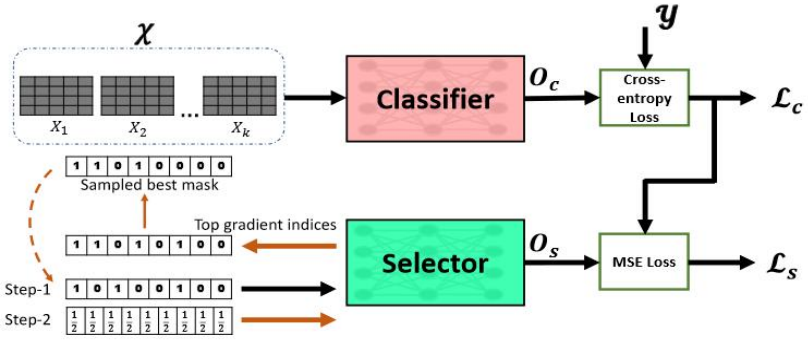


图 5-3 Dual-Net GNN 阶段二训练流程[14]

在训练时，分类器采用交叉熵损失，通过最小化选择器输出的预测性能与分类器真实的分类性能之间的均方误差来优化选择器。具体的训练流程分为三个阶段，第一阶段在节点特征矩阵的不同组合上训练分类器和选择器。对于每次前向传播，会随机采样一个节点特征子集作为分类器的输入。同样，对应的掩码向量也被设定为选择器的输入。分类器和选择器使用其对应的损失函数进行优化。训练直到分类器对每种掩码组合产生稳定的损失，且选择器能够学会根据掩码得到分类器的性能。如图 5-3 所示，第二阶段目标是利用训练好的选择器为分类器生成可能的最优掩码。Dual-Net GNN 认为输入中具有较大梯度的成分对模型输出的贡献更大，因此，首先使用一个所有索引权重相等的掩码向量（如 1/2）作为输入给选择器，然后计算相对于输入向量的梯度并挑选前p大梯度对应的索引。但最优子集可能是一个更小的子集，于是从中固定采样若干组合，并计算分类器的验证损失，选择损失最小的掩码对分类器和选择器进行一次训练。第二阶段重复若干次后进入第三阶段，选择验证损失最小的输入掩码，仅对分类器继续训练，直到收敛。

3. 特征补全

大多数图神经网络假设图中的节点特征是完整的，但这一假设在实际应用中往往并不能成立，主要原因包括以下几个方面：(1) 数据收集过程中出现的机器或人为错误；(2) 收集完整数据集在实际中成本很高；(3) 许多用户由于隐私保护不愿提供完整个人信息。因此，为了解决图中特征不完整的问题，特征补全作为一种重要解决方案，旨在填补图中缺失的节点特征。根据不同类型的图数据，现有方法可分为基于同质图的特征补全和基于异质图的特征补全。

1) 基于同质图的特征补全

大多数的同质图图神经网络往往假设图具有完整的特征信息，没有针对特征缺失图进行设

计，因此无法提供令人满意的学习效果。结构-属性转换器（SAT[15]）对图提出了共享潜在空间的假设，并开发了一种基于分布匹配的图神经网络，用于处理特征缺失的图。SAT 采用了一种结构与属性解耦的方案，能够进行同质图节点特征补全任务。

SAT 认为学习属性缺失图的一种可能方法是以解耦的方式输入结构和属性，同时允许结构和属性的联合分布建模。如图 5-4 所示，SAT 首先将属性和结构转换到潜在空间，然后通过对抗性分布匹配对配对的潜在表示进行对齐，最后解码回原始属性和结构，即配对的结构-属性匹配。具体来说，SAT 使用两个编码器：结构编码器（EA）负责处理图的结构信息，属性编码器（EX）负责处理节点的属性向量，这两个编码器将输入的数据映射到一个共享的潜在空间中，生成结构潜在因子z_a和特征潜在因子z_x，在潜在空间中，SAT 通过对抗性分布匹配方法，将编码得到的潜在表示与真实的先验分布进行对齐，最后 SAT 通过结构解码器（DA）和属性解码器（DX）重构图结构和节点属性。

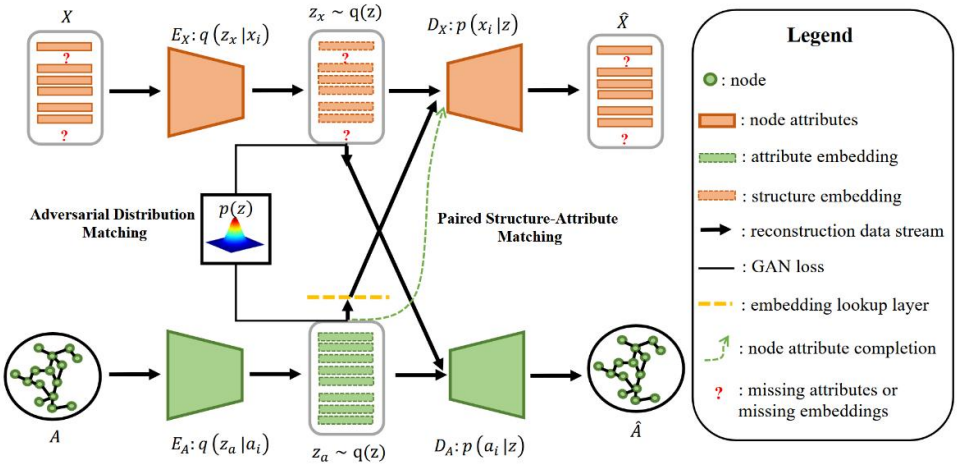


图 5-4 SAT 整体架构图[15]

2) 基于异质图的特征补全

和同质图不同，异质图包含多种类型节点，各种节点的特征并不在同一特征空间，给特征补全带来更大挑战。此外，异质图存在特定类型节点整体特征缺失问题，如图 5-5 所示, 在 IMDB 数据集中，只有电影节点具有原始属性，而在 DBLP 数据集中，只有论文节点具有原始属性，其他类型的节点则没有属性，原因往往是出于隐私考虑只能收集到非隐私的特征。在针对性的解决方案提出之前，人们普遍采用一些简单通用的手工补全方法，例如使用求和或平均值来替代缺失的特征。HGNN-AC[16]首次针对异质图的特征缺失提出解决方案，它不仅能够充分利用异质图的拓扑结构，还能结合不同类型节点和边之间的关联信息来进行节点属性的恢复。

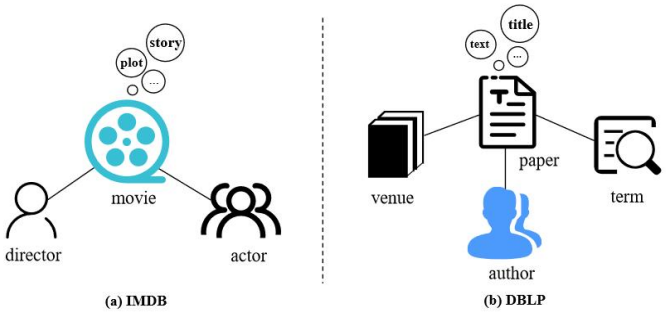


图 5-5 存在特征缺失的 IMDB 和 DBLP 网络[16]

如图 5-6 所示，HGNN-AC 对于给定的缺失特征的异质图，首先通过现有的异质图嵌入方法计算节点嵌入。接着，随机丢弃一些属性并进行属性补全。属性补全过程本质上是直接相连邻居属性的加权聚合，权重由节点嵌入推导出的注意力机制决定。在属性补全后，获得一个所

有节点都有属性的新异质图，并将其输入到任意的异质信息网络（HIN）模型中。由丢弃的属性与重构的属性构造补全损失，补全损失与预测损失的结合使整个模型可以端到端进行训练。

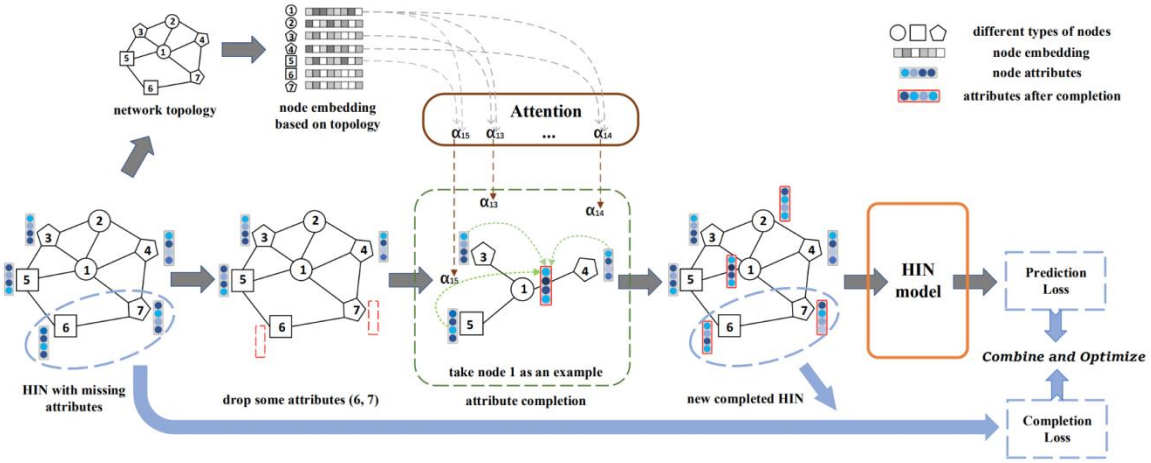


图 5-6 HGNN-AC 整体框架图[16]

5.1.3 图标签优化

在图数据中，标签通常用于描述图中节点或边的特征或类别。在这一部分中，将探讨多种技术，旨在优化图数据的标签信息，以此缓解过拟合和噪声标签等问题。这些技术包括：标签混合方法，它通过混合来自不同实例的标签来创建新的训练示例；以及伪标签，它通过训练好的模型扩展标签集，为未标记的节点分配标签。此外，还讨论了主动学习方法，在考虑标记成本的情况下，从数据集中选择最有效的数据进行标记，以获得最佳的模型性能。

1. 标签混合

标签混合是一种基于标签的增强学习技术，通过将两个不同实例及其关联标签结合为一个新实例，从而扩展训练集。这种方法不仅创造了新的训练样本，还增加了数据的多样性，对于提高模型的泛化能力至关重要。此外，模型还能够更好地应对各种输入，提升其在未见数据上的表现，从而有效减少过拟合现象。标签混合可以分为节点级混合和图级混合，具体介绍如下。

1) 节点级混合

节点级混合用于混合不同节点的标签，侧重于节点分类任务。节点级混合的一个主要挑战是图的拓扑结构复杂且不规则，需要在进行节点混合时同时考虑节点的拓扑结构。为了应对这个挑战，一个典型的方法是双分支的节点级混合[17]，它对节点特征和标签使用一样的过程进行混合，如图 5-7 所示。图的左侧是传统的图卷积，节点（用红色表示）通过聚合其邻居节点和自身当前表示的信息来更新其表示；图的右侧是提出的双分支图卷积方法，首先混合一对节点（一个红色节点和一个蓝色节点）的属性，即对两个节点特征和标签进行加权组合，之后对于每个节点进行单独的图卷积，最后将获得的两个分支的聚合表示进行混合并传递到下一层。

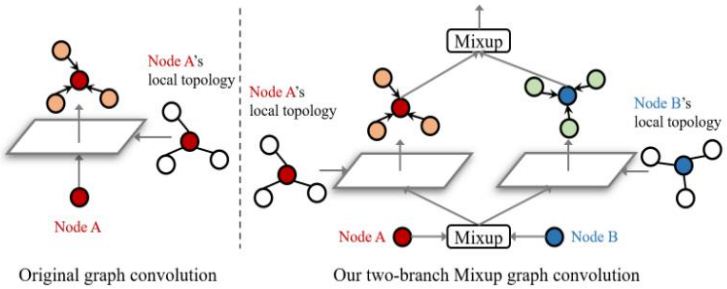


图 5-7 两分支图卷积方法[17]

2) 图级混合

与节点级混合不同，图级混合往往设计两个图的组合，需要直接混合两个图的标签。图级混合往往涉及多个步骤，如子图选择、图的移植、链接预测、标签混合等。一个代表性图级混合方法是 Graph Transplant[18]，如图 5-8 所示。Graph Transplant 将源图与目标图的混合分为四个步骤，第一步，计算源图G_π和目标图G的节点重要性向量$\boldsymbol{S}_\pi$和$\boldsymbol{S}$。第二步，在源图中提取以重要节点$\hat{V}_\pi$为锚的 K-hop 子图$\bar{G}_\pi$，在目标图中提取以随机节点$\hat{V}$为锚的 K-hop 子图$\bar{G}$。第三步，计算源图子图和目标图子图重要性。第四步，将源 K-hop 子图$\bar{G}_\pi$移植到剩余的目标子图$\bar{G}$中得到混合图，混合图的标签y'根据子图重要性自适应确定。

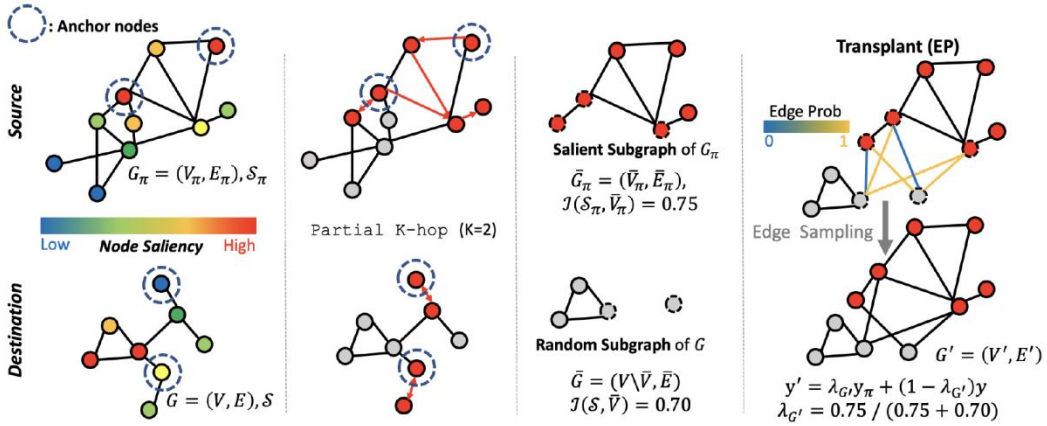


图 5-8 Graph Transplant 框架图[18]

2. 伪标签

图神经网络往往需要大量的标记数据用于模型的训练与验证，而标记数据往往非常昂贵。为了克服这一局限性，伪标签技术在模型预测的基础上，将未标记数据的预测结果作为“伪标签”，并与真实标记数据一起用于模型的后续训练，被广泛应用在有大量未标记数据但只有少量标记数据的场景。

基于这种思想，Li 等人[19]提出利用协同训练（Co-training）和自训练（Self-training）来扩展训练集。对于协同训练，在训练 GCN 的同时协同训练一个随机游走模型，利用随机游走模型探索图的全局结构，弥补 GCN 仅能处理局部结构不足。算法首先计算归一化吸收概率矩阵$\boldsymbol{P} = (\boldsymbol{L} + \alpha\boldsymbol{\Lambda})^{-1}$，其中$\boldsymbol{L}$是图的拉普拉斯矩阵，$\alpha$是正则化参数，$\boldsymbol{\Lambda}$是对角矩阵，$\boldsymbol{P}_{i,j}$表明了顶点$i$和$j$ 归属同一类别的可能性。之后，将标记节点按照类别分为不同的集合$S_1, S_2, ..., S_k$，其中S_k是类别k的标记数据集合。对于每个类别k, 计算置信向量$p = \sum_{j \in S_k} P[:, j]$, 其中$p \in \mathbb{R}^n$,$p_i$表示顶点$i$属于类别$k$的置信度。最后，找到置信度最高的$t$个顶点,并将它们添加到训练集中,赋予类别$k$的标签。对于自训练，使用给定的标记数据训练一个 GCN 模型，得到节点的预测结果$\boldsymbol{Z}$，对于每个类别k，通过比较 Softmax 得分，选择最自信的预测实例，将这些最自信的预测节点添加到对应类别的标签集中，并赋予它们类别k的标签。使用扩展后的标签集继续训练 GCN，并用预训练好的 GCN 参数来初始化。

3. 主动学习

主动学习的目标是在标记成本有限的情况下，如何从数据集中选择最有效的数据进行标记，以获得最佳的模型性能，它是一种选择数据的方法而不是标记数据的方法。虽然许多主动学习方法已应用于图数据，但大多数仍集中于节点分类问题。

图上的主动学习往往假设节点之间存在关联，考虑节点之间的相互作用。这一思想避免了选择的节点相似且聚集在一起，导致信息的重复。接下来以一个基于聚类的通用图数据主动学习框架 FeatProp[20]为例，介绍如何最大限度地确保选择的节点的多样性，避免冗余信息。该框架主要包括两个步骤：1）使用节点特征 $\boldsymbol{X}$ 和图结构 G 计算距离矩阵或函数 $d_{\boldsymbol{X},G}$；2）在该距

离矩阵上应用具有 b 个中心的聚类，并从每个簇中选择离聚类中心最近的节点。将这些节点赋予标签（由矩阵 $\boldsymbol{Y}$ 提供）后，基于 $\boldsymbol{X}$、G 和 $\boldsymbol{Y}$ 训练图神经网络，用于节点分类任务。具体来说，FeatProp 定义了对应节点特征经过传播后的成对节点距离，使用 L_2 范数来表示：$d_{\boldsymbol{X},G}(v_i, v_j) = \| (\boldsymbol{S}^K\boldsymbol{X})_i - (\boldsymbol{S}^K\boldsymbol{X})_j \|_2$, 其中$K$是传播总层数，$\boldsymbol{S}$是归一化的邻接矩阵。从直观上讲，这消除了未训练参数对距离的影响，同时仍考虑了图结构。对于聚类方法，FeatProp 应用 K-Medoids 聚类。K-Medoids 与K-Means 相似，但所选的中心必须是数据集中的真实样本节点。这对主动学习至关重要，因为无法对K-Means 产生的虚假聚类中心进行标记。

5.2 架构优化

随着图神经网络的发展，模型网络层数不断增加，数据规模不断扩展，在原有的 GNN 架构下出现了诸多挑战。例如，网络层数的加深会导致节点间特征趋于一致，远距离节点间信息丢失；图数据的扩大会导致计算量骤增，影响性能效率等。因此，涌现了大量的架构优化方法以应对出现的挑战。在本节中，将主要从信息传递、采样、池化三个方面介绍如何进行架构优化。这些优化技术有助于增强深层图神经网络的性能，同时降低计算负担并保持信息完整性。

