【摘　要】

图神经网络对学习图结构数据具有优异的性能，在社交网络、推荐系统、生物医学发现、金融风控等不同领域中取得了广泛的应用。其中在推荐系统、金融风控等领域的应用中，对实时性有较高的要求。在非欧式空间中，图神经网络对数据特征的获取依赖于多次跳跃的邻居节点信息。这导致图神经网络的推导速度无法从根本上得到提高，实际部署困难。图神经网络现有的推理加速方法，如剪枝和量化，虽然可以通过减少乘法和累积操作，在一定程度上加快推理速度。但是，由于图神经网络对图的依赖性没有得到解决，它们的改进是有限的。现有的知识蒸馏方法，试图通过来自教师模型图神经网络的软标签，训练学生模型多层感知机来解决模型在推理时对图的依赖。然而，训练后的多层感知机有效性和鲁棒性都受到了限制。受限的原因主要是：1、模型没有捕捉到节点的位置信息；2、对教师输出的严格硬匹配，使模型无法实际学习到图神经网络对图结构信息的捕捉知识，在噪声的影响下，模型的性能显著下降，经过知识蒸馏后的学生模型鲁棒性较差。为了提高知识蒸馏后的多层感知机的有效性，本文提出将节点的位置特征进行提取，将其作为节点内容特征的补充，以填补模型无法捕捉节点位置信息的不足。同时本文还提出将教师模型中节点的隐藏层特征蒸馏给学生模型，以帮助多层感知机更好地学习教师模型内部的表征能力。本研究工作通过一系列实验验证了本文提出的方法的有效性，并分析了其合理性和可行性。但是，本研究还没有在较大的数据集上对本模型的有效性进行验证，这是本文后期工作应该补充的地方。

关键词： 图神经网络，知识蒸馏，鲁棒性，多层感知机，模型效率

[ABSTRACT]

Graph Neural Networks (GNNs) has excellent performance in processing graph structure data, and has been widely used in social network, recommendation system, biomedical discovery, financial risk control and other fields. There are high requirements for real-time performance in the application of recommendation system, financial risk control etc. In non-Euclidian space, the node-fetching latency by graph neural network is mainly caused by multi-hop data dependency. As a result, GNNs are difficult to be deployed in real applications due to the speed of graph neural network cannot be improved fundamentally. GNNs existing inference acceleration methods, such as pruning and quantization, although can be reduced by multiplication and accumulation operations, to some extent speed up the inference. However, their improvement is limited because the dependence of GNNs on graph has not been solved. The existing Knowledge Distillation (KD) method attempts to train Multilayer Perceptrons (MLPs) as students' model by using soft tags from GNNs to solve the model's dependence on graph when inferring. However, the effectiveness and robustness of the trained MLPs are limited. The reasons for the limitation are as follows: 1. The model does not capture the position information of nodes; 2. Strict matching of teachers' soft tags makes it impossible to actually learn the knowledge of capturing graph structure information and under the influence of noise, the performance of the model is significantly decreased. The robustness of the student model after knowledge distillation is poor. In order to improve the effectiveness of the MLP after KD, we propose to extract the position features of nodes as a supplement to the node content features, so as to fill the deficiency of the model that cannot capture the node position information. At the same time, we propose distillate the hidden layer features of the nodes in the teacher model into the student model to help MLP learn the representation ability of the teacher model. We verify the validity of our method through a series of experiments, and analyze its rationality and feasibility. However, we have not verified the validity of our model on a large data set, which is what our later work should be supplemented.

Keywords: Graph Neural Networks, Knowledge Distillation, Robustness, Multi-layer Perceptrons, Model Efficiency

目录

1 绪论

本章主要对选题背景与意义进行阐述，介绍国内外的研究现状和相关工作，最后简短地总结了本文的工作和创新点。

1.1 选题背景与意义

虽然深度学习已经能够很好的捕捉欧几里得数据的特征，但是在越来越多的应用中，数据从非欧几里得空间中生成，并被表示为节点间具有复杂关系和相互依赖关系的图。因此，如何有效拓展深度学习在图数据的应用得到了越来越多学者的关注。

现实生活中，图是一种被普遍使用的数据结构，能够捕获节点间的相互作用信息，具有实体交互的知识容易被保存在图结构中，便于存储和使用。从社交网络到推荐系统，从铁路系统到生物医学，图结构无处不在。很多应用场景都可以用图结构来表达，通过对图结构的分析，能帮助我们获得许多重要的隐藏信息。如在化学研究领域中，分子的结构被建模为图，它们的生物活性往往跟它们的结构息息相关；在电子商务系统中，用户与商品之间的交互可以被表征为图，对图进行分析，可以高效提取用户与商品的关系，从而对用户进行进一步的商品推荐和个性化服务；在论文引用网络中，每篇论文被建模为节点，论文之间的引用是节点之间的联系，被建模为边，通过对网络图进行分析，可以科学地对每篇文章进行归类和划分[1]。