5.2.1 信息传递优化

基于消息传递的图神经网络在各种分类任务中表现出色，主要归功于其能够通过消息传递机制有效地聚合邻居节点的信息，从而捕捉复杂的图结构与节点特征之间的关系。然而，这种信息传递机制仍然存在一些局限性。首先，随着模型层数的增加，容易出现过平滑问题，即节点的表示逐渐变得相似，最终导致不同类别节点难以区分。其次，过压缩问题也会影响性能，随着网络层次加深，虽然信息传递量呈指数增长，但传递到目标节点的信息却被压缩为固定大小，无法有效传递长距离依赖信息。此外，消息传递图神经网络表达能力的上限为 1-WL 测试算法，因此在处理复杂图结构时，模型的判别能力有限。

1. 过平滑

过平滑定义: 假设G是一个无向的连通图，$\boldsymbol{X}^n \in \mathbb{R}^{v\times m}$表示在图$G$上定义的$N$-层 GNN 的第$n$层隐藏特征。此外，称$\mu: \mathbb{R}^{v\times m} \to \mathbb{R}_{\geq 0}$为一个节点相似性度量，如果它满足：

(1) $\exists \boldsymbol{c} \in \mathbb{R}^m$,当对所有节点$i \in V$，$\boldsymbol{X}_i = \boldsymbol{c}$时，满足$\mu(\boldsymbol{X}) = 0$，$\boldsymbol{X} \in \mathbb{R}^{v\times m}$。

(2) $\forall \boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^{v\times m}$，$\mu(\boldsymbol{X}+\boldsymbol{Y}) \leq \mu(\boldsymbol{X}) + \mu(\boldsymbol{Y})$

那么，过平滑被定义为相似性度量μ随着 GNN 层数n增加逐渐收敛到 0 的过程，即：

(3) $\mu(\boldsymbol{X}^n) \leq C_1 e^{-C_2 n}, \quad n = 0, \ldots, N$.其中$C_1$和$C_2$是正常数。

在这个定义中，为了保持一般性，假设相似性度量μ收敛到 0。如果某种特定的相似性度量收敛到一个非零常数，则可以重新定义相似性度量使其收敛到 0。这确保了定义适用于各种情境，不依赖于某个特定的相似性度量。接下来对定义进行进一步解释。这个定义提供了一个精确的方式来描述深度 GNN 中的过平滑现象。它从数学上给出了节点特征如何在层数增加时趋同，并且通过指数衰减来量化这种收敛过程。定义中的条件(1)形式化了一个普遍接受的观点，即过平滑是由节点特征收敛到一个常数向量引起的，而条件(3)提供了一个更严格的量化标准，用于衡量这种收敛。条件(2)中的三角不等式或次可加性排除了相似性度量中的退化选择。定义仅适用于连通图的情况。然而，该定义可以直接推广到非连通图。在这种情况下，对每个连通分量$S \subseteq V$应用节点相似性度量μ_S,并定义全局相似性度量为各个连通分量上的节点相似性度量之和，即$\mu = \sum_S \mu_S$。这样就可以处理不同连通分量分别收敛到不同常数节点值的情况。下面介绍几种缓解过平滑的方法。

1) 归一化和正则化

正则化是一种防止机器学习模型过拟合的技术，旨在通过对模型增加约束或噪声，使模型在训练数据上表现良好的同时，也能在未见过的数据上具有较好的泛化能力。正则化方法在防

止图神经网络的过平滑方面也起着重要的作用。例如，之前介绍过的 DropEdge[3]是一种有效的正则化技术，它通过在训练过程中随机删除图中的边来引入噪声。这种方法不仅增加了训练过程中的随机性，还促使模型在每次训练迭代时，图结构进行动态变化。具体来说，当一些边被随机删除时，信息传播路径也随之改变，这使得节点在信息传播过程中不再过度依赖于相邻节点的特征，从而有效阻止了信息在深层网络中过度传播。通过这种方式，DropEdge 使得节点之间的差异得以保持，进而缓解了过平滑问题。

归一化是一种常用的数据处理技术，旨在调整不同特征的尺度，使其在模型训练时更具可比性。在机器学习和深度学习中，归一化的主要目的是提高模型的收敛速度和稳定性，减少训练过程中可能出现的数值不稳定性。归一化在防止过平滑方面也发挥积极作用。它通过保持特征的多样性和稳定性，帮助 GNN 在深层网络中实现更好的特征表示。例如 PairNorm[21]方法通过在每一层的 GNN 之后对节点特征进行归一化处理，确保节点对之间的距离保持恒定，可以形式化表示如下：

$$\begin{aligned}\widehat{\boldsymbol{X}}_i &= \boldsymbol{X}_i - \frac{1}{v}\sum_{j=1}^{v}\boldsymbol{X}_j \\ \boldsymbol{X}_i &= \frac{s\widehat{\boldsymbol{X}}_i}{\sqrt{\frac{1}{v}\sum_{j=1}^{v}\|\widehat{\boldsymbol{X}}_j\|_2^2}}\end{aligned}$$

首先，它从每个节点的特征向量中减去所有节点特征的均值，以确保节点特征在网络中不会因为所有特征逐渐趋同而失去区分度。然后，按照所有节点特征的ℓ_2范数的均值进行归一化处理，并通过一个超参数s进行缩放。这一步确保节点特征在网络中的数值范围不会过大或过小，从而提高网络的稳定性和表现。

2) 改变图神经网络的消息传递过程

改变图神经网络的消息传递过程也是一种缓解深层图神经网络的过平滑问题的方法。一种经典的网络架构是图耦合振荡器网络 GraphCON[22]。在传统的图卷积网络中，节点特征的传播机制通常采用扩散模型，这种模型通过多层的信息聚合逐渐使节点特征趋于相似，进而导致过平滑问题。GraphCON 通过将信息传播的机制转变为基于非线性振荡器的动态，避免了特征的快速收敛。GraphCON 的消息传递过程可以形式化表示为：

$$\begin{aligned}\boldsymbol{Y}^n &= \boldsymbol{Y}^{n-1} + \Delta t\left[\sigma\left(F_{\theta^n}(\boldsymbol{X}^{n-1}, G)\right) - \gamma\boldsymbol{X}^{n-1} - \alpha\boldsymbol{Y}^{n-1}\right], \\ \boldsymbol{X}^n &= \boldsymbol{X}^{n-1} + \Delta t\boldsymbol{Y}^n,\end{aligned}$$

其中，$\boldsymbol{Y}^n$是辅助节点特征，用于捕获节点间的动态交互。$\Delta t > 0$ 表示时间步长 (通常设置为$\Delta t = 1$)，这一参数决定了特征更新的速度。$F_{\theta_n}(\boldsymbol{X}^{n-1}, G)$是图神经网络中的消息传递函数，它从图$G$和前一层的节点特征$\boldsymbol{X}^{n-1}$中生成新的信息。这个函数是模型的核心，用于聚合来自邻居节点的信息。σ是非线性激活函数，用于引入非线性特性，增强模型的表达能力。γ和α控制特征消息传递的参数。它们影响特征更新的方式，能够调节节点特征在传播过程中的变化速度。

非线性振荡器作为一种动态建模方法，虽然在缓解 GNN 的过平滑问题上表现出色，但也存在一些缺点，如依赖于特定的超参数、对数据敏感等，这些缺点促使研究者提出了梯度门控（Gradient Gating，G^2）[23]方法。G^2方法的关键方程如下：

$$\hat{\tau}^n = \sigma\left(\widehat{F}_{\hat{\theta}^n}(\boldsymbol{X}^{n-1}, G)\right),$$

$$\tau_{ik}^n = tanh\left(\sum_{j\in\mathcal{N}_i}\left|\hat{\tau}_{jk}^n - \hat{\tau}_{ik}^n\right|^p\right),$$

$$X^n = (1-\tau^n)\odot \boldsymbol{X}^{n-1} + \tau^n \odot \sigma\left(F_{\theta^n}(\boldsymbol{X}^{n-1}, G)\right).$$

G^2方法首先定义了一个可学习的门控函数，得到初步门控值；然后利用了与节点i相邻的所有节点的门控值来计算当前节点的更细致的控制信号，这个控制信号引入了局部结构信息。最后，通过门控方法有效减缓不必要的信息过度传播。G^2有助于保留节点特征的多样性，避免所有节点特征过快地趋同于常量，从而提高 GNN 在各类任务中的表现。

3) 残差连接

残差连接是深度学习中的一种结构，它最早在 2016 年由 Kaiming He 等人提出于 ResNet（Residual Network）中。其核心思想是在神经网络的每一层中，将输入直接与该层的输出相加，从而形成一种“残差”学习的方式。残差连接的引入有效缓解了深层网络训练中的梯度消失和信息丢失问题，深层网络中，梯度往往在反向传播时逐渐减小，导致难以训练。残差连接提供了直接的路径，使得梯度能够更有效地传播；由于输入和输出的相加，网络能够更好地保留原始输入的信息，从而防止特征在层与层之间的过平滑。在图神经网络中，残差连接具有以下的形式[24]：

$$\boldsymbol{X}^n = \boldsymbol{X}^{n-1} + F_{\theta_n}(\boldsymbol{X}^{n-1}, G)$$

其中，$\boldsymbol{X}^n$是第n层的节点特征，$F_{\theta_n}(\boldsymbol{X}^{n-1}, G)$是基于当前特征和图结构计算得到的消息传递输出。这一形式的优点在于可以使得模型在更深的层次上保留输入特征，从而有效缓解信息丢失和过平滑的问题。Chen 等人 [25]进一步发展了这一思想，采用了一种缩放的残差连接，它通过引入可调节的缩放因子来控制在残差学习中的信息流动，将初始节点特征添加到每一层的输出：

$$\boldsymbol{X}^n = \sigma\left[\left((1-\alpha_n)\widehat{\boldsymbol{D}}^{-\frac{1}{2}}\widehat{\boldsymbol{A}}\widehat{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{X}^{n-1} + \alpha_n X^0\right)\left((1-\beta_n)\boldsymbol{I} + \beta_n \boldsymbol{W}^n\right)\right]$$

其中，$\boldsymbol{X}_0$是初始节点特征，$\widehat{\boldsymbol{D}}$和$\widehat{\boldsymbol{A}}$分别是加入了自环的图的度矩阵和邻接矩阵，α_n是一个在层n中的缩放因子，控制输入特征的贡献程度。通过调整α_n，模型可以在当前层的输出与初始输入特征之间进行平衡。β_n是另一个缩放因子，用于调整不同部分的权重，使得模型在每一层中能够适应不同的特征传播方式。

2. 过压缩

在图神经网络中，随着网络层数的增加，节点特征信息在消息传递过程中被压缩成固定大小的向量。这种现象类似于一个“瓶颈”，导致信息的有效传递受到限制。具体而言，当图的层数增加时，信息从源节点传递到目标节点的路径变长。在这个过程中，节点的特征信息在消息聚合时被压缩，从而导致来自远节点的信息减弱或丢失。换句话说，尽管信息可能呈指数级增长，但最终传递到目标节点的却是有限且固定大小的信息。这个“瓶颈”被称为图神经网络中的过压缩问题。这个问题最早是通过经验观察到的，为了更深入的研究，人们尝试通过衡量每个节点表示$\boldsymbol{h}_i^{(l)}$对输入特征x_s的敏感度$\frac{\partial \boldsymbol{h}_i^{(l)}}{\partial \boldsymbol{x}_s}$来量化这一问题，并定义了过压缩分数。

过压缩分数定义: 经过l层消息传递后，节点i和节点s之间的过压缩分数定义为：

$$OSQ(i,s) = \left\|\frac{\partial \boldsymbol{h}_i^{(l)}}{\partial \boldsymbol{x}_s}\right\|$$

其中$\|\cdot\|$是矩阵的谱范数。

过压缩分数上界定义: 设$i, s \in V$且$s \in S_{r+1}(i)$，如果在 $0 \le l \le r$的范围内满足$\| \nabla \phi_l \| \le \alpha$和$\| \nabla \psi_l \| \le \beta$,则有

$$\| \frac{\partial \boldsymbol{h}_i^{(r+1)}}{\partial \boldsymbol{x}_s} \| \le (\alpha\beta)^{r+1} \widehat{\boldsymbol{A}}_{is}^{r+1}$$

其中，$S_{r+1}(i)$是节点i的邻居集合，其在图G上到节点i的最短路径距离为$r+1 \in \mathbb{N}$。ψ_l和ϕ_l分别表示第l层网络的消息聚合函数和更新函数。$\widehat{\boldsymbol{A}}^{r+1}$是指$\widehat{A}$的$r+1$ 次幂。这个过压缩分数的上界表明，如果$\nabla\phi_l$和$\nabla\psi_l$的导数是有界的，则消息的传播受到$\widehat{\boldsymbol{A}}$的控制，该定义也适用于重新加权的邻接矩阵。通过这个界限，能够理解如何在多层网络中控制信息传播，进而应对过压缩带来的挑战。下面介绍几种缓解过压缩的方法。

1) 动力学启发的消息传递

有些研究可能并不主要关注于过压缩问题，甚至没有重连的过程，但这些模型中的信息传递规则可以等效地视为在一个更稠密或完全连接的图上更新节点特征，因此可视为隐式重连范式。接下来介绍的基于分数邻接矩阵的消息传递就是一个例子。

在经典的图神经网络中，消息传递主要集中在节点的局部邻域。为了解决这种局部邻域的消息传递导致的过压缩问题，一些基于非局部动态的消息传递神经网络应运而生。例如，近期的研究[26]引入了一种称为“分数邻接矩阵”的新概念，记作$\widehat{\boldsymbol{A}}^{\tau}$，其中$\tau \in \mathbb{R}$。基于分数邻接矩阵就可以实现非局部消息传递。具体而言，消息传递规则为$\frac{\partial \boldsymbol{H}}{\partial t} = -\widehat{\boldsymbol{A}}^{\tau}\boldsymbol{H}$，其中$\tau$决定了连接的密度和传播的范围。通过选择适当的$\tau$值，可以使图的连接性变得更加密集，这种调整允许在不同的连接水平上传递节点特征信息，从而有效地将信息从一个节点传递到更远的节点。

2) Graph Transformers

受到 Transformer 在文本和图像领域成功的启发，研究者们开始将其引入到图领域中，形成了 Graph Transformers。这些模型利用 Transformer 的能力来处理图结构数据。和传统图神经网络不同，Graph Transformers 具有全局感受野，即它为所有节点对分配注意力，而不论它们在原始图中是否相连。通过这种方式，即使原本不直接相连的节点之间也能有效地进行信息传递，从而缓解过压缩问题。接下来介绍一个典型的例子 Graphormer[27]。

Graphormer 采用空间编码和边编码的方法为图的整个邻接矩阵分配注意力，从而增强模型对节点特征和整体图结构的理解。具体来说，与经典 Transformer 相似，节点特征首先通过查询-键编码方案进行编码，即$\boldsymbol{Q} = \boldsymbol{H}\boldsymbol{W}_Q, \boldsymbol{K} = \boldsymbol{H}\boldsymbol{W}_K$，其中$\boldsymbol{W}_Q$和$\boldsymbol{W}_K \in \mathbb{R}^{c_0 \times d_K}$是节点特征的初始投影。节点特征可以通过与度数相关的相对重要性进行增强（即中心性）。具体地，对于每个输入特征向量$x_i^{(0)}$，可以构造$h_i^{(0)} = x_i^{(0)} + \deg^-(i) + \deg^+(i)$，其中$\deg^-(i)$ 和$\deg^+(i) \in \mathbb{R}^{c_0}$ 是可学习的嵌入向量，分别由节点 i 的入度和出度决定。 在初始编码阶段之后，图的邻接矩阵通过空间编码和边编码来丰富。对于空间编码，Graphormer 考虑一个函数$\zeta(i,j): V \times V \to \mathbb{R}$，用于测量节点$i$和$j$之间的空间关系，如果节点$i$连接到节点$j$,则选择短路径距离，否则为 -1。结果$\zeta(i,j)$作为邻接矩阵的偏置项，即

$$A_{i,j} = \frac{\left(\boldsymbol{h}_i \boldsymbol{W}_Q\right)\left(\boldsymbol{h}_j \boldsymbol{W}_K\right)^{\top}}{\sqrt{d}} + b_{\zeta(i,j)}, \forall i,j \in \mathcal{V}$$

其中$b_{\zeta(i,j)}$是一个根据$\zeta(i,j)$索引的可学习标量，在所有层中共享。当$b_{\zeta(i,j)}$是一个关于 $\zeta(i,j)$的递减函数时，模型将更加关注低度邻居的信息，而减少对距离较远邻居的关注。 对于边编码，Graphormer 进一步将边信息引入建模节点之间的相关性。对于每一对有序节点i, j, Graphormer 计算从i到j的最短路径$SP(i,j) = (e_1, e_2, ..., e_N) \subseteq d_G(i,j)$,并计算沿路径的边特征和可学习嵌入的点积平均值。结合空间编码，最终的邻接矩阵可以表述为

$$\boldsymbol{A}_{i,j}=\frac{\left(\boldsymbol{h}_i\boldsymbol{W}_Q\right)\left(\boldsymbol{h}_j\boldsymbol{W}_K\right)^{\top}}{\sqrt{d}}+b_{\zeta(i,j)}+c_{i,j}$$

其中，$\forall i,j\in\mathcal{V}$，$c_{i,j}=\frac{1}{N}\sum_{n=1}^{N}\boldsymbol{x}_{e_n}\left(\boldsymbol{w}_n^E\right)^{\top}$。$\boldsymbol{x}_{e_n}$是$SP(i,j)$中第$n$条边$e_n$的特征，$\boldsymbol{w}_n^E\in\mathbb{R}^{d_E}$是第$n$条边的可学习权重嵌入。最后，Graphormer 通过自注意力机制与前馈块的结合，利用设计的邻接矩阵，能够高效地处理图数据并提高信息传递的准确性。在大规模分子数据集上的实验进一步说明了其在面对过压缩问题时的鲁棒性，展示了其在实际应用中的有效性。

3. 表达能力受限

1-WL 算法是一种图同构测试方法, 它的工作原理是基于节点的“标签”传播：每个节点根据它的邻居来更新自己的标签，并迭代多次，直到标签不再变化。如果两个图在多轮迭代后每个节点的标签都相同，则认为这两个图在结构上是相同的（同构）。尽管 1-WL 在许多场景中表现良好，但它并不能区分所有的图。比如有些图虽然在结构上不同，但 1-WL 无法将它们区分，这是因为 1-WL 的标签传播机制在某些情况下不够强大，无法捕捉图中的所有结构信息。

许多早期的 GNN 设计受到 1-WL 的启发，通过聚合节点及其邻居的特征来学习节点和图的表示。这种方法类似于 1-WL 中的“标签传播”，因此这些 GNN 的表达能力与 1-WL 是等效的。换句话说，这些 GNN 能够处理和区分图结构的能力的上限即为 1-WL 算法，导致在一些任务上存在局限性。例如，1-WL 和基于 1-WL 的 GNN 难以检测和计数一些重要的图子结构，尤其是三角形，而三角形在社交网络分析、分子结构分析等许多应用中非常关键。鉴于此，许多研究试图设计能够超越 1-WL 表达能力的 GNN，来更好地处理复杂的图结构和子结构，让 GNN 具备更强的区分和表达能力。下面介绍几种经典的模型。

1) 高阶图神经网络模型

使图神经网络超越 1-WL 的一种直接方式是基于k-WL 算法。Morris 等人提出了一种基于 k-set WL 算法的k-GNN[28]，其中k-set WL 是k-WL 的一种变体。具体来说，考虑包含 k 个节点的子图，而不是 k-元组的节点，从而使得算法更高效，同时也能考虑图结构信息。形式上，包含所有 k-set 的集合记为： $[V]_k=\{S\subseteq V||S|=k\}$ 它们就是包含 k 个节点的子图。此外，k-set S 的邻居定义为只差一个节点的 k-set，即： $N_{V,k}(S)=\{J\in[V]_k||J\cap S|=k-1\}$ 尽管 k-set WL 比 k-WL 弱，但它比 1-WL 更强大且比 k-WL 更具扩展性。在 k-GNN 中，第 t 层中 k 个节点的子图 S 的嵌入表示为 $\boldsymbol{x}_k^{(t)}(S)$，并且分配给每个由 S 引出的子图的初始特征表示相应子图的同构类型。然后，k-GNN 的节点嵌入可以通过消息传递机制根据下式进行更新：

$$\boldsymbol{x}_k^{(t)}(S)=\sigma\left(\boldsymbol{W}_1^{(t)}\boldsymbol{x}_k^{(t-1)}(S)+\sum_{U\in N_{V,k}(S)}\boldsymbol{W}_2^{(t)}\,\boldsymbol{x}_k^{(t-1)}(U)\right)$$

由于集合 k-WL 比 1-WL 更强大，因此k-GNN 比消息传递神经网络（MPNNs）更强大，并且通过适当的参数矩阵初始化，k-GNN 被证明可以与集合k-WL 一样强大。

2) 基于子图的图神经网络

基于k-WL 的更具表达能力的架构在扩展到大图时可能面临挑战。为了解决这一问题，研究者提出通过移除、提取或标记更小的子图来增强 GNN 的表达能力。受到非同构图总是具有非同构子图这一观察的启发，基于子图的 GNN 使 GNN 能够有效利用给定图中的更多结构模式，从而实质上打破了由 GNN 局部聚合函数引起的对称性。接下来介绍一种数据驱动的基于子图的 GNN 架构，k-有序子图消息传递神经网络[29]，它的预测准确性相比于非数据驱动的增强子图图神经网络有所提高，同时减少了计算时间。

在初始化时，神经网络为每个有序子图 g 以及节点 v 学习两个特征：

$$\boldsymbol{h}_{v,g}^{(0)} := \mathrm{UPD}(v, g)$$

$$\boldsymbol{\pi}_{v,g} := \mathrm{UPD}_{\pi}(v, g)$$

其中 UPD 和 UPD_{π} 是可微分的参数化函数。第一个特征与其余的节点特征连接到一起，第二个特征 $\boldsymbol{\pi}_{v,g}$ 用来选择与节点 v 相关的有序子图。在每一层 $(i+1)$ 中，根据有序子图 g 更新顶点 v 的特征：

$$\boldsymbol{h}_{v,g}^{(i+1)} := \mathrm{UPD}^{(i+1)}\left(\boldsymbol{h}_{v,g}^{(i)}, \mathrm{AGG}^{(i+1)}\left(\left\{\boldsymbol{h}_{u,g}^{(i)} | u \in N_G(v)\right\}\right)\right)$$

经过 T 层这样的更新后，对于每个顶点 v， 根据$\boldsymbol{\pi}_{v,g}$ 学习一个联合特征：

$$\boldsymbol{h}_{v}^{(T)} := \mathrm{SAGG}\left(\{\boldsymbol{h}_{v,g}^{(T)} | g \in G_k \; s.t. \; \boldsymbol{\pi}_{v,g} \neq \boldsymbol{0}\}\right)$$

其中SAGG是子图聚集函数，它利用 $\pi_{v,g}$ 来选择 k-有序子图集合的一个子集。上述方法无法在所有包括k个节点的子图上进行消息传递，数据驱动的子图选择弥补了这一缺陷并提升了模型可扩展性。具体而言，设 G 为一个图，其中每个节点 v 具有初始特征 $\boldsymbol{h}_{v}^{(0)}$，所有节点的特征按行堆叠成特征矩阵 $\boldsymbol{H} \in \mathbb{R}^{n \times d}$。此外，设 $\boldsymbol{h}_{W_1}: G \times \mathbb{R}^{n \times d} \to \mathbb{R}^{m \times n}$ 为一个置换等变函数，例如消息传递 GNN，由 $\boldsymbol{W}_1$ 参数化，映射图G及其初始特征H到参数矩阵：

$$\boldsymbol{\theta} := \boldsymbol{h}_{W_1}(G, \boldsymbol{H}) \in \mathbb{R}^{m \times n}$$

直观上，每个参数$\boldsymbol{\theta}_{ij}$是顶点j属于图G的第i个抽样子图的一种先验概率。可以使用扰动-最大后验（perturb-and-MAP）的方法来采样子图，该方法对于有序和无序子图都适用。使用扰动-最大后验方法采样第 i 个有序 k 个节点子图 g_i的过程可以形式化表示为：

$$g_i := z^*(\boldsymbol{\theta}_i + \boldsymbol{\varepsilon}_i),$$

其中 $\boldsymbol{\varepsilon}_i \sim \rho(\boldsymbol{\varepsilon})$，$\rho(\boldsymbol{\varepsilon})$ 是一种噪声分布，例如 Gumbel 分布。对于无序子图，仅需确定 $\boldsymbol{\theta}_i + \boldsymbol{\varepsilon}_i$ 中的最大的前 k 个值；而对于有序图还需要确定排序，最坏情况运行时间为 $O(n + k\log k)$。

3) 基于非等变操作的图神经网络

某些非等变操作可以直接打破传统消息传递图神经网络的对称性，从而增强图神经网络的表达能力，超越 1-WL。例如，RP-GNN[30]在计算图的嵌入时，通过对图上所有可能的排列取平均，确保了对称性，即结果不受节点顺序影响。形式上，关系池化通过任意函数 f 来获得图 G 的嵌入，具体如下：

$$f(\boldsymbol{A}, \boldsymbol{X}) = \frac{1}{n!} \sum_{\sigma \in S_n} f\,(\sigma \cdot \boldsymbol{A}, \sigma \cdot \boldsymbol{X}),$$

其中σ是在对称群S_n中定义的置换，即节点的排列。为了提高图神经网络的表达能力，DJ Aldous 等人[31]为每个节点附加一个对置换敏感的标识符，这可以通过将独热编码附加到节点特征上来实现。由此衍生出的新神经网络架构定义如下：

$$f(\boldsymbol{A}, \boldsymbol{X}) = \frac{1}{n!} \sum_{\sigma \in S_n} f\,(\boldsymbol{A}, [\boldsymbol{X}, \sigma \cdot \boldsymbol{I}_n]),$$

其中 $\boldsymbol{I}_n \in \mathbb{R}^{n \times n}$ 是单位矩阵。通过这种方式，模型变得非等变，因为它开始考虑节点的身份而不仅仅是其特征。

5.2.2 图采样优化

在大规模图数据中，节点往往拥有大量的邻居，传统图神经网络直接聚合所有邻居，带来不可控制的计算和存储代价。图采样是处理大规模图数据的重要技术之一，它通过采样一部分邻居节点进行图神经网络聚合和训练，从而减少了计算资源的消耗，并且加快模型收敛。传统图采样一般基于固定的随机采样，即在每轮迭代中以等概率采样邻居节点，忽略了邻居节点的重要性，并且不能自适应的基于当前训练状态进行调整，造成模型性能的降低。在本小节中，将探讨如何优化现有图采样过程，一方面，将介绍基于重要性的图采样，通过聚合关键信息，提升效率的同时保证效果；另一方面，将介绍自适应的图采样，通过在训练过程中动态调整采样策略，更好地利用训练过程中的图信息。

1. 重要性采样

在最基本的随机采样策略中，同一节点采样不同邻居节点的概率相同。与随机采样不同，重要性采样会给不同的采样节点分配不同的采样概率，而这些概率是由其在图中重要性程度决定的。这样使得采样得到信息更重要的节点，减少由采样引起的方差，从而使训练过程更稳定并提高模型性能。根据采样对象在图结构中的不同层次，可以分为以下三种采样方法：节点级采样、层级采样、基于子图采样，下面将从这三个方面介绍重要性采样。

1) 节点级采样

最基本的节点级采样是由 GraphSage[32]首次提出的，针对聚合所有邻居会导致较大计算量的问题，提出只采样部分邻居节点进行聚合。它对邻居节点的采样方式采用的是随机采样，将所有邻居节点视为同等重要。而在基于重要性的节点级采样中，邻居节点具有不同程度的重要性，采样时考虑更为重要的邻居节点。其中一个较为典型的方法是 PinSage[33]，它定义的节点邻居不仅仅是一阶邻居，而是在其周围选取最为重要的邻居节点，可能包含多阶邻居。具体来说，它从目标节点开始进行随机游走（Random Walk），计算各个邻居节点的访问数，归一化后作为其重要性权重，最终采样选取权重最大的前 t 个邻居节点。例如在图 5-9 中，从节点 0 出发随机游走得到了四个节点序列，并统计得到其中每个节点被访问的次数，从而选取了前 3 个访问数最高的节点作为节点 0 的邻居节点（即节点 5、10、11）。

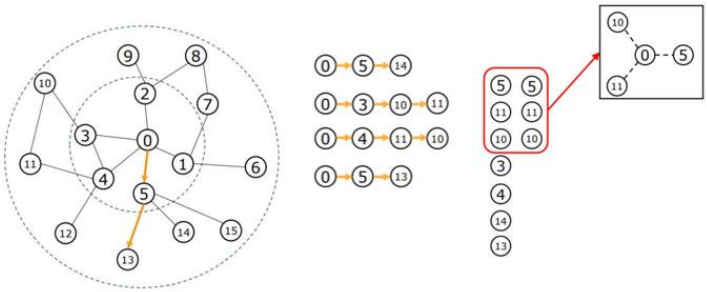


图 5-9 PinSage 的重要性采样[33]

2) 层级采样

层级采样是 FastGCN[34]最初提出的，针对节点级采样随着层数的加深依旧存在邻域爆炸的问题，提出通过限制每一层中的采样数量，使得采样的邻居数量随着层数的加深只会线性增长。具体来说，对于某一层，它会从整个节点集合中采样出一组共享的节点集合，那么这一层所有节点表示的更新都会基于这组共享采样节点计算（如下图 5-10）。其中假设V表示所有节点，V^k表示第 k 层所需聚合的节点，层次级采样会在V中采样一定数量的节点，那么V^{k-1}即为这些节点和V^k邻居节点的交集。如此往复，依次采样得到每层的采样节点集合。

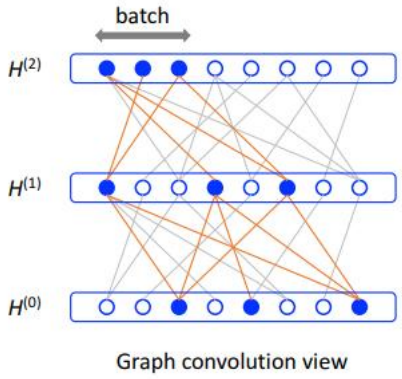


图 5-10 层级采样[34]

为了保证按层采样是无偏的，FastGCN 采用了重要性采样。其在每一层采样时每个节点服从以下概率分布：

$$q(u)=\frac{\|\widehat{\boldsymbol{A}}(:,u)\|^2}{\sum_{u'\in V}\|\widehat{\boldsymbol{A}}(:,u')\|^2},\quad u\in V$$

其中，$\widehat{\boldsymbol{A}}(:,u)$表示矩阵$\widehat{\boldsymbol{A}}$中节点$u$的一列，$V$为所有节点，并据此概率分布采样$t$个节点。从另一个角度看，这是近似将节点度数作为衡量重要性的标准。尽管 FastGCN 十分高效，但由于采样是按层单独进行的，连续两层中采样的节点可能并不相连或者连接十分稀疏，某些节点可能无法从其邻居节点获得任何信息，从而可能导致最终近似的嵌入具有较大的方差。

3) 基于子图采样

最基本的基于子图的采样是由 ClusterGCN[35]提出的，其核心思想是只在采样的子图上进行完整的消息传递，从而减少内存和计算资源的消耗。为了在采样时进一步考虑子图的重要性，GraphSAINT[36]提出了四种可选择的采样器（节点采样器、边采样器、随机游走采样器、多维度随机游走采样器），下面具体以节点采样器和边采样器为例介绍。对于节点采样器，GraphSAINT 采用与 FastGCN 相同的采样节点概率（$P(u):=\|\widehat{\boldsymbol{A}}(:,u)\|^2/\sum_{u'\in V}\|\widehat{\boldsymbol{A}}(:,u')\|^2$），即某个节点采样概率为某个节点的度/图上所有节点的度之和。随后，从节点集合V中根据此概率（放回地）采样n个节点，生成节点集V_s。最终，通过V_s形成G的一个诱导子图(induced subgraph) G_s；对于边采样器，采样边的概率分布为：$P\big((u,v)\big):=\left(\frac{1}{\deg(u)}+\frac{1}{\deg(v)}\right)/\sum_{(u',v')\in E}\left(\frac{1}{\deg(u')}+\frac{1}{\deg(v')}\right)$，类似地从边集合$E$中根据此概率随机（放回地）采样$m$条边，每条边的顶点组成节点集$V_s$。最终，同样通过 V_s 形成 G 的一个诱导子图 G_s。可以看出，两者对重要性的考虑都在于节点的度。

2. 自适应采样

与重要性采样的固定采样策略不同，自适应采样会在模型的训练过程中根据目前模型的效果表现或者其他训练信息动态地调整采样策略，从而充分利用训练过程中的图信息，提高模型性能。同样的，根据采样的对象在图结构中的不同层次，自适应采样可以分为节点级采样和层级采样，下面将分别进行介绍。

1) 节点级采样

基本的节点级采样之前已经介绍过，即对于每个目标节点都会采样固定数量的邻居节点。对于节点级的自适应采样，下面以 VR-GCN[37]为例介绍。它和 GraphSage 一样每次采样都是固定数量的节点，但将 VR-GCN 考虑为自适应采样，是由于它保留并利用之前的节点信息进行当前节点特征的更新，这些信息会随着模型学习逼近未采样的节点。

具体来说，VR-GCN 通过前一次训练批次中的节点特征近似当前批次的节点特征，但是如果仅仅这样会导致与真实值间会存在偏差，所以考虑通过采样部分真实值与近似值间的差值进

一步逼近真实的节点信息。那么，对于基于传统节点级采样得到的节点信息$\boldsymbol{H}^{(l+1)}=\sigma\left(\widehat{\boldsymbol{P}}^{(l)}\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)}\right)$，可以转换为以下方式：

$$\boldsymbol{H}^{(l+1)}=\sigma\left(\left(\widehat{\boldsymbol{P}}^{(l)}\left(\boldsymbol{H}^{(l)}-\widetilde{\boldsymbol{H}}^{(l)}\right)+\boldsymbol{P}^{(l)}\widetilde{\boldsymbol{H}}^{(l)}\right)\boldsymbol{W}^{(l)}\right)$$

其中，$\boldsymbol{P}$表示归一化后的邻接矩阵；$\widehat{\boldsymbol{P}}$表示采样后的邻接矩阵，即如果节点v在节点u的采样集合$\hat{n}^{(l)}(u)$中，那么$\widehat{\boldsymbol{P}}_{uv}^{(l)}=\frac{|N(u)|}{|\hat{n}^{(l)}(u)|}\boldsymbol{P}_{uv}$，否则$\widehat{\boldsymbol{P}}_{uv}^{(l)}=0$；$\widetilde{\boldsymbol{H}}$表示历史激活值，即前一次训练批次中的节点信息，用于近似当前批次中的节点信息。在每一轮训练批次结束时，历史激活值$\widetilde{\boldsymbol{h}}_v^{(l)}$通过$\boldsymbol{h}_v^{(l)}$直接赋值得到，以供下一次训练批次使用。随着训练进行，历史激活值$\widetilde{\boldsymbol{h}}_v^{(l)}$会逼近真实的隐藏特征$\boldsymbol{h}_v^{(l)}$从而使得$\boldsymbol{h}_v^{(l+1)}$逐渐收敛，通过这种方式可以保证在采样较少数量邻居的同时，减少采样导致的方差。

2) 层级采样

对于自适应的层次级采样，下面以 AS-GCN[38]为例介绍。它根据顶层节点采样底层节点，以自顶向下的方式在每一层采样固定数量的节点。但需要注意的是，下层的节点是根据上层的节点有条件地采样，而不是在每一层独立采样。而 AS-GCN 体现自适应的地方在于将采样概率参数化并通过目标函数最小化采样带来的方差。

首先，考虑基于层级采样得到节点嵌入的一般形式。由于下层的节点是根据上层的节点有条件地采样，可以将基于节点级采样的 GCN 改写成下面形式：

$$\boldsymbol{h}^{(l+1)}(v_i)=\sigma_{W^{(l)}}\left(N(v_i)\mu_q(v_i)\right)$$

$$\mu_q(v_i)=\frac{1}{n}\sum_{j=1}^{n}\frac{p\left(u_j|v_i\right)}{q\left(u_j|v_1,\ldots,v_n\right)}\boldsymbol{h}^{(l)}\left(u_j\right),\quad u_j\sim q\left(u_j|v_1,\ldots,v_n\right)$$