图神经网络在处理非欧几里得数据结构表现优越，并且在图挖掘任务中取得了最先进的性能。现代的图神经网络的成功建立在消息传递机制的使用之上。在非欧式空间中，图神经网络对数据特征的获取依赖于多次跳跃的邻居节点的信息，这个过程是耗时和计算密集型的，导致图神经网络的推理速度无法从根本上得到提高[2]，对模型进行频繁更新的成本也比较高[3]。这使得图神经网络在大规模应用中的使用受到限制。其中在多个以图为数据结构的应用领域中，如推荐系统、金融风控等，对模型分析数据的实时性表现出较高的要求。为了改善图神经网络更新成本高的缺陷，我们必须从模型结构部署出发，改善模型。

知识蒸馏技术的核心思想，是将一个拥有良好性能的、复杂的大模型学习到的知识，转移到一个表达能力有限的小模型上，使得小模型具有与大模型相当的性能，但是参数数量大幅降低，从而实现模型的压缩与加速[4]。受知识蒸馏思想

的影响，有学者尝试在图神经网络模型上进行知识蒸馏，虽然模型推理速度有所提升，但是由于种种原因，经过知识蒸馏后的学生模型在鲁棒性，有效性等方面的表现仍不如原来的教师模型。如何进一步改善和优化经过图神经网络知识蒸馏后的学生模型的鲁棒性和有效性，成为改善模型更新成本、提高模型预测效率的关键和难点所在。

1.2 国内外研究现状和相关工作

1.2.1 神经网络的相关工作

1) **提升 GNNs 推理速度的方法**

图神经网络（Graph Neural Networks，GNNs）的推理速度受限于多跳邻居节点信息的获取，为了加速 GNNs 的推理速度，有在针对硬件进行改进的方法[5] [6]，如针对深度学习的加速器 Eyeriss 通过提供更好的数据复用来降低模型的功耗[5]；也有针对算法进行改进的方法，如剪枝[7]、量化等[8]，这些方法通过减少乘法和累积操作，从而加快 GNNs 的推理速度，但是并没有从根本上解决对邻居节点信息获取的高延迟问题，使得对 GNNs 的改善十分有限。同时，Graph-MLP 试图通过训练一个具有邻居对比损失的多层感知机（Multilayer Perceptron，MLP）避免 GNN 对邻居节点信息的获取[9]，但它只考虑了直推式（transductive, tran）设置，而没有考虑更符合实际的归纳式（inductive, ind）设置。

2) **图神经网络的知识蒸馏**

现有的知识蒸馏方法试图将知识从一个大的图神经网络蒸馏给一个轻量级的图神经网络。LSP[10] 和 TinyGNN[11]都是在进行知识蒸馏的同时保存局部信息。CPF[12] 从一个任意学习的 GNN 模型（老师模型）上提取知识到一个设计良好的学生模型，学生模型由标签传播和特征转换两种预测机制组成，分别保留数据基于结构和基于特征的先验知识，该方法可以同时兼顾到 GNNs 学习到的知识和数据的先验知识，但由于 CPF 仍然使用标签传播机制，所以它对图仍具有严重的依赖性。GraphSAIL[3]提出了一种基于知识蒸馏方法的增量学习范式训练图神经网络，将蒸馏分文本地结构蒸馏、全局结构蒸馏和自嵌入蒸馏三部分，其中自嵌入蒸馏可以防止每个嵌入向量的剧烈变化，局部和全局结构的蒸馏，用于保存拓扑信息。GLNN[2]将 GNN 模型学到的知识过滤到 MLP 上，解决 GNNs 模型对图的依赖性问题，且加快了推理速度，但该方法没有考虑数据的先验知识，经过训练后的 MLP 对噪声敏感，鲁棒

性和泛化能力差表现都不如 GNNs。

3) **节点位置特征的提取**

大多数 GNNs 都是基于消息传递机制工作的，即它们通过聚合局部的邻节点信息来建立当前节点的表示。这意味着这类 GNNs 从根本上是结构性的，即节点的表示只依赖于图的局部结构。这种表示对节点特征功能的诠释具有局限性[13]。举个例子，在消息传递机制中，一个分子中具有相同邻域的两个原子会具有相似的表示，但是，由于这两个原子在分子中的位置是不同的，所以它们的作用可能是不同的，它们对分子整体的相同特征也可能是有限的，因此只基于图的结构信息塑造节点特征是有欠缺的。流行的消息传递 GNN 无法区分具有相同的 1 跳局部结构的两个节点。