其中，$p\left(u_j|v_i\right)=\frac{\hat{a}(v_i,u_j)}{N(v_i)}$表示在给定$v_i$的条件下采样$u_j$的概率，可以通过在归一化邻接矩阵中的占比衡量；$q\left(u_j|v_1,\ldots,v_n\right)$表示在给定当前层所有节点（即$v_1,\ldots,v_n$）的条件下，采样下层节点$u_j$的概率；$n$为每层采样的节点的固定数量；$\mu_q(v_i)$即为通过层级采样得到的结果。那么，对于层级采样只需要确定采样概率$q\left(u_j|v_1,\ldots,v_n\right)$就能得到每一层的隐藏特征。由于需要保证方差是可控的，可以从最小化方差的角度考虑确定采样概率。下面将$q\left(u_j|v_1,\ldots,v_n\right)$简化成$q\left(u_j\right)$表示，则近似期望$\mu_q(v_i)$的方差和推导得到使方差最小的最佳采样概率$q^*\left(u_j\right)$为：

$$\mathrm{Var}_q\left(\mu_q(v_i)\right)=\frac{1}{n}\mathbb{E}_{q(u_j)}\left[\frac{\left(p(u_j|v_i)|\boldsymbol{h}^{(l)}\left(u_j\right)|-\mu_q(v_i)q\left(u_j\right)\right)^2}{q^2\left(u_j\right)}\right]$$

$$q^*\left(u_j\right)=\frac{\sum_{i=1}^{n}p\left(u_j|v_i\right)|\boldsymbol{h}^{(l)}\left(u_j\right)|}{\sum_{j=1}^{N}\sum_{i=1}^{n}p\left(u_j|v_i\right)|\boldsymbol{h}^{(l)}\left(u_j\right)|}$$

但是最佳采样概率无法计算，因为最佳采样概率需要先得到隐藏特征$\boldsymbol{h}^{(l)}\left(u_j\right)$。而一般过程是先通过采样概率进行层级采样，再开始按层计算隐藏特征，此时还未得到隐藏特征$\boldsymbol{h}^{(l)}\left(u_j\right)$，导致采样无法进行。因此，AS-GCN 用可学习的线性函数$g\left(x\left(u_j\right)\right)$近似替代不可计算的部分，其中$\boldsymbol{x}\left(u_j\right)$表示节点的$u_j$特征：

$$q^*\left(u_j\right)=\frac{\sum_{i=1}^{n}p\left(u_j|v_i\right)|g\left(\boldsymbol{x}\left(u_j\right)\right)|}{\sum_{j=1}^{N}\sum_{i=1}^{n}p\left(u_j|v_i\right)|g\left(\boldsymbol{x}\left(v_j\right)\right)|}$$

因此，可以通过定义$g\left(\boldsymbol{x}\left(u_j\right)\right)=\boldsymbol{W}_g\boldsymbol{x}\left(u_j\right)$来计算最佳采样概率。此外，在训练过程中，采样器引起的方差$\mathrm{Var}_q\left(\mu_q(v_i)\right)$被添加到损失函数中进行最小化。其自适应也就体现在将采样器参数化，从而自适应地减少方差，提高了稳定性。

5.2.3 图池化优化

在图神经网络的下游任务中，图级任务是很重要的一部分，通常要求模型理解整个图的语义和关系，并对输入的不同大小和拓扑结构的图结构进行整体的图级表示。因此，对于图级任务，池化机制是一个至关重要的部分，它将由图神经网络生成的节点表示压缩成单个向量。一些较为简单的池化机制如求和池化（Sum Pooling）和平均池化（Mean Pooling）忽略了邻居节点之间的差异性，导致模型无法捕捉到复杂的局部结构或重要的节点特征。在本小节，将探讨一些更为复杂的池化机制。这些池化大致可以分为平铺池化（Flat Pooling）和层级池化（Hierarchical Pooling），前者直接在一步中生成图级表示，后者考虑在池化过程中更好地保留图的层次结构信息，逐渐将一个图粗化为尺寸较小的图。

1. 平铺池化

平铺池化是在一步中直接生成图级别表示的池化操作，需满足两个要求：(1)在输入图的大小不同时输出固定大小图表示；(2)当输入图的节点顺序变化时输出相同的表示。显然，求和池化和平均池化等池化方式都属于平铺池化，下面介绍一些更为复杂的平铺池化。

1) 非线性变换

为了提高池化方法的表达能力，很多方法基于 DeepSet[39]提出的置换不变性（Permutation Invariance）理论对池化进行额外的非线性转换。以 GFN[40]为例，生成图级别表示包含两个部分：图过滤部分，执行基于图的邻居聚合操作；集合函数部分，将隐藏的节点特征组合起来进行预测。用于得到图级表示的读出函数（readout function）相当于是一个集合操作，要求输入顺序变化不影响结果的值，节点数量变化不影响结果的形状大小。将其中图过滤部分线性化得到$\boldsymbol{X}^G$，并基于 DeepSet 的结论定义了 GFN：

$$\boldsymbol{X}^G = \left[d, \boldsymbol{X}, \tilde{\boldsymbol{A}}\boldsymbol{X}, \tilde{\boldsymbol{A}}^2\boldsymbol{X}, ..., \tilde{\boldsymbol{A}}^K\boldsymbol{X}\right], \mathrm{GFN}(G, \boldsymbol{X}) = \rho\left(\sum_{v\in\mathcal{V}} \boldsymbol{\phi}\left(\boldsymbol{X}_v^G\right)\right)$$

其中，$d \in \mathbb{R}^{n\times 1}$是所有节点的度，$\tilde{A}$是归一化的邻接矩阵，最后将所有特征拼接成$\boldsymbol{X}^G$，这里也可以使用其他的传播算子。$\rho(\cdot)$和$\boldsymbol{\phi}(\cdot)$都由神经网络参数化。具体来说，将函数$\boldsymbol{\phi}(\cdot)$参数化为多层感知机（MLP），即$\boldsymbol{\phi}(\boldsymbol{x}) = \sigma\left(\sigma\left(\cdots\sigma\left(\boldsymbol{x}^T\boldsymbol{W}^{(1)}\right)\cdots\right)\boldsymbol{W}^{(T)}\right)$，$\rho(\cdot)$参数化为另一个 MLP。论文中通过实验证明了一个结论，线性化图过滤（即 GFN）对性能几乎没有影响，而同时线性化图过滤和集合函数（即 Graph Linear Network，$\mathrm{GLN}(G, \boldsymbol{X}) = \sigma(\boldsymbol{W}\sum_{v\in\mathcal{V}}\boldsymbol{X}_v^G)$）则导致性能下降。

2) 注意力机制

为了更好地利用不同节点的表示信息，很多方法引入了软注意机制（即权重是连续的概率分布）来确定最终图级表示中每个顶点的权重。下面以面向图的自注意力池化（Graph-focused self-attention）[42]和多层注意力池化（Multi-Level Attention Pooling）[43]为例介绍。

面向图的自注意力池化主要考虑到对于常规的池化（例如 Sum-pooling）存在不同的图产生相同的图级表示的情况，同时直接对图中所有节点表示求和缺乏对图中重要部分的关注。所以考虑利用自注意力机制来控制每个节点的重要性，使之更关注整个图中更重要的节点。其过程可以表述成如下形式：

$$\alpha_i^m = \mathrm{softmax}\left(\boldsymbol{W}_2^m\left(\sigma(\boldsymbol{W}_1^m\boldsymbol{h}_i + \boldsymbol{b}_1^m)\right) + \boldsymbol{b}_2^m\right)$$

$$g = \Vert_{m=1}^{M}\sigma\left(\sum_{i\in G}\alpha_i^m\left(\boldsymbol{W}_2^m\boldsymbol{h}_i + \boldsymbol{b}_2^m\right)\right)$$

其中，$\boldsymbol{W}_1^m, \boldsymbol{W}_2^m \in \mathbb{R}^{d_\alpha\times d_h}$，$\boldsymbol{b}_1^m, \boldsymbol{b}_2^m \in \mathbb{R}^{d_\alpha}$，$h_i$为节点$i$的特征，同时使用多头注意力，执行了 M 个独立的注意力机制来生成多头特征，最后将这些特征拼接作为最终的表示。

一般的图级任务，通常只将最后一层消息传递层的结果用于池化操作，但这样会影响GNN模型的表达能力，因为它在计算图表示时只能使用固定局部范围内的信息。所以，多层注意力池化对每个消息传递层都有一个专门的池化层，用于计算层级的图表示（如图5-11）。整体过程可以表述成以下两步：（1）对每一层GNN的结果进行池化：$\boldsymbol{h}_G^{(l)} = \text{Pool}^{(l)}\left(\left\{\boldsymbol{h}_n^{(l)} | n \in N\right\}\right), \forall l \in \{1, ..., L\}$。其中$L$表示GNN的层数，$N$表示节点集合，$h_G^{(l)}$表示每层得到的图表示，$\boldsymbol{h}_n^{(l)}$表示节点特征，$\boldsymbol{h}_G$表示最终的图表示。这里采用了注意力池化，表示如下：

$$\boldsymbol{h}_G^{(l)} = \sum_{n \in \mathcal{N}} \text{softmax}\left(f_{\text{gate}}^{(l)}\left(\boldsymbol{h}_n^{(l)}\right)\right)\boldsymbol{h}_n^{(l)} = \sum_{n \in \mathcal{N}} \frac{\exp\left(f_{\text{gate}}^{(l)}\left(\boldsymbol{h}_n^{(l)}\right)\right)}{\sum_{n' \in \mathcal{N}} \exp\left(f_{\text{gate}}^{(l)}\left(\boldsymbol{h}_{n'}^{(l)}\right)\right)}\boldsymbol{h}_n^{(l)}$$

其中，$f_{\text{gate}}^{(l)}$用来计算注意力分数，可以是神经网络。（2）将每层的池化结果进行聚合：$\boldsymbol{h}_G = f_{\text{agg}}\left(\left\{\boldsymbol{h}_G^{(l)} | l \in \{1, ..., L\}\right\}\right)$；$f_{\text{agg}}$可以是直接相加，也可以是加权相加来学习每一层的重要性。

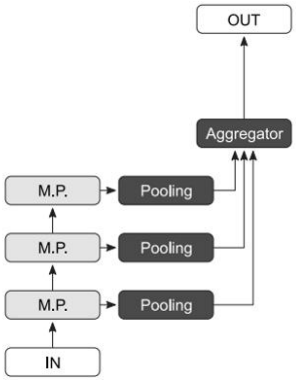


图5-11 多层注意力池化[43]

2. 层级池化

前面介绍的平铺池化方法，都是将所有的节点一步聚合为图嵌入，但它忽略了层级结构，对于某些具有层级结构特点的任务，可能会影响模型性能。层级池化方法旨在通过将原图逐步粗化为尺寸较小的图，来保留图的层次结构信息。根据其粗化图的方式，层级池化大致可分为节点聚类池化（Node Clustering Pooling）、节点丢弃池化（Node Drop Pooling）[41]。这两者的主要区别在于节点聚类池化为粗化后的图生成新节点，而节点丢弃池化会保留原始图中的节点。

1) 节点聚类池化

节点聚类池化将图池化视为一个节点聚类问题，将节点映射到一组簇中，这些簇被视为粗化后图的新节点。具体来讲，可以将节点聚类池化的过程分成两个模块[41]：簇分配矩阵（CAM）生成器：对于给定输入图，CAM生成器预测每个节点的软分配（一个节点可以属于多个簇）或者硬分配（一个节点只能属于一簇）；图粗化（COARSEN）：利用分配矩阵，得到新的特征矩阵和邻接矩阵，从而从原始图粗化得到新图。以上两个部分可以形式化表述为：

$$\boldsymbol{C}^{(l)} = \text{CAM}\left(\boldsymbol{X}^{(l)}, \boldsymbol{A}^{(l)}\right)$$

$$\boldsymbol{X}^{(l+1)}, \boldsymbol{A}^{(l+1)} = \text{COARSEN}\left(\boldsymbol{X}^{(l)}, \boldsymbol{A}^{(l)}, \boldsymbol{C}^{(l)}\right)$$

其中，$\boldsymbol{C}^{(l)} \in \mathbb{R}^{n_l \times n_{l+1}}$表示学习到的分配矩阵，$n_l$表示$l$层的簇数（节点数），$n_{l+1}$表示$l+1$层的簇数（节点数），分配矩阵表示第$l$层的每一个节点分配到第$l+1$层的每一个节点（簇）的概率。不同方法的主要区别在于生成CAM的部分，目前大部分方法的粗化部分都使用相同的方法：根据分配矩阵，对当前簇的嵌入加权求和得到合并后的节点表示$\boldsymbol{X}^{(l+1)} = \boldsymbol{C}^{(l)T}\boldsymbol{X}^{(l)} \in \mathbb{R}^{n_{l+1} \times d}$；通过对簇之间的边进行加权求和得到粗化后的邻接矩阵$\boldsymbol{A}^{(l+1)} = \boldsymbol{C}^{(l)T}\boldsymbol{A}^{(l)}\boldsymbol{C}^{(l)} \in \mathbb{R}^{n_{l+1} \times n_{l+1}}$，表示不同簇之间的连接强度。下面以DiffPool[13]为例说明：

对于CAM生成器部分，DiffPool直接使用了一个GNN模型来生成分配矩阵，并使用另一个GNN模型得到对应新的嵌入，对应过程可以表示如下：

$$\boldsymbol{C}^{(l)} = \text{softmax}\left(\text{GNN}_{l,\text{pool}}\left(\boldsymbol{A}^{(l)}, \boldsymbol{X}^{(l)}\right)\right)$$
$$\boldsymbol{Z}^{(l)} = \text{GNN}_{l,\text{embed}}\left(\boldsymbol{A}^{(l)}, \boldsymbol{X}^{(l)}\right)$$

其中，$\boldsymbol{A}^{(l)} \in \mathbb{R}^{n_l \times n_l}$表示第$\boldsymbol{l}$层时的邻接矩阵，$\boldsymbol{X}^{(l)} \in \mathbb{R}^{n_l \times d_l}$表示第$\boldsymbol{l}$层的节点嵌入，$\boldsymbol{C}^{(l)} \in \mathbb{R}^{n_l \times n_{l+1}}$表示学习到的分配矩阵，$\boldsymbol{Z}^{(l)} \in \mathbb{R}^{n_l \times d_l'}$表示学习到的新的节点嵌入，$n_{l+1}$为超参数。对于粗化部分，会根据前面两个不同 GNN 学习到的新嵌入和分配矩阵来计算，和前面介绍的稍有不同的是，使用的不是原来的嵌入$\boldsymbol{X}^{(l)}$而是学习到的新嵌入$\boldsymbol{Z}^{(l)}$：

$$\boldsymbol{X}^{(l+1)} = \boldsymbol{C}^{(l)T}\boldsymbol{Z}^{(l)} \in \mathbb{R}^{n_{l+1} \times d}$$
$$\boldsymbol{A}^{(l+1)} = \boldsymbol{C}^{(l)T}\boldsymbol{A}^{(l)}\boldsymbol{C}^{(l)} \in \mathbb{R}^{n_{l+1} \times n_{l+1}}$$

2) 节点丢弃池化

节点丢弃池化利用可学习的评分函数丢弃相对重要性评分较低的节点，从而得到结点数更少的图。具体来讲，可以将节点丢弃池化分成三个模块[41]：评分生成器会计算给定图中每个节点的重要性分数；节点选择器会选择重要性分数前 k 高的节点；图粗化会利用选择出来的节点，得到新的特征矩阵和邻接矩阵，从而由原始图粗化得到新图。以上三个部分可以形式化表述为：

$$\boldsymbol{S}^{(l)} = \text{SCORE}\left(\boldsymbol{X}^{(l)}, \boldsymbol{A}^{(l)}\right)$$
$$\text{idx}^{(l+1)} = \text{TOP}_k\left(\boldsymbol{S}^{(l)}\right)$$
$$\boldsymbol{X}^{(l+1)}, \boldsymbol{A}^{(l+1)} = \text{COARSEN}\left(\boldsymbol{X}^{(l)}, \boldsymbol{A}^{(l)}, \boldsymbol{S}^{(l)}, \text{idx}^{(l+1)}\right)$$

其中，不同方法在这三个过程中使用的函数 SCORE、TOP_k、COARSEN 不同，$\boldsymbol{S}^{(l)} \in \mathbb{R}^{n \times 1}$表示重要性分数，$\text{TOP}_k$用于排序并返回$\boldsymbol{S}^{(l)}$中最大的 k 个值对应的索引，$\text{idx}^{(l+1)}$表示在新图中保留的节点的索引。很多方法倾向于设计更复杂的评分生成器和更合理的图粗化方式，来选择更具代表性的节点并保留更重要的结构信息，下面以 TopKPool[14]和 GSAPool[46]为例介绍：

TopKPool 以一种相对直接的方式实现节点丢弃池化。对于评分生成器部分，通过不同节点在投影向量上的投影值来度量重要性，即$\boldsymbol{S}^{(l)} = \boldsymbol{X}^{(l)}p^{(l)}/\| p^{(l)} \|$，其中$p^{(l)}$表示第$\boldsymbol{l}$层的可学习的投影向量；对于节点选择器部分，同样通过排序$\boldsymbol{S}^{(l)}$并返回其中最大的 k 个值对应的索引来选择需要保留的节点；对于图粗化部分，新的邻接矩阵由筛选出来的节点构成（即$\boldsymbol{A}^{(l+1)} = \boldsymbol{A}^{(l)}_{\text{idx,idx}}$）,而新的特征矩阵由筛选出来的节点特征和其对应重要性（投影值）相乘得到，即$\boldsymbol{X}^{(l+1)} = \boldsymbol{X}^{(l)}_{\text{idx}} \odot \sigma\left(\boldsymbol{S}^{(l)}_{\text{idx}}\right)$，其中$\sigma(\cdot)$为非线性变换。相比 TopKPool，GSAPool 考虑了更多的信息。对于评分生成器考虑了两类信息，一类通过 GNN 得到结构信息来衡量重要性（即$\boldsymbol{S}_1^{(l)} = \sigma\left(\text{GNN}\left(\boldsymbol{X}^{(l)}, \boldsymbol{A}^{(l)}\right)\right)$），另一类根据节点特征信息衡量重要性（即$\boldsymbol{S}_2^{(l)} = \sigma\left(\text{MLP}\left(\boldsymbol{X}^{(l)}\right)\right)$），而总体重要性经超参数$\alpha$加权得到（即$\boldsymbol{S}^{(l)} = \alpha\boldsymbol{S}_1^{(l)} + (1-\alpha)\boldsymbol{S}_2^{(l)}$）；对于图粗化部分，新的特征矩阵先考虑在原图上进行一次聚合再选择所需节点，即$\boldsymbol{X}^{(l+1)} = \left(\boldsymbol{A}^{(l)}\boldsymbol{X}^{(l)}\boldsymbol{W}^{(l)}\right)_{\text{idx}} \odot \boldsymbol{S}^{(l)}_{\text{idx}}$，其中$\boldsymbol{W}^{(l)}$为可学习的权重矩阵，从而将部分节点选择器未选到的节点信息也保留到了池化后的图中。

可以看到，TopKPool 和 GSAPool 考虑的方面略有不同。对于评分生成器，TopKPool 只从节点特征一个角度考虑，GSAPool 从局部角度（利用节点特征信息）和全局角度（利用 GNN 得到结构信息）考虑；对于图粗化，TopKPool 直接从所选的节点得到新图，GSAPool 使用了所选和非所选的节点得到新图，保留了更多的结构和特征信息。

5.3 训练优化

在机器学习模型的训练过程中，训练策略的选择直接影响模型的性能表现和训练效率。传统的监督学习依赖于大量标注数据，而在许多场景下，获取足够的高质量标签数据极为困难且成本昂贵。同时，模型在训练早期直接面对复杂样本，可能会陷入局部最优或者训练效率低下。这一节将重点探讨训练优化策略中的自监督学习（Self-Supervised Learning）和课程学习（Curriculum Learning），前者通过设计代理任务挖掘数据中的隐含信息，后者则通过逐步增加训练任务的难度提高模型的学习效果。通过理解这些方法背后的原理与实现过程，可以更有

效地应用这些技术提升模型的表现。

5.3.1 图自监督学习

自监督学习是无需标注数据的一种训练方式，它通过设计代理任务从未标注数据中自动生成监督信息，通过这种构造的监督信息对模型进行训练。同样地，在图自监督学习中，通过充分利用节点特征信息和图结构信息，也可以有效地提取数据的潜在特征表示。在本小节，将探讨图自监督学习的几种主要方法，包括对比式、生成式自监督学习，每种方法都通过不同的任务和目标函数实现自监督的特征提取。

1. 对比式自监督

受到对比学习在 CV 和 NLP 领域进展的启发，很多工作开始提出将对比学习应用到图数据上。对比学习的核心思想是通过对比正负样本来学习通用特征。实际中，通常通过数据增强为每个样本生成多个视图，对于某一样本来讲，其生成的视图与之构成正样本对，其他样本生成的视图与此样本构成负样本对。对比学习的目标就是最大化正样本对的一致性，最小化负样本的一致性，而一致性一般由互信息（Mutual Information, MI）衡量。具体来讲，给定一个图$G=(\boldsymbol{A},\boldsymbol{X})$，可以对其应用$K$种不同的变换$T_1,T_2,\dots,T_K$，从而获得多个视图:$\boldsymbol{A}_k,\boldsymbol{X}_k=T_k(A,X),k=1,2,\dots,K$; 其次，一组图编码器$\{f_{\theta_k}\}_{k=1}^K$（可能相同或共享权重）可用于生成每个视图的不同表示$\boldsymbol{h}_k=f_{\theta_k}(\boldsymbol{A}_k,\boldsymbol{X}_k),k=1,2,\dots,K$，而对比学习会通过最大化同一样本的不同视图之间的互信息来优化目标：

$$\max_{\theta_1,\theta_2,\dots,\theta_K}\sum_i\sum_{j\neq i}\alpha_{i,j}\,\mathrm{MI}(\boldsymbol{h}_i,\boldsymbol{h}_j)$$

其中，$i,j\in\{1,2,\dots,K\}$，$\{\boldsymbol{h}_i\}_{i=1}^K$是从图$G=(\boldsymbol{A},\boldsymbol{X})$生成的表示，被视为正样本对；$\alpha_{i,j}\in\{0,1\}$具体取值在不同方法中不同；$\mathrm{MI}(\boldsymbol{h}_i,\boldsymbol{h}_j)$是两个表示向量$\boldsymbol{h}_i$和$\boldsymbol{h}_j$之间的互信息，互信息是概率论和信息论中重要的概念，它表示的是一个随机变量中包含另一个随机变量的信息量，可以理解成两个随机变量之间的相关程度。根据不同的代理任务，$\{\boldsymbol{h}_k\}_{k=1}^K$并不一定是同一层次的表示，还可以包含不同的层次的表示，比如节点级别、图级别。对于负样本，可以由另一个图$\widetilde{G}=(\widetilde{\boldsymbol{A}},\widetilde{\boldsymbol{X}})$生成的表示$\{\widetilde{\boldsymbol{h}}_j\}_{i=1}^K$得到。

总的来说，图对比学习的过程可以分为三个部分[47]：(1)数据增强（Data Augmentation），通过对原始图进行随机扰动得到多样化的样本，在大多数情况下将其作为正样本；(2)代理任务（Pretext Tasks），通常与主任务并无直接关联，为模型提供训练信号进行辅助；(3)目标函数（Contrastive Objectives），定义模型学习的优化目标。

1) 数据增强

近几年的研究表明，对比学习的成功在很大程度上依赖于设计良好的数据增强策略。在这里，将用于图数据的数据增强策略分为四类[47]：基于特征（Feature-based）、基于结构（Structue-based）、基于采样（Sampling-based）和自适应（Adaptive）增强。

基于特征增强（Feature-based Augmentation）只会在节点特征或者边特征进行变换，对于输入图$G=(\boldsymbol{A},\boldsymbol{X})$，即有$\widetilde{\boldsymbol{A}},\widetilde{\boldsymbol{X}}=T(\boldsymbol{A},\boldsymbol{X})=\boldsymbol{A},T_X(\boldsymbol{X})$。其中$T_X(\cdot)$有以下几种方法：

(1) 特征屏蔽（Attribute Masking）：随机屏蔽一小部分属性，屏蔽的部分可以用常量或随机量替换，也可以直接在原属性上添加随机噪声。需要注意的是屏蔽的方式可以是对一个节点特征其中几维进行屏蔽，而不是针对所有维进行屏蔽（如图 5-12）。

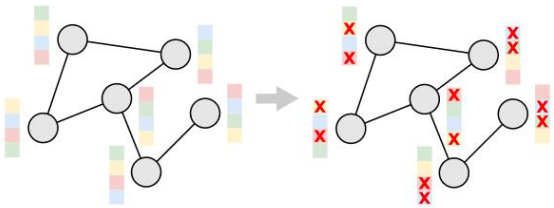


图 5-12 特征屏蔽

(2) 特征重排（Attribute Shuffling）：随机打乱节点特征顺序。增强后的图包含与原始图相同的节点，但它们在图中的位置发生了变化，从而接收不同的上下文信息。

基于结构增强（Structue-based Augmentation）只会在邻接矩阵上进行变换，即$\widetilde{\boldsymbol{A}}, \widetilde{\boldsymbol{X}} = T(\boldsymbol{A}, \boldsymbol{X}) = T_A(\boldsymbol{A}), \boldsymbol{X}$。其中$T_A(\cdot)$有以下几种方法：

(1) 边扰动（Edge Perturbation）：随机添加或去掉一定比例的边，通常通过随机地选择一些节点对，如果节点对间存在边就去掉此边，不存在就添加此边。

(2) 节点插入（Node Insertion）：将一定数量的节点插入到原节点集合中，并在这些节点之间以及它们与原节点集合之间随机添加一定比例的边。

(3) 边扩散（Edge Diffusion）：将不同距离的邻居直接当作邻居节点，具体来说可以用以下形式进行表示：

$$T_A(\boldsymbol{A}) = \sum_{k=0}^{\infty} \Theta_k \boldsymbol{S}^k$$

其中，$\boldsymbol{S} \in \mathbb{R}^{N \times N}$是广义转移矩阵；$\Theta$是加权系数，满足$\sum_{k=0}^{\infty} \Theta_k = 1$，$\Theta_k \in [0,1]$，用来调整不同距离$k$的邻居节点对扩散结果的影响程度。例如在 Personalized PageRank（PPR）扩散中，广义转移矩阵为归一化后的邻接矩阵$\boldsymbol{S} = \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$，加权系数为$\Theta_k = \alpha(1-\alpha)^k$，随着距离 k 增加，其对应的系数也相应减小，这使得距离越远的节点对扩散结果的影响越小。同时，如果按照原式计算需要通过无穷项相加才能得到扩散结果，这是无法直接计算的，所以考虑推导原式得到以下 PPR 的闭式解（Closed-form Solution）：

$$T_A^{PPR}(\boldsymbol{A}) = \alpha\left(\boldsymbol{I_n} - (1-\alpha)\boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}\right)^{-1}$$

基于采样增强（Sampling-based Augmentation）会在邻接矩阵和特征上都进行变换，即$\widetilde{\boldsymbol{A}}, \widetilde{\boldsymbol{X}} = T(\boldsymbol{A},) = \boldsymbol{A}[S, S], \boldsymbol{X}[S, :]$。其中$S \in V$，通常有下面几种策略进行采样来得到$S$：

(1) 均匀采样（Uniform Sampling）：等概率地保留给定数量的节点。从另一个角度讲，相当于等概率地删除一定比例的节点，所以也被称为节点删除。

(2) 自我网络采样（Ego-nets Sampling）：以某个节点v_i为中心，采样所有最短距离小于等于 L 的邻居作为最终的采样结果。

(3) 随机游走采样（Random Walk Sampling）：以节点v_i为起点在图g上进行随机游走。游走时迭代地遍历邻居，走向某一邻居的概率与边权重成正比，每一步并以概率α返回到起始节点v_i。最终，访问的节点被收集到一个节点子集S中。

(4) 重要性采样（Importance Sampling）：重要性采样会基于特定节点的邻居节点的重要性，采样得到子图。通常对于给定的节点v_i，选择以v_i为锚点的前k个重要邻居节点组成一个子图，其中重要性可以通过 Personalized PageRank 算法的方式进行评分：$\boldsymbol{M} = \alpha \cdot (I_n - (1-\alpha) \cdot \boldsymbol{A}\boldsymbol{D}^{-1})$，其中$\boldsymbol{M}$为重要性得分矩阵，$\alpha \in [0,1]$，则$\boldsymbol{M}(i,:)$表示其它节点相对于节点$v_i$的重要性，那么采样的子图即包含其中最大的 k 个值对应的节点。

基于自适应增强（Adaptive Augmentation）会根据训练过程中的信息动态地进行数据增强，下面主要介绍两种方法：

(1) 基于重要性：通常根据训练过程中的信息对节点或边进行重要性评分，并据此进行数据增强。例如 GCA[48]，它保持重要的结构和特征不变，而扰动相对不重要的边和特征，从而以重要性决定边删除和特征屏蔽的概率。比如对于边删除的概率，先考虑给定一个节点中心性的度量函数$\varphi_c(\cdot): V \to \mathbb{R}^+$，那么边的中心性（重要性）定义为两个相邻节点中心性分数的平均

值，即 $s_{i,j}=\log\frac{\varphi_c(v_i)+\varphi_c(v_j)}{2}$；进而，可以定义边$e_{i,j}$的删除概率为：

$$p_{i,j}=\min\left(\frac{s_{\max}-s_{i,j}}{s_{\max}-\mu_s}\cdot p_e,p_\tau\right)$$

其中，p_e是一个超参数，控制删除边的总体概率；$s_{\max}$和μ_s分别是$s_{i,j}$的最大值和平均值，$p_\tau<1$ 是一个截断概率，用来截断过高的概率，因为过高的删除概率会严重破坏图结构。同时，节点中心性$\varphi_c(\cdot)$可以定义为度中心性、特征向量中心性或 PageRank 中心性。

(2) 基于梯度：GROC[49]利用梯度信息对边进行了数据增强。首先确定候选集，对于给定的节点v_i，边删除候选集和边插入候选集分别定义为：

$$S^-=\left\{(v_i,v_k)|v_k\in N_i^{(l)}\right\},\quad S^+=\left\{(v_i,v_k)|v_k\in\left(\bigcup_{v_m\in B}N_m^{(l)}\setminus N_i^{(l)}\right)\right\}$$

其中，$N_i^{(l)}$表示锚节点v_i的l-跳邻域内节点集合；$B\subset V$为一个节点批次。那么，S^-表示节点v_i与其l-跳邻域内的节点之间的边集合；S^+为一组边(v_i,v_k)，其中v_i是锚节点，而v_k位于某些其它锚节点$v_m(v_m\neq v_i)$的l-跳邻域内，但不在节点v_i的l-跳邻域内。然后，再考虑如何利用梯度信息作为删除和添加的衡量标准。具体来说，对图$G=(\boldsymbol{A},\boldsymbol{X})$应用两个随机变换 $T_1(\cdot)$和$T_2(\cdot)$，从而生成两个视图。在生成这些视图的过程中，会以概率r_1和r_2独立地对节点特征进行屏蔽。然后，计算这两个视图之间的对比损失L_{ssl}。对损失L_{ssl}进行反向传播，以获取集合S^-和S^+中每条边的梯度强度值。通过移除S^-中具有最小梯度幅值的部分边，并插入S^+中具有最大梯度幅值的部分边，从而进一步应用基于梯度的自适应增强。

2) 代理任务

代理任务是一种间接设计的任务，旨在通过从丰富的无标签数据中学习可迁移的知识，从而能将这些知识迁移到具有特定监督信号的下游任务中，以实现特定的训练目标。代理任务可以利用不同层面视图进行对比学习，其中视图规模可以是局部的（Local）、上下文的（Context）或全局的（Global），在图中对应于节点级别、子图级别或图级别的信息。下面将从这几个对比角度介绍其中一些代表性的方法：全局-全局对比、上下文-上下文对比、局部-局部对比、局部-全局对比、局部-上下文对比、上下文-全局对比。

(1) 全局-全局对比（Global-Global Contrasting）

GraphCL[50]：通过四种数据增强策略，得到数据增强后的图，目标是学习这些图是否来自同一个图。具体来说，为了结合不同的先验知识，考虑了四种图增强方法$\{T_k\}_{k=1}^4$：节点删除操作$T_1(\cdot)$、边扰动操作$T_2(\cdot)$、特征屏蔽操作$T_3(\cdot)$、子图采样操作$T_4(\cdot)$。对于一个给定图$g_i=(\boldsymbol{A}_i,\boldsymbol{X}_i)\in G$，首先从$\{T_k\}_{k=1}^4$中随机选择一系列图增强操作$T(\cdot)$来生成一个增强后的图$\tilde{g}_i=\left(\widetilde{\boldsymbol{A}}_i,\widetilde{\boldsymbol{X}}_i\right)=T(\boldsymbol{A}_i,\boldsymbol{X}_i)$。然后，使用相同的图级别编码器$f_\gamma(\cdot)$用于获取图级别的表示，其中$\boldsymbol{h}_{g_i}=f_\gamma(\boldsymbol{A}_i,\boldsymbol{X}_i)$和$\widetilde{\boldsymbol{h}}_{g_i}=f_\gamma\left(\widetilde{\boldsymbol{A}}_i,\widetilde{\boldsymbol{X}}_i\right)$，分别表示原始图和增强图的表示。最终，目标是学习预测两个增强后的图是否源自同一个图，有如下定义：

$$\max_\theta\frac{1}{|\mathrm{G}|}\sum_{g_i\in\mathrm{G}}\mathrm{MI}\left(\boldsymbol{h}_{g_i},\widetilde{\boldsymbol{h}}_{\tilde{g}_i}\right)$$

Label Contrastive Coding (LCC) [20]：用于监督对比学习，将同类样本作为正样本对、不同类样本作为负样本对来进行对比学习。具体来说，查询图(g_q,y_q)和键图(g_k,y_k)通过两个不同的图级编码器$f_{\gamma_q}(\cdot)$和$f_{\gamma_k}(\cdot)$获得图级表示$\boldsymbol{h}_{g_q}$和$\boldsymbol{h}_{g_k}$。如果$\boldsymbol{h}_{g_q}$和$\boldsymbol{h}_{g_k}$具有相同的标签，它们被视为正样本对，否则为负样本对。对于编码的查询(g_q,y_q)，其标签对比损失通过以下公式计算：

$$\max_{\gamma_q}\log\frac{\sum_{i=1}^m\mathbb{I}_{y_i=y_q}\cdot\exp\left(\boldsymbol{h}_q\cdot\boldsymbol{h}_k^{(i)}/\tau\right)}{\sum_{i=1}^m\exp\left(\boldsymbol{h}_q\cdot\boldsymbol{h}_k^{(i)}/\tau\right)}$$

其中，m是动态标签记忆库的大小；τ是温度超参数，当τ较大时，样本之间的相似度差异被缩小，输出的概率分布更加平滑，样本的区分度较低；$\mathbb{I}_{y_i=y_q}$用于指示记忆库中第i个键图$g_k^{(i)}$的标签是否与y_q相同。参数γ_k遵循基于动量（Momentum）的更新机制，更新公式为：$\gamma_k\leftarrow$

$\alpha\gamma_k + (1-\alpha)\gamma_q$，其中$\alpha \in [0,1]$是控制$\gamma_k$更新速度的动量权重。

(2) 上下文-上下文对比（Context-Context Contrasting）

Graph Contrastive Coding (GCC)[52]：将图结构中的不同部分视为学习目标，使得结构相似的子图具有相近的表示，结构不同的则有较大的表示差异，捕捉多个图中的通用图拓扑属性。GCC 和前面提到的 Label Contrastive Coding 极为类似，但不需要标签。具体来说，它首先通过随机游走为每个图$g \in G$采样多个子图，并将其收集到一个记忆库S中。然后，查询子图 $g_q \in S$ 和键子图$g_k \in S$通过两个图级别编码器$f_{\gamma_q}(\cdot)$和$f_{\gamma_k}(\cdot)$编码，分别得到图级表示 $\boldsymbol{h}_{g_q}$和$\boldsymbol{h}_{g_k}$。如果g_q和g_k来自同一个图，它们被视为正样本对，否则为负样本对（如图 5-13）。对于编码的查询(g_q, y_q)，其中y_q是查询图所采样的原始图的索引，其图对比损失计算公式如下：

$$\max_{\gamma_q} \log \frac{\sum_{i=1}^{|S|} \mathbb{I}_{y_i=y_q} \cdot \exp\left(\boldsymbol{h}_{g_q} \cdot \boldsymbol{h}_{g_k}^{(i)}/\tau\right)}{\sum_{i=1}^{|S|} \exp\left(\boldsymbol{h}_{g_q} \cdot \boldsymbol{h}_{g_k}^{(i)}/\tau\right)}$$