GNNs 已经成为图学习的首选框架，但是由于节点信息缺乏一个标准的位置表征，降低了 GNNs 的表示能力。目前，提取节点位置信息的一个可行方法是，引入节点位置编码，并将其输入到模型的输入层中。现有的方法中，NOSMOG[14]使用 DeepWalk 算法[15]对节点进行位置编码，节点的位置特征仅由图的结构特征和节点在图中的位置决定；MPGNNs-LSPE[13] 提出了一个使 GNN 能同时学习结构和位置信息表示的框架，节点位置信息的初始化由随机游走编码确定。但这些获取节点位置信息的方法存在计算复杂度高，低效等缺陷。

1.2.2 知识蒸馏的知识形式

知识蒸馏的思想与迁移学习[16]的思想相似，都涉及到知识的迁移。原始的知识蒸馏模型只是对教师模型输出的软标签进行匹配，但是当教师模型比较复杂时，这是不够的，由此衍生出了对教师模型中的其他知识进行蒸馏的学习方式。目前蒸馏的主要知识形式有输出特征知识、中间特征知识、中间特征知识、关系特征知识和结构特征知识[17]。四种知识关系如图 1.1图所示。

1) **中间特征知识**

Gotmare 等人[18]的研究表，教师模型输出的软标签可以指导学生模型在深层网络的训练，但在特征提取层的指导较少。当学生模型的网络层数增多时，即使学生模型通过软标签对教师模型进行了学习，教师模型和学生模型之间对数据的表征能力还是有所差距。提取教师模型的中间层特征进行蒸馏是解决学生模型和教师模型隐层特征差距较大的一种方法。中间特征知识是从教师模型的隐藏层中提取出的特征知识。中间特征知识蒸馏的思想是将从教师模型提取出来的隐藏层特征，用于在学生模型训练时充当学生模型隐藏层的

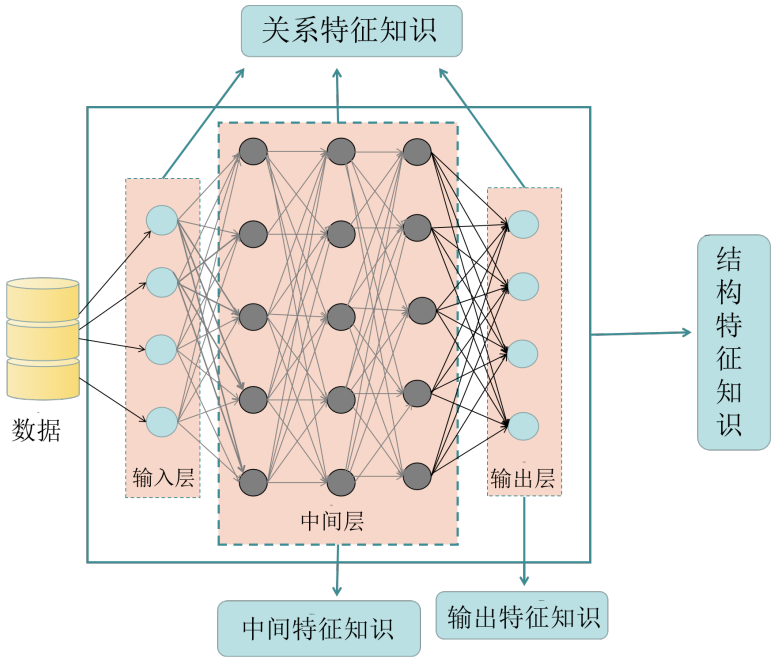

图 1.1 蒸馏的知识形式

提示，引导学生模型学到更多不同特征的表达能力，目标是要最小化教师与学生之间的中间特征映射距离，从而减少教师模型和学生模型在表征能力上存在的差距问题。最早使用教师模型隐藏层特征的是 FitNets[19]，它通过定义一个隐藏层损失，使学生模型和教师模型的隐藏层输出尽可能接近。它将隐藏层的损失定义为

$$L_{Hint}(W_{Guided}, W_r) = \frac{1}{2}||u_h(x; W_{Hint}) - r(u_g(x; W_{Guided}); W_r)||^2 , \qquad (1.1)$$