其中，$\mathbb{I}_{y_i=y_q}$用于指示记忆库中的第i个键图$g_k^{(i)}$与查询图g_q是否来自同一个图。

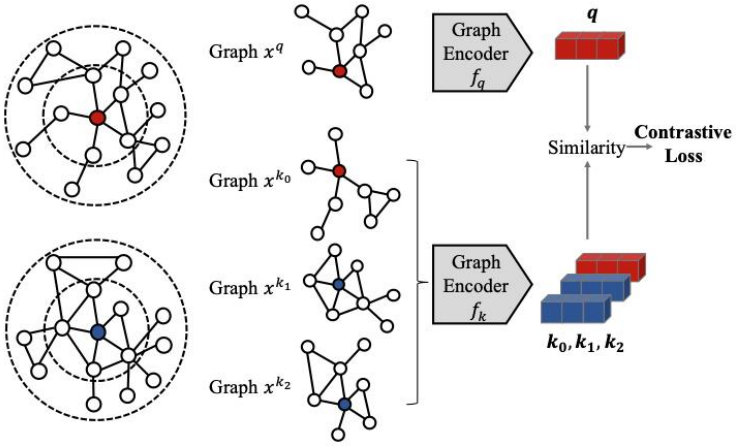


图 5-13 GCC 示意图[52]

(3) 局部-局部对比（Local-Local Contrasting）

GRACE[22]：直接使相同节点的不同视图表示尽可能相近，而不同节点的表示尽可能区分开。具体来说，对于图$g = (\boldsymbol{A}, \boldsymbol{X})$，首先生成两个数据增强后的图$G^{(1)} = \left(\boldsymbol{A}^{(1)}, \boldsymbol{X}^{(1)}\right) = T_1(\boldsymbol{A}, \boldsymbol{X})$和$G^{(2)} = \left(\boldsymbol{A}^{(2)}, \boldsymbol{X}^{(2)}\right) = T_2(\boldsymbol{A}, \boldsymbol{X})$。然后，通过相同的编码器$f_\theta(\cdot)$来生成它们的节点嵌入表示$\boldsymbol{H}^{(1)} = f_\theta\left(\boldsymbol{A}^{(1)}, \boldsymbol{X}^{(1)}\right)$和 $\boldsymbol{H}^{(2)} = f_\theta\left(\boldsymbol{A}^{(2)}, \boldsymbol{X}^{(2)}\right)$。对于对比目标（如图 5-14），来源于同一张图的两个增强图中相同的节点视为正样本对，即$\left(\boldsymbol{h}_i^{(1)}, \boldsymbol{h}_i^{(2)}\right)$;不同视图或同一视图的不同节点视为负样本对，即$\left(\boldsymbol{h}_i^{(1)}, \boldsymbol{h}_k^{(2)}\right)$ 或$\left(\boldsymbol{h}_i^{(1)}, \boldsymbol{h}_k^{(1)}\right)$。最终，每对正样本$\left(\boldsymbol{h}_i^{(1)}, \boldsymbol{h}_i^{(2)}\right)$的目标定义为：

$$L\left(\boldsymbol{h}_i^{(1)}, \boldsymbol{h}_i^{(2)}\right) = \log \frac{\exp\left(D\left(\boldsymbol{h}_i^{(1)}, \boldsymbol{h}_i^{(2)}\right)/\tau\right)}{\exp\left(D\left(\boldsymbol{h}_i^{(1)}, \boldsymbol{h}_i^{(2)}\right)/\tau\right) + Neg}$$

$$Neg = \sum_{k=1, k\neq i}^{N} \left[\exp\left(D\left(\boldsymbol{h}_i^{(1)}, \boldsymbol{h}_k^{(1)}\right)/\tau\right) + \exp\left(D\left(\boldsymbol{h}_i^{(1)}, \boldsymbol{h}_k^{(2)}\right)/\tau\right)\right]$$

其中，$\exp\left(D\left(\boldsymbol{h}_i^{(1)}, \boldsymbol{h}_k^{(1)}\right)/\tau\right)$表示视图内负样本对，$\exp\left(D\left(\boldsymbol{h}_i^{(1)}, \boldsymbol{h}_k^{(2)}\right)/\tau\right)$表示视图间负样本对；判别器$D(\cdot,\cdot)$用于计算一致性得分。那么，最终要最大化的总体目标定义为：

$$\max_\theta \frac{1}{2N} \sum_{i=1}^{N} \left[L\left(\boldsymbol{h}_i^{(1)}, \boldsymbol{h}_i^{(2)}\right) + L\left(\boldsymbol{h}_i^{(2)}, \boldsymbol{h}_i^{(1)}\right)\right]$$

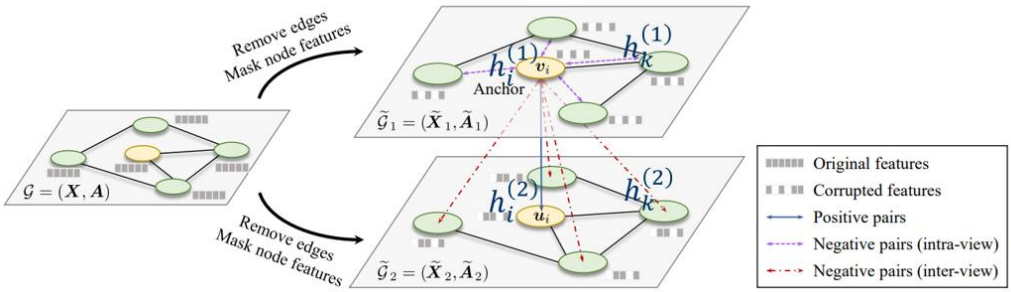


图 5-14 GRACE 示意图[22]

(4) 局部-全局对比（Local-Global Contrasting）

Deep Graph Infomax (DGI)[54]：考虑通过最大化局部节点表示与全局图表示之间的互信息，来训练编码器学习节点的有效表示（如图 5-15）。具体来说，首先使用增强转换$T(\cdot)$来获得增强图$\widetilde{G}=(\widetilde{\boldsymbol{A}},\widetilde{\boldsymbol{X}})=T(\boldsymbol{A},\boldsymbol{X})$。接着，将这两个图输入到图编码器$f_\theta(\cdot)$中，分别得到节点嵌入矩阵$\widetilde{\boldsymbol{H}}=f_\theta(\boldsymbol{A},\widetilde{\boldsymbol{X}})$和$\boldsymbol{H}=f_\theta(\boldsymbol{A},\boldsymbol{X})$。然后，READOUT 函数用于获得图表示$\boldsymbol{h}_g=\text{READOUT}(\boldsymbol{H})$。最后，学习目标定义如下：

$$\max_\theta \frac{1}{N}\sum_{v_i\in\mathcal{V}} \text{MI}\left(\boldsymbol{h}_g,\boldsymbol{h}_i\right)$$

其中，$\boldsymbol{h}_i$是节点v_i的节点嵌入；负样本用于与$\boldsymbol{h}_g$进行对比，定义为$\text{Neg}(\boldsymbol{h}_g)=\{\widetilde{\boldsymbol{h}}_j\}_{v_j\in\mathcal{V}}$，即与增强图中的节点嵌入进行对比。实际的目标函数会采用 Jensen-Shannon 散度，可以有效的最大化正例的互信息，这部分会在目标函数部分介绍。

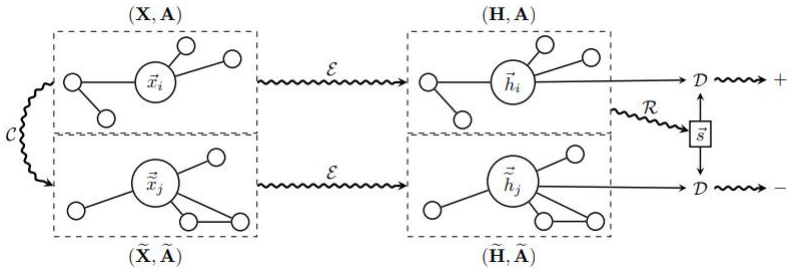


图 5-15 DGI 示意图[54]

(5) 局部-上下文对比（Local-Context Contrasting）

SUBG-CON[55]：通过对比锚点节点表示和以其为中心的子图表示，实现并行化的大规模自监督图表示学习（如图 5-16）。具体来说，给定一个图$G=(\boldsymbol{A},\boldsymbol{X})$，SUBG-CON 首先从节点集合$V$中选取一个锚点节点集$S$，然后通过重要性采样策略对其上下文子图进行采样得到$\{g_i=(\boldsymbol{A}^{(i)},\boldsymbol{X}^{(i)})\}_{i=1}^{|S|}$；接着，对这些子图应用一个相同的图编码器$f_\theta(\cdot)$和一个 READOUT 函数，以获取节点嵌入矩阵$\{\boldsymbol{H}^{(1)},\boldsymbol{H}^{(2)},...,\boldsymbol{H}^{(|S|)}\}$，其中：$\boldsymbol{H}^{(i)}=f_\theta(\boldsymbol{A}^{(i)},\boldsymbol{X}^{(i)})$；而图级别的表示为$\{\boldsymbol{s}_{g_1},\boldsymbol{s}_{g_2},\cdots,\boldsymbol{s}_{g_{|S|}}\}$，其中：$\boldsymbol{s}_{g_i}=\text{READOUT}(\boldsymbol{H}^{(i)})$；正样本对为锚点节点表示和其对应生成的子图表示，那么目标定义为最大化锚点节点和其对应子图的表示的互信息：

$$\max_\theta \frac{1}{|S|}\sum_{v_i\in S} \text{MI}\left(\boldsymbol{h}_i^{(i)},\boldsymbol{s}_{g_i}\right)$$

其中，$\boldsymbol{h}_i^{(i)}$是在节点嵌入矩阵$\boldsymbol{H}^{(i)}$中的锚节点v_i的表示。负样本用于与$\boldsymbol{h}_i^{(i)}$进行对比，其定义为$\text{Neg}\left(\boldsymbol{h}_i^{(i)}\right)=\{\boldsymbol{s}_{g_j}\}_{v_j\in S,j\neq i}$，即负样本对为和其他锚点生成的子图表示。

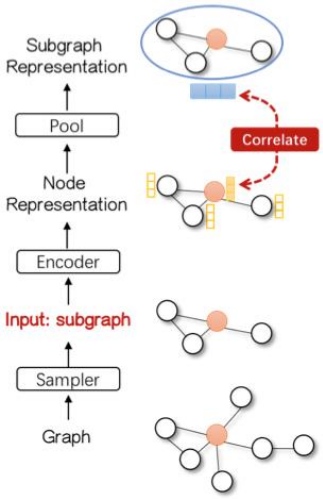


图 5-16 SUBG-CON 示意图[55]

(6) 上下文-全局对比（Context-Global Contrasting）

InfoGraph[56]：考虑通过多层图编码器得到子图信息，从而最大化子图与全局图表示之间的互信息，以实现自监督学习。具体来说，给定一个图$g=(\boldsymbol{A},\boldsymbol{X})$，首先通过数据增强得到增强图$\tilde{g}=(\tilde{\boldsymbol{A}},\tilde{\boldsymbol{X}})$，然后通过一个相同的$L$-层图编码器$f_\theta(\cdot)$得到节点嵌入矩阵序列$\{\boldsymbol{H}^{(l)}\}_{l=1}^{L}$和$\{\tilde{\boldsymbol{H}}^{(l)}\}_{l=1}^{L}$；然后，将从每一层学习到的表示进行拼接，相当于得到从节点v_i采样L层邻居的子图表示：$\boldsymbol{h}_i=\text{CONCAT}\left(\{\boldsymbol{h}_i^{(l)}\}_{l=1}^{L}\right)$，$\tilde{\boldsymbol{h}}_i=\text{CONCAT}\left(\{\tilde{\boldsymbol{h}}_i^{(l)}\}_{l=1}^{L}\right)$，其中，$\boldsymbol{h}_i^{(l)}$是节点$v_i$在第$l$层编码器中学习到的嵌入；最后，同样通过一个 READOUT 函数获取图级别的表示：$\boldsymbol{s}_g=\text{READOUT}\left(\{\boldsymbol{h}_i\}_{i=1}^{N}\right)$；正样本对为同一张图的子图表示和图级表示，负样本为不属于此张图的子图表示。学习目标定义为最大化子图表示和图级表示的互信息：

$$\max_{\theta}\sum_{g\in\mathrm{G}}\frac{1}{|g|}\sum_{v_i\in g}\text{MI}\left(\boldsymbol{s}_g,\boldsymbol{h}_i\right)$$

其中，用于与$\boldsymbol{s}_g$进行对比的负样本即为增强图中节点表示$\{\tilde{\boldsymbol{h}}_i\}_{v_i\in\mathcal{V}}$，这里同样可以使用 Jensen-Shannon 进行互信息估计。

3) 目标函数

对比学习优化的主要方式是将两个表示$\boldsymbol{h}_i$和$\boldsymbol{h}_j$看作随机变量，并通过最大化它们的互信息来实现，即：$\text{MI}(\boldsymbol{h}_i,\boldsymbol{h}_j)=\mathbb{E}_{p(\boldsymbol{h}_i,\boldsymbol{h}_j)}\left[\log\frac{p(\boldsymbol{h}_i,\boldsymbol{h}_j)}{p(\boldsymbol{h}_i)p(\boldsymbol{h}_j)}\right]$。为了便于计算对比学习中的互信息，通常通过最大化互信息的下界间接地最大化互信息。下面介绍三种常见的互信息下界形式：

(1) Donsker-Varadhan (DV) Estimator[57]：

$$\text{MI}_{DV}(\boldsymbol{h}_i,\boldsymbol{h}_j)=\mathbb{E}_{p(\boldsymbol{h}_i,\boldsymbol{h}_j)}[D(\boldsymbol{h}_i,\boldsymbol{h}_j)]-\log\mathbb{E}_{p(\boldsymbol{h}_i)p(\boldsymbol{h}_j)}\left[\exp\left(D(\boldsymbol{h}_i,\boldsymbol{h}_j)\right)\right]$$

DV 下界在理论上较为紧凑，但其训练过程可能会遇到梯度爆炸或不稳定现象。其中，$p(\boldsymbol{h}_i,\boldsymbol{h}_j)$表示两个视图$\boldsymbol{h}_i$和$\boldsymbol{h}_j$的联合分布，而$p(\boldsymbol{h}_i)p(\boldsymbol{h}_j)$表示边缘分布的乘积。$D:\mathbb{R}^q\times\mathbb{R}^q\to\mathbb{R}$将两个视图$\boldsymbol{h}_i$和$\boldsymbol{h}_j$映射为一个一致性得分。通常，可以选择先将$\boldsymbol{h}_i$映射为$\boldsymbol{z}_i=g_\omega(\boldsymbol{h}_i)$，然后再利用判别器$D$计算一致性得分，其中$g_\omega(\cdot)$可以是线性映射、非线性映射、恒等映射（即$\boldsymbol{z}_i=\boldsymbol{h}_i$）。判别器$D$可以采取各种形式，例如：标准的内积$D(\boldsymbol{z}_i,\boldsymbol{z}_j)=\boldsymbol{z}_i^T\boldsymbol{z}_j$、余弦相似度$D(\boldsymbol{z}_i,\boldsymbol{z}_j)=\frac{\boldsymbol{z}_i^T\boldsymbol{z}_j}{\|\boldsymbol{z}_i\|\|\boldsymbol{z}_j\|}$和高斯相似度$D(\boldsymbol{z}_i,\boldsymbol{z}_j)=\exp\left(-\frac{\|\boldsymbol{z}_i-\boldsymbol{z}_j\|_2^2}{2\sigma^2}\right)$等。

(2) Jensen-Shannon (JS) Estimator[58]：

$$\text{MI}_{JS}(\boldsymbol{h}_i,\boldsymbol{h}_j)=\mathbb{E}_{p(\boldsymbol{h}_i,\boldsymbol{h}_j)}\left[\log\left(D(\boldsymbol{h}_i,\boldsymbol{h}_j)\right)\right]-\log\mathbb{E}_{p(\boldsymbol{h}_i)p(\boldsymbol{h}_j)}\left[\log\left(1-D(\boldsymbol{h}_i,\boldsymbol{h}_j)\right)\right]$$

JS 下界是将互信息公式中的 KL 散度替换成 JS 散度得到的。如果令$D(\boldsymbol{h}_i,\boldsymbol{h}_j)=$

sigmoid $\left(D^{'}\left(\boldsymbol{h}_i,\boldsymbol{h}_j\right)\right)$，那么还可以被重写为 softplus (SP) 版本，其中 $\mathrm{sp}(\boldsymbol{x})=\log(1+\mathrm{e}^{\boldsymbol{x}})$：

$$\mathrm{MI}_{SP}\left(\boldsymbol{h}_i,\boldsymbol{h}_j\right)=\mathbb{E}_{p(\boldsymbol{h}_i,\boldsymbol{h}_j)}\left[-\mathrm{sp}\left(-D^{'}\left(\boldsymbol{h}_i,\boldsymbol{h}_j\right)\right)\right]-\log\mathbb{E}_{p(\boldsymbol{h}_i)p(\boldsymbol{h}_j)}\left[\mathrm{sp}\left(D^{'}\left(\boldsymbol{h}_i,\boldsymbol{h}_j\right)\right)\right]$$

(3) InfoNCE Estimator[28]：

$$\mathrm{MI}_{NCE}\left(\boldsymbol{h}_i,\boldsymbol{h}_j\right)=\mathbb{E}_{p(\boldsymbol{h}_i,\boldsymbol{h}_j)}\left[D\left(\boldsymbol{h}_i,\boldsymbol{h}_j\right)-\mathbb{E}_{K\sim\mathcal{P}^N}\left[\log\frac{1}{N}\sum_{\boldsymbol{h}_j^{'}\in K}\exp\left(D\left(\boldsymbol{h}_i,\boldsymbol{h}_j^{'}\right)\right)\right]\right]$$

InfoNCE 是最常用的互信息下界估计之一，提供了较为宽松的下界。并且在大规模数据上，InfoNCE 训练效率高且稳定。其中，K包含从相同且独立分布中采样的N个随机变量。在实际中，InfoNCE 通常在大小为N的 mini-batch 上计算，那么上式可以被重写为（忽略常数 $\log N$）：

$$\mathrm{MI}_{NCE}=-\frac{1}{N}\sum_{(\boldsymbol{A},\boldsymbol{X})\in B}\log\frac{\exp\left(D\left(\boldsymbol{h}_i,\boldsymbol{h}_j\right)\right)}{\sum_{(\boldsymbol{A}^{'},\boldsymbol{X}^{'})\in B}\exp\left(D\left(\boldsymbol{h}_i,\boldsymbol{h}_j^{'}\right)\right)}$$

需要注意的是，对比学习并不是必须最大化互信息。还有其他很多方法也可以用于对比学习当中，例如 Triplet Margin Loss[60]直观地最大化正样本的相似性，保证正样本和负样本对之间的差异程度：（$L\left(\boldsymbol{h}_i,\boldsymbol{h}_j,\boldsymbol{h}^{'}_j\right)=\max\left\{D\left(\boldsymbol{h}_i,\boldsymbol{h}^{'}_j\right)-D\left(\boldsymbol{h}_i,\boldsymbol{h}_j\right)+\epsilon,0\right\}$）。

2. 生成式自监督

生成式自监督旨在重建输入数据并将输入数据用作监督信号。这类方法起源于自动编码器，它通过编码器将数据向量压缩成低维表示，然后尝试通过解码器重建输入向量。与以向量形式表示的输入数据不同，图数据是相互连接的。所以，对于图来讲输入数据通常是整张图。具体来说，对于图的生成式方法考虑将图数据重构任务作为代理任务，它从两个角度出发：特征和结构，也就是关注节点特征、边特征和图的邻接矩阵重构。其目标可以如下表述：

$$\min_{\theta,\phi}L_{ssl}\left(p_\phi\left(f_\theta\left(\widetilde{G}\right)\right),G\right)$$

其中，$f_\theta(\cdot)$是图编码器（Graph Encoder），$p_\phi(\cdot)$是代理任务解码器（Pretext Task Decoder）。$\widetilde{G}$表示相对原始图带有扰动的节点特征、边特征或邻接矩阵的图数据。对于大多数生成式方法，通常定义自监督的目标函数L_{ssl}来度量重构图与原始图数据之间的差异。

根据重建的对象，可以将生成式方法分为两个类别：(1) 学习重建图的特征信息的特征生成；(2) 学习重建图的拓扑结构信息的结构生成。由于不同的学习目标，它们在解码器和损失函数的设计上有不同，特征生成方法通常捕获节点级别知识；结构生成学习得到的表示通常包含更多的节点对级别信息，因为结构生成注重边的重建。下面将从这两个方面介绍。

1) 特征生成

特征生成方法通过从扰动的或原始图中重建特征信息来学习，可以形式化为：

$$\min_{\theta,\phi}L_{mse}\left(p_\phi\left(f_\theta\left(\widetilde{G}\right)\right),\boldsymbol{X}\right)$$

其中，$p_\phi(\cdot)$是用于特征回归的解码器，L_{mse}是均方误差（MSE）损失函数，X是各种特征矩阵的通用表达式，例如节点特征矩阵、边特征矩阵或低维特征矩阵。一些比较经典的方法有掩码特征回归（Masked Feature Regression）、重建含噪声的特征等，下面通过几个例子说明。

受到 CV 中图像修复启发，特征生成中一个较为典型的方法是掩码特征回归，对掩盖的特征进行回归预测重建。例如 Graph Completion[61]提出将某些节点的特征掩码，通过未掩盖的特征重建被掩盖的特征（如图 5-17）。具体来讲，对于一个节点v_i，随机地对它的特征$\boldsymbol{x}_i$进行掩盖，即$\widehat{\boldsymbol{x}}_i=\boldsymbol{x}_i\odot\boldsymbol{m}_i$，其中$\boldsymbol{m}_i\in\{0,1\}$，就可以得到新的特征矩阵$\widehat{\boldsymbol{X}}$和图$\widetilde{G}=\left(\boldsymbol{A},\widehat{\boldsymbol{X}}\right)$，对应原特征生成方法形式化描述中的$\widetilde{G}$。学习目标是通过 GCN 编码器使用未被掩盖的邻居节点的特征预测被掩盖的节点特征。另一类方法考虑通过含噪声的特征生成无噪的特征。例如 Node Attribute Denoising[31]对原始图中的节点特征添加随机噪声，以获得含噪声的节点特征矩阵$\widehat{\boldsymbol{X}}=$

$\boldsymbol{X}+N(0,\boldsymbol{\Sigma})$和$\widetilde{\boldsymbol{G}}=\left(\boldsymbol{A},\widehat{\boldsymbol{X}}\right)$，从而训练模型重建无噪声的特征。类似地，还可以对节点嵌入添加噪声进行学习，即$\widehat{\boldsymbol{H}}=\boldsymbol{H}+N(0,\boldsymbol{\Sigma})$。

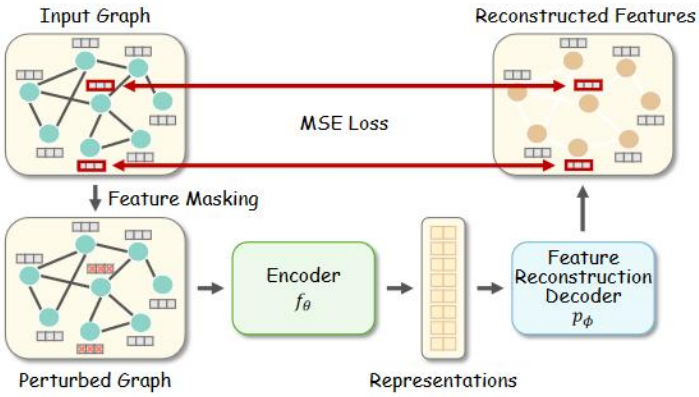


图 5-17 Graph Completion 示意图[61]

2) 结构生成

与重建特征信息的方法不同，结构生成方法通过重建结构信息进行学习。在大多数情况下，目标是重建邻接矩阵，可以形式化表示如下：

$$\min_{\theta,\phi} L_{mse}\left(p_\phi\left(f_\theta(\widetilde{G})\right),\boldsymbol{A}\right)$$

其中，$p_\phi(\cdot)$是用于特征回归的解码器，$\boldsymbol{A}$可以是全部或部分的邻接矩阵。例如，GAE[63]直接通过编码器得到的节点嵌入生成邻接矩阵。具体来说，GAE 中直接采用 GCN 作为编码器，即$\boldsymbol{Z}=\mathrm{GCN}(\boldsymbol{X},\boldsymbol{A})\in\mathbb{R}^{N\times f}$；同时采用内积作为解码器来重建邻接矩阵，即$\widehat{\boldsymbol{A}}=\sigma(\boldsymbol{Z}\boldsymbol{Z}^{\mathrm{T}})$，其中$\sigma(\cdot)$可使用 sigmoid 函数。损失函数采用 BCE loss 来最大化重建邻接矩阵与原始邻接矩阵之间的相似性。

基于 GAE 出现了很多扩展工作，从不同角度优化特征的表示。例如 VGAE[63]将变分自编码器（Variational Autoencoder）的概念引入到 GAE 中（图 5-18），考虑学习数据的潜在分布，能够更有效地捕捉输入数据的深层特征。具体来说，首先需要确定一个高斯分布，通过 GCN 来确定$\boldsymbol{\mu}$和$\boldsymbol{\sigma}$，即$\boldsymbol{\mu}=\mathrm{GCN}_\mu(\boldsymbol{X},\boldsymbol{A})$，$\log\boldsymbol{\sigma}=\mathrm{GCN}_\sigma(\boldsymbol{X},\boldsymbol{A})$，$\mathrm{GCN}_\mu$和$\mathrm{GCN}_\sigma$使用两层的 GCN 并共享第一层 GCN。然后，通过采样高斯分布得到$\boldsymbol{Z}$。但直接从$N(\mu,\epsilon)$采样无法得到梯度信息，所以采用重参数化（Reparameterization），即通过从$N(0,1)$采样ϵ得到$\boldsymbol{Z}=\boldsymbol{\mu}+\epsilon\boldsymbol{\sigma}$。解码器同样使用内积来重建邻接矩阵，即$\widehat{\boldsymbol{A}}=\sigma(\boldsymbol{Z}\boldsymbol{Z}^{\mathrm{T}})$。为了使 GCN 计算出来的分布$q(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{A})$与高斯分布$p(\boldsymbol{Z})$接近，损失函数由 BCE loss 和 KL 散度组成，即$L=L_{BCE}+KL[q(\boldsymbol{Z}|\boldsymbol{X},\boldsymbol{A})||p(\boldsymbol{Z})]$。

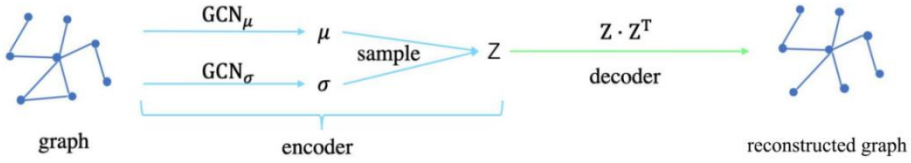


图 5-18 VGAE 示意图[63]

5.3.2 图课程学习

为了让学生更好地掌握知识，课程通常采取由简到难的方式。图课程学习正是模仿人类的学习方式，通过让图模型逐渐从简单到复杂的任务中进行学习，可以有助于图模型的收敛和提高泛化能力。大多数图课程学习方法具有两个重要部分：难度测量器（Difficulty Measurer）和训练调度器（Training Scheduler）。前者评估训练数据的难度，为学习提供先验知识；后者决定如何从简单到复杂的样本中学习。

具体来说，课程学习可以定义如下：设$C=\langle Q_1,\ldots,Q_t,\ldots,Q_T\rangle$表示训练图模型的课程，其

中包括了T个训练步骤中使用的一系列训练数据子集。每个数据子集Q_t包含了当前训练样本，在时间步t送到模型中学习。所有数据子集的顺序由难度衡量器和训练调度器确定。那么，给定训练集（即节点、边或图）$D=\{(X_i,Y_i)\}_{i=1}^{|D|}$，其中$X_i$是输入实例，$Y_i$表示标签，目标是通过课程$C$划分学习顺序以尽可能优化图学习模型，以在测试集上达到最佳性能。本小节将介绍不同级别图任务所使用的图课程学习方法，大致将其分为三类，即节点级、边级、图级。同时，根据训练策略又可以分为两类，一类是预定义的课程学习，在训练过程之前定义启发式指标来衡量训练的难度，手动设计训练顺序，通常在预处理阶段进行；一类是自动课程学习，在训练过程中考虑模型的反馈，以动态调整优化状态，依赖于可计算的指标来自动设计模型训练策略。

1. 节点级课程学习

一般来说，解决节点级任务是通过训练图模型来学习节点表示。因此，对节点级图课程学习的很多研究，是通过先输入简单的节点，并逐渐包含更难的节点来训练图模型。

1) 预定义的节点级课程学习

对于节点级来讲，预定义课程学习中的启发式指标可以是节点的拓扑结构、特征或标签等属性来衡量节点训练的难度，下面以 CLNode[64]为例介绍。CLNode（Curriculum Learning for Node Classification）提出了多角度的难度测量器，用于根据标签信息衡量训练节点的难度，可以从局部和全局两个角度来评估节点的难度。对于局部难度测量器，它衡量局部标签分布情况来识别哪些节点的邻居具有不同标签，这些节点可以看作跨类别节点。一般认为这些节点是难节点，因为聚合的邻居信息混杂了多个类别的特征，导致它们难以学习。具体过程如下：

$$P_c(u)=\frac{\left|\{\widetilde{Y}[v]=c|v\in\widehat{N}(u)\}\right|}{\left|\widehat{N}(u)\right|},D_{\text{local}}(u)=-\sum_{c\in C}P_c(u)\log\left(P_c(u)\right)$$

其中，$\widetilde{Y}[v]$表示节点 v 的伪标签，$\widehat{N}(u)$表示 $N(u)\cup\{u\}$，而$P_c(u)$表示邻域$\widehat{N}(u)$中属于类别c的节点的比例。局部信息熵D_{local}越大，表示节点的邻居越复杂（即属于多个类别），更可能是跨类别的节点。因此，在邻居聚合过程中，这些节点会聚合来自不同类别的邻居的特征，导致模糊的表示，使得 GNN 很难学习这些节点。但是伪标签可能是不正确的，所以需要引入全局难度测量器。它通过分析节点的特征相似性来识别标签错误的节点，因为误标的节点通常与其类别的典型节点特征不相似，相似度越小说明节点训练难度越大。具体过程如下：

$$\mathcal{V}_c=\{v|\widetilde{Y}[v]=c\},\boldsymbol{h_c}=\text{Avg}(\boldsymbol{h_v}|v\in\mathcal{V}_c),S(u)=\frac{\exp\left(\boldsymbol{h_u}\cdot\boldsymbol{h}_{c_u}\right)}{\max\limits_{c\in C}\exp(\boldsymbol{h_u}\cdot\boldsymbol{h_c})},D_{\text{global}}(u)=1-S(u)$$

其中，$\mathcal{V}_c$表示属于类别c的节点，$\boldsymbol{h_c}$是类别c的代表性特征。而c_u表示节点u的标签类别，$S(u)$计算$\boldsymbol{h_u}$和$\boldsymbol{h}_{c_u}$之间的特征相似性，所以误标的节点往往比正确标记的节点具有更小的$S(u)$。D_{global}从全局角度衡量节点的难度，以此来识别误标的训练节点。从而，CLNode 选择性地将这些节点排除在训练过程之外，提高了 GNN 对标签噪声的鲁棒性。结合来自局部和全局的两个难度测量器，最终将节点u的难度定义如下，其中α为超参数：

$$D(u)=D_{\text{local}}(u)+\alpha\cdot D_{\text{global}}(u)$$

对于训练调度器，采用的是连续的训练调度器。具体来讲，首先根据节点难度对训练集进行升序排序；然后，使用节奏函数（Pacing Function）$g(t)$将训练轮数t映射到标量λ_t，其范围是$(0,1]$，这意味着在第t轮训练中，将最容易的训练节点按比例λ_t作为训练子集使用。设λ_0为初始最容易节点的比例，T为$g(t)$首次达到 1 的训练轮次。其中节奏函数可以分为三种，分别是线性、根号型和几何型（例如，线性的节奏函数为$\lambda_t=\min\left(1,\lambda_0+(1-\lambda_0)\cdot\frac{t}{T}\right)$）。总的来说，CLNode 可以兼容大多数现有的 GNN，并按照从易到难的顺序提供训练节点，提升它们的性能，而不增加时间复杂性。

2) 自动的节点级课程学习

自动图课程学习也分为自步学习（Self-paced Learning）、迁移教师（Transfer Teacher）、强化学习教师（RL Teacher）等，其中自步学习让学生自己充当教师，根据训练损失来衡量训

练样本的难度；迁移教师方法则通过一个经过预训练的模型来充当教师，根据教师在训练样本的表现来衡量训练实例的难度；强化学习教师方法采用强化学习，教师根据学生的反馈动态衡量训练实例的难度，两者共同进步。下面自动图课程学习的部分都将主要介绍自步学习方法。

传统的自步学习目标函数可以表示如下：

$$\min_{\boldsymbol{w},q_i} E(\boldsymbol{w},q_i;\lambda)=\sum_{i=1}^{N} q_i\,L\big(y_i,f(\boldsymbol{x_i},\boldsymbol{w})\big)-\lambda\sum_{i=1}^{N} q_i\,,\quad q_i^*=\begin{cases}1, & \text{if } L\big(y_i,f(\boldsymbol{x}_i,\boldsymbol{w})\big)<\lambda\\ 0, & \text{otherwise.}\end{cases}$$

其中，y_i和$f(\boldsymbol{x}_i,\boldsymbol{w})$分别表示节点$i$的真实标签和预测标签。$L\big(y_i,f(\boldsymbol{x_i},\boldsymbol{w})\big)$表示基于节点$i$的训练损失。$\lambda\sum_{i=1}^{N} q_i$表示自步正则项，其中$q_i\in[0,1]$是控制节点$i$对训练损失贡献的权重，$N$是训练实例的数量，$\lambda$是手动指定的超参数，用于平衡正则项在整体损失中的贡献。模型根据可学习的模型参数$\boldsymbol{w}$和样本权重q_i进行训练，对于固定的$\boldsymbol{w}$，最优的q_i^*只将损失小于λ的节点纳入训练，随着λ的逐渐增加，更复杂的数据样本会被加入训练。

DSP-GCN[65]也采用了此方法，逐步将预测置信度较高的未标记节点纳入训练集。其采用的损失函数由两部分构成，即有标签节点的损失和无标签节点的损失：

$$L_{\text{node}}=\sum_{i=1}^{l} l\left(y_i,g(\boldsymbol{x}_i,\boldsymbol{A},\boldsymbol{W})\right)+\sum_{k=l+1}^{N} q_k\,l\big(y_k,g(\boldsymbol{x}_k,\boldsymbol{A},\boldsymbol{W})\big)-\lambda_{\text{node}}\sum_{k=l+1}^{N} q_k$$

其中，y_i和y_k分别表示已标记节点的给定标签和未标记节点的预测标签。$q_k\in\{0,1\}$是分配给未标记节点v_k的权重，用于指示该节点v_k及其预测标签y_k是否已被加入训练集。同样地，如果λ_{node}增加，更多的未标记节点将被加入到训练集中，即更多未标记节点满足$q_k^*=1$的条件，反之亦然。预测模型$g(\boldsymbol{x}_k,\boldsymbol{A},\boldsymbol{W})$可以是 GCN，损失函数$l(\cdot,\cdot)$为交叉熵。在固定$\boldsymbol{W}$的情况下，$q_k$取值与前面$q_i^*$类似的最优解。需要注意的是，DSP-GCN 最终的损失函数也纳入了对边级自步学习，在这里不过多介绍。

2. 边级课程学习

与节点级图课程学习相比，边级图课程学习衡量边的难度并调度边的训练。目前这方面的研究主要集中在自动图课程学习，因此接下来将主要介绍自动的边级课程学习。

与节点级别的自步学习不同，边级别的自步学习在训练过程中逐渐引入边。其中最基本的自步学习已经在节点级课程学习中介绍，下面以 SCCABG[66]为例介绍边级别的自步学习。它主要采用自步学习一致性聚类（Consensus Clustering）方法，逐步加入可靠性较低的数据，是一种通过整合多个弱聚类结果来生成一个更加稳健的聚类结果的框架。

具体来说，首先需要构建二分图，两侧分别由所有节点和其弱聚类后的簇节点构成，后者可能包含了多个弱聚类的结果，其邻接矩阵为$\boldsymbol{G}=\begin{bmatrix}0 & \boldsymbol{Y}\\ \boldsymbol{Y}^{\mathrm{T}} & 0\end{bmatrix}$。而目标是学习一个新的结构特征明显的邻接矩阵，作为一致性聚类。具体来讲，需要得到新邻接矩阵$\boldsymbol{G}'=\begin{bmatrix}0 & \boldsymbol{S}\\ \boldsymbol{S}^{\mathrm{T}} & 0\end{bmatrix}$，其中包含$c$个连通分量，即同一个连通分量的节点属于同一聚类，最终得到 c 个聚类。为了尽可能保留原邻接矩阵的特征，那么需要遵循以下目标：

$$\min_{\boldsymbol{S}}\|\boldsymbol{S}-\boldsymbol{Y}\|_F^2,\quad \text{s.t.}\quad 0\le S_{ij}\le 1,\text{rank}(\boldsymbol{L})=n+k-c$$

其中，$\boldsymbol{L}=\boldsymbol{I}-\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{G}'\boldsymbol{D}^{-\frac{1}{2}}$为拉普拉斯矩阵，$n$为节点数，$k$为弱聚类的簇数，由$c$个连通分量可得到$c=n+k-\text{rank}(\boldsymbol{L})$。最后，在此基础上引入自步学习，于是引入权重矩阵$\boldsymbol{W}\in[0,1]^{n\times k}$，其中$W_{ij}$的值越大，表示相应的边越可靠，通过$\boldsymbol{W}$不断引入更困难的边进行学习:

$$\min_{S,W}\|\boldsymbol{W}\odot(\boldsymbol{S}-\boldsymbol{Y})\|_F^2-\lambda\|\boldsymbol{W}\|_1$$

实际的损失函数中，还会考虑引入其他约束以更有效地训练参数。比如考虑到不同簇间的相似性，对于相似性越大的两个簇意味着一个节点要么同时属于它们，要么同时不属于它们。

3. 图级课程学习

不同于上述节点级和边级的课程学习，图级课程学习将图信息纳入到难度测量器和训练调度器中。目前这方面的研究主要集中在预定义的图课程学习，因此下面主要介绍预定义的图级

课程学习方法。

预定义的图级课程学习以 CuCo[67]为例介绍。CuCo 使用课程学习逐步选取困难的负样本进行对比学习，并基于负样本和正样本之间的嵌入相似性来确定难度。具体来说，对于正样本，对原图g_i通过数据增强得到g_j，再通过图编码器得到图级表示$\{\boldsymbol{z}_i, \boldsymbol{z}_j\}$，将其视为对比学习中的正样本对；对于负样本，创建了一个内存库，其中包含了每个正样本对$\{\boldsymbol{z}_i, \boldsymbol{z}_j\}$对应的$K$个负样本图，可以通过图编码器得到对应的图级表示$\{\boldsymbol{z}_k\}_{k=1}^{K}$。同样地，CuCo 的过程可以分为难度测量器和训练调度器：对于难度测量器，会对每一个负样本图嵌入进行评分。具体来讲，会将一个负样本嵌入向量$\boldsymbol{z}_k$映射为一个数值得分$S(\boldsymbol{z}_k)$，用于衡量其难度。可以将评分函数$S(\boldsymbol{z}_k)$设置为 $\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k) \in \mathbb{R}$，即基于负样本和正样本图级表示之间的相似性来确定难度，因为越相似意味着区分它们的难度更大。可以考虑下面两种相似度的评分函数（余弦相似度、点积相似度）：

$$S(\boldsymbol{z}_k) = \frac{|\boldsymbol{z}_k \cdot \boldsymbol{z}_i|}{|\boldsymbol{z}_k||\boldsymbol{z}_i|} \quad 或 \quad S(\boldsymbol{z}_k) = \boldsymbol{z}_k \cdot \boldsymbol{z}_i$$

对于训练调度器，同样引入了负样本的节奏函数，会在训练第t轮的内存库包含得分最低的$g(t)$个样本。将完整的内存库大小记为K，将训练的总步数记为T。这里考虑了四个常见的函数：对数函数、线性函数、二次函数和根函数:

$$g(t) = \left[1 + 0.1\log\left(\frac{t}{T} + \mathrm{e}^{-10}\right)\right] \cdot K \quad 或 \quad g(t) = \left(\frac{t}{T}\right)^{\lambda} \cdot K$$

其中，第一个式子为对数节奏函数，第二个式子中λ是控制训练过程中进度的平滑参数，$\lambda = 1/2$，1，2 分别表示根、线性和二次节奏函数。

5.4 本章小结

本章主要介绍了图神经网络的进阶知识，从数据、架构、训练三个角度审视了图神经网络自身的不足以及实际应用中的困难，并从这三个角度详细介绍了对图神经网络的各种优化算法，使读者能更加深刻地理解图神经网络的内在原理，以及熟练利用图神经网络来解决实际问题。

第一节介绍了数据优化相关技术，围绕图结构、图特征和图标签的三类图数据。探讨了图结构优化如何通过结构缩减降低计算复杂度，采用结构增强提升模型泛化能力，并利用特征生成得到多样化样本；阐释了图特征优化如何通过特征增强、选择和补全，丰富节点特征，提升模型学习能力，避免过拟合；讨论了图标签优化如何通过标签混合、伪标签和主动学习增加标签多样性，减少噪声影响。

第二节介绍了架构优化相关技术，从消息传递、采样、池化三个角度出发，重点讨论了如何改进传统图神经网络的信息传递机制，解决过平滑、长距离依赖和表达能力不足的问题；并阐述了采样和池化技术的相关改进方法，使得在处理大规模图数据时更具效率和可扩展性。

第三节介绍了训练优化相关技术。重点介绍了通过图自监督学习从未标记数据中学习有用的特征表示，降低对大规模标注数据的依赖；探讨了通过设计合理的课程学习逐步增加任务的难度，从而使模型训练过程更加稳定，提升模型的收敛速度和训练效果。

扩展阅读材料

(1) Graph Neural Networks: Foundations, Frontiers, and Applications[M]. Springer Nature, 2022.

(2) Liu Y, Jin M, Pan S, et al. Graph self-supervised learning: A survey[J]. IEEE transactions on knowledge and data engineering, 2022, 35(6): 5879-5900.

(3) Grattarola D, Zambon D, Bianchi F M, et al. Understanding pooling in graph neural networks[J]. IEEE transactions on neural networks and learning systems, 2022, 35(2): 2708-2718.

(4) Li H, Wang X, Zhu W. Curriculum graph machine learning: a survey[C]//Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence. 2023: 667-6682.

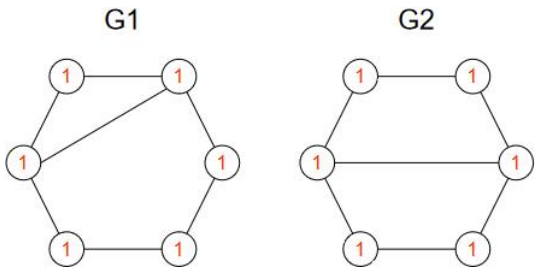
(5) Shi C, Wang X, Yang C. Advances in Graph Neural Networks[M]. Springer, 2023.

(6) Wu Z, Pan S, Chen F, et al. A comprehensive survey on graph neural networks[J]. IEEE

transactions on neural networks and learning systems, 2020, 32(1): 4-24.

习题

(1) 分别阐述结构优化、特征优化和标签优化的适用场景，并举例说明。
(2) 简述图神经网络过平滑现象的原因，并举例说明 2 种常见的缓解过平滑问题的技术。
(3) 思考：图神经网络的过压缩和过平滑现象有什么区别和联系？
(4) 用 WL-test 算法证明下面两个图非同构。



两个非同构图

(5) 证明消息传递图神经网络区分能力的上限是 1-WL 测试。
(6) 举例并说明常见的图对比学习数据增强手段。
(7) 对比式和生成式自监督各自的优劣势。

参考文献

[1]Loukas A, Vandergheynst P. Spectrally approximating large graphs with smaller graphs[C]//International conference on machine learning. PMLR, 2018: 3237-3246.
[2]Jin W, Zhao L, Zhang S, et al. Graph condensation for graph neural networks[J]. arXiv preprint arXiv:2110.07580, 2021.
[3]Rong Y, Huang W, Xu T, et al. Dropedge: Towards deep graph convolutional networks on node classification[J]. arXiv preprint arXiv:1907.10903, 2019.
[4]Sun M, Xing J, Wang H, et al. MoCL: data-driven molecular fingerprint via knowledge-aware contrastive learning from molecular graph[C]//Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining. 2021: 3585-3594.
[5]Jin W, Ma Y, Liu X, et al. Graph structure learning for robust graph neural networks[C]//Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. 2020: 66-74.
[6]Liu G, Zhao T, Xu J, et al. Graph rationalization with environment-based augmentations[C]//Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2022: 1069-1078.
[7]You Y, Chen T, Shen Y, et al. Graph contrastive learning automated[C]//International Conference on Machine Learning. PMLR, 2021: 12121-12132.
[8]You J, Ying R, Ren X, et al. Graphrnn: Generating realistic graphs with deep auto-regressive models[C]//International conference on machine learning. PMLR, 2018: 5708-5717.
[9]Chen X, He J, Han X, et al. Efficient and degree-guided graph generation via discrete diffusion modeling[J]. arXiv preprint arXiv:2305.04111, 2023.
[10]Kipf T N, Welling M. Variational graph auto-encoders[J]. arXiv preprint arXiv:1611.07308, 2016.
[11]Zhang Y, Pal S, Coates M, et al. Bayesian graph convolutional neural networks for semi-supervised classification[C]//Proceedings of the AAAI conference on artificial intelligence. 2019, 33(01): 5829-5836.
[12]You J, Ying R, Leskovec J. Position-aware graph neural networks[C]//International conference on machine learning. PMLR, 2019: 7134-7143.
[13]Jiang B, Wang B, Luo B. Sparse norm regularized attribute selection for graph neural networks[J].

Pattern Recognition, 2023, 137: 109265.
[14]Maurya S K, Liu X, Murata T. Not all neighbors are friendly: Learning to choose hop features to improve node classification[C]//Proceedings of the 31st ACM International Conference on Information & Knowledge Management. 2022: 4334-4338.
[15]Chen X, Chen S, Yao J, et al. Learning on attribute-missing graphs[J]. IEEE transactions on pattern analysis and machine intelligence, 2020, 44(2): 740-757.
[16]Jin D, Huo C, Liang C, et al. Heterogeneous graph neural network via attribute completion[C]//Proceedings of the web conference 2021. 2021: 391-400.
[17]Wang Y, Wang W, Liang Y, et al. Mixup for node and graph classification[C]//Proceedings of the Web Conference 2021. 2021: 3663-3674.
[18]Park J, Shim H, Yang E. Graph transplant: Node saliency-guided graph mixup with local structure preservation[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2022, 36(7): 7966-7974.
[19]Li Q, Han Z, Wu X M. Deeper insights into graph convolutional networks for semi-supervised learning[C]//Proceedings of the AAAI conference on artificial intelligence. 2018, 32(1).
[20]Wu Y, Xu Y, Singh A, et al. Active learning for graph neural networks via node feature propagation[J]. arXiv preprint arXiv:1910.07567, 2019.
[21]Zhao L, Akoglu L. Pairnorm: Tackling oversmoothing in gnns[J]. arXiv preprint arXiv:1909.12223, 2019.
[22]Rusch T K, Chamberlain B, Rowbottom J, et al. Graph-coupled oscillator networks[C]//International Conference on Machine Learning. PMLR, 2022: 18888-18909.
[23]Rusch T K, Chamberlain B P, Mahoney M W, et al. Gradient gating for deep multi-rate learning on graphs[J]. arXiv preprint arXiv:2210.00513, 2022.
[24]Li G, Muller M, Thabet A, et al. Deepgcns: Can gcns go as deep as cnns?[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2019: 9267-9276.
[25]Chen M, Wei Z, Huang Z, et al. Simple and deep graph convolutional networks[C]//International conference on machine learning. PMLR, 2020: 1725-1735.
[26]Maskey S, Paolino R, Bacho A, et al. A fractional graph laplacian approach to oversmoothing[J]. Advances in Neural Information Processing Systems, 2024, 36.
[27]Ying C, Cai T, Luo S, et al. Do transformers really perform badly for graph representation?[J]. Advances in neural information processing systems, 2021, 34: 28877-28888.
[28]Morris C, Ritzert M, Fey M, et al. Weisfeiler and leman go neural: Higher-order graph neural networks[C]//Proceedings of the AAAI conference on artificial intelligence. 2019, 33(01): 4602-4609.
[29]Qian C, Rattan G, Geerts F, et al. Ordered subgraph aggregation networks[J]. Advances in Neural Information Processing Systems, 2022, 35: 21030-21045.
[30]Murphy R, Srinivasan B, Rao V, et al. Relational pooling for graph representations[C]//International Conference on Machine Learning. PMLR, 2019: 4663-4673.
[31]Aldous D J. Representations for partially exchangeable arrays of random variables[J]. Journal of Multivariate Analysis, 1981, 11(4): 581-598.
[32]Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs[J]. Advances in neural information processing systems, 2017, 30.
[33]Ying R, He R, Chen K, et al. Graph convolutional neural networks for web-scale recommender systems[C]//Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining. 2018: 974-983.
[34]Chen J, Ma T, Xiao C. Fastgcn: fast learning with graph convolutional networks via importance sampling[J]. arXiv preprint arXiv:1801.10247, 2018.
[35]Chiang W L, Liu X, Si S, et al. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks[C]//Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. 2019: 257-266.
[36]Zeng H, Zhou H, Srivastava A, et al. Graphsaint: Graph sampling based inductive learning method[J]. arXiv preprint arXiv:1907.04931, 2019.
[37]Chen J, Zhu J, Song L. Stochastic training of graph convolutional networks with variance

reduction[J]. arXiv preprint arXiv:1710.10568, 2017.
[38]Huang W, Zhang T, Rong Y, et al. Adaptive sampling towards fast graph representation learning[J]. Advances in neural information processing systems, 2018, 31.
[39]Zaheer M, Kottur S, Ravanbakhsh S, et al. Deep sets[J]. Advances in neural information processing systems, 2017, 30.
[40]Chen T, Bian S, Sun Y. Are powerful graph neural nets necessary? a dissection on graph classification[J]. arXiv preprint arXiv:1905.04579, 2019.
[41]Liu C, Zhan Y, Wu J, et al. Graph pooling for graph neural networks: Progress, challenges, and opportunities[J]. arXiv preprint arXiv:2204.07321, 2022.
[42]Fan X, Gong M, Xie Y, et al. Structured self-attention architecture for graph-level representation learning[J]. Pattern Recognition, 2020, 100: 107084.
[43]Itoh T D, Kubo T, Ikeda K. Multi-level attention pooling for graph neural networks: Unifying graph representations with multiple localities[J]. Neural Networks, 2022, 145: 356-373.
[44]Ying Z, You J, Morris C, et al. Hierarchical graph representation learning with differentiable pooling[J]. Advances in neural information processing systems, 2018, 31.
[45]Gao H, Ji S. Graph u-nets[C]//international conference on machine learning. PMLR, 2019: 2083-2092.
[46]Zhang L, Wang X, Li H, et al. Structure-feature based graph self-adaptive pooling[C]//Proceedings of The Web Conference 2020. 2020: 3098-3104.
[47]Wu L, Lin H, Tan C, et al. Self-supervised learning on graphs: Contrastive, generative, or predictive[J]. IEEE Transactions on Knowledge and Data Engineering, 2021, 35(4): 4216-4235.
[48]Zhu Y, Xu Y, Yu F, et al. Graph contrastive learning with adaptive augmentation[C]//Proceedings of the web conference 2021. 2021: 2069-2080
[49]Jovanović N, Meng Z, Faber L, et al. Towards robust graph contrastive learning[J]. arXiv preprint arXiv:2102.13085, 2021.
[50]You Y, Chen T, Sui Y, et al. Graph contrastive learning with augmentations[J]. Advances in neural information processing systems, 2020, 33: 5812-5823.
[51]Ren Y, Bai J, Zhang J. Label contrastive coding based graph neural network for graph classification[C]//Database Systems for Advanced Applications: 26th International Conference, DASFAA 2021, Taipei, Taiwan, April 11 – 14, 2021, Proceedings, Part I 26. Springer International Publishing, 2021: 123-140.
[52]Qiu J, Chen Q, Dong Y, et al. Gcc: Graph contrastive coding for graph neural network pre-training[C]//Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. 2020: 1150-1160.
[53]Yanqiao Z, Yichen X, Feng Y, et al. Deep graph contrastive representation learning[J]. arXiv preprint arXiv:2006.04131, 2020.
[54]Velickovic P, Fedus W, Hamilton W L, et al. Deep graph infomax[J]. ICLR (Poster), 2019, 2(3): 4.
[55]Jiao Y, Xiong Y, Zhang J, et al. Sub-graph contrast for scalable self-supervised graph representation learning[C]//2020 IEEE international conference on data mining (ICDM). IEEE, 2020: 222-231.
[56]Sun F Y, Hoffmann J, Verma V, et al. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization[J]. arXiv preprint arXiv:1908.01000, 2019.
[57]Belghazi M I, Baratin A, Rajeshwar S, et al. Mutual information neural estimation[C]//International conference on machine learning. PMLR, 2018: 531-540.
[58]Nowozin S, Cseke B, Tomioka R. f-gan: Training generative neural samplers using variational divergence minimization[J]. Advances in neural information processing systems, 2016, 29.
[59]Gutmann M, Hyvärinen A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models[C]//Proceedings of the thirteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings, 2010: 297-304.
[60]Schroff F, Kalenichenko D, Philbin J. Facenet: A unified embedding for face recognition and clustering[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 815-823.

[61]You Y, Chen T, Wang Z, et al. When does self-supervision help graph convolutional networks?[C]//international conference on machine learning. PMLR, 2020: 10871-10880.
[62]Manessi F, Rozza A. Graph-based neural network models with multiple self-supervised auxiliary tasks[J]. Pattern Recognition Letters, 2021, 148: 15-21.
[63]Kipf T N, Welling M. Variational graph auto-encoders[J]. arXiv preprint arXiv:1611.07308, 2016
[64]Wei X, Gong X, Zhan Y, et al. Clnode: Curriculum learning for node classification[C]//Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining. 2023: 670-678.
[65]Yang L, Chen Z, Gu J, et al. Dual Self-Paced Graph Convolutional Network: Towards Reducing Attribute Distortions Induced by Topology[C]//IJCAI. 2019: 4062-4069.
[66]Zhou P, Liu X, Du L, et al. Self-paced adaptive bipartite graph learning for consensus clustering[J]. ACM Transactions on Knowledge Discovery from Data, 2023, 17(5): 1-35.
[67]Chu G, Wang X, Shi C, et al. CuCo: Graph Representation with Curriculum Contrastive Learning[C]//IJCAI. 2021: 2300-2306.