其中 W_{Hint} 是教师模型第 h 层的权重，W_{Guided} 是学生模型第 g 层的权重，W_r 是回归函数，用于处理学生模型的隐藏层特征与教师模型的隐藏层特征数据尺寸不一致问题，使得学生模型的隐藏层特征在经过 W_r 处理后，尺寸与教师模型相近。使用中间特征的知识蒸馏将隐藏层的损失函数，与原本教师模型输出的软标签的损失函数一起，组成一个新的损失函数。公式定义为

$$Loss_{total} = \lambda Loss_{hidden} + (1 - \lambda) Loss_{soft} , \qquad (1.2)$$

其中 $\lambda \in [0, 1]$。

2) **输出特征知识**

输出特征知识是指教师模型最后输出的逻辑单元和软标签。蒸馏输出特征知识主要是为了让学生模型的输出接近教师模型的输出，以达到和教师模型类

似的预测效果。一般在硬目标（原始数据的分类结果或标签）损失函数之外，会定义一个软目标的损失函数，用于最小化教师模型的输出和学生模型输出的差异。

3) **关系特征知识**

关系特征知识是指教师模型中不同层和数据样本知识之间的关系。关系特征知识的重点是为学生模型学习教师模型的关系知识提供一个恒等的从网络层到样本数据的关系映射。最早的关系特征知识蒸馏是 Yim 等人[20]的 “Flow of Solution Procedure”（FSP）矩阵，其通过模仿老师生成的 FSP 矩阵对学生进行指导。FSP 矩阵用于网络层的关系特征知识蒸馏要求网络中间层具有相同大小和数量的过滤器，这对不同架构的网络层并不适用。为了寻求更具普遍性的可以捕获网络层内部关系特征的方法，相关学者使用了雅可比矩阵[21]和使用径向基函数计算网络层间的相似性[22]。

4) **结构特征知识**

网络的性能不仅取决于网络的参数或关系，还取决于教师模型完整的知识体系。结构特征知识即教师模型完整的知识体系，包括教师模型的输出特征知识，关系特征知识，中间特征知识和区域特征分布知识等。对不同的任务，结构特征知识的组成成分有所不同，有结合样本特征、样本间关系和特征空间变换作为结构化的知识，也有由隐藏层特征、输出特征和全局预测特征组成的结构特征知识。结构特征知识的获取方式多种多样，可以通过生成对抗神经网络（Generative Adversative Networks, GANs）[23]生成，也可以借助对比学习[24]来获得。

1.3 论文主要工作和创新点

本文首先对图神经网络和知识蒸馏这两方面进行了深入的调研，对各类提升 GNNs 性能的方法以及知识蒸馏在图神经网络方面的应用进行了总结和归纳，分析出它们的优缺点。针对现有方法的不足，我们提出一种在新的图神经网络上的知识蒸馏方法，主要做了以下两点创新：1. 针对学生模型推理出的内容特征与标签空间不匹配问题[14]，在原节点内容特征的基础上加入了节点的位置特征进行训练；2. 针对知识蒸馏后的 MLP 对噪声敏感，导致在噪声影响下，模型性能降低的问题[14]，我们提出让学生模型也学习教师模型 GNNs 的隐藏层特征，而不仅是对教师模型输出的软标签进行硬匹配，从而增强学生模型的鲁棒性。

具体地，节点位置特征的提取，我们采用对图的拉普拉斯矩阵进行特征值分

解，分别取最大的 100 个特征值和最小的 100 个特征值所对应的特征向量，作为节点的位置特征，与节点内容特征拼接在一起，作为节点的总特征，对模型进行训练。此外，我们还为每个特征向量分别设置了一个可学习权重，根据每个向量提供的节点位置信息量自适应调整权重。降低对模型噪声的敏感性，我们定义了一个隐层特征相似性，衡量教师模型 GNNs 和学生模型 MLP 的隐层特征相似程度。在训练学生模型时，利用隐层特征相似性，对 MLP 进行训练，使学生模型能够学习到教师模型的隐藏层特征。我们使用了四种流行的图神经网络体系结构和五个著名的文章引用网络数据集，设计实验并验证了我们方法的有效性和优越性。

总的来说，我们的创新点可以总结为以下几个方面：

1. 我们提出将节点位置特征与节点的内容特征拼接，作为节点的新特征，进行学生模型的训练，解决节点内容特征与空间特征不匹配问题，填补了学生模型 MLP 无法捕捉图结构信息的不足。

2. 我们注意到学生模型 MLP 相比教师模型 GNNs 对噪声表现更敏感，于是提出让学生模型也学习教师模型的隐藏层特征，降低学生模型对噪声的敏感性，增强学生模型的鲁棒性。

1.4 本文的论文结构与章节安排

本文共分为五章，各章节内容安排如下：

第一章为绪论。简单说明本文章的选题背景与意义，国内外研究现状和相关工作，以及本文的主要工作和创新点。

第二章为相关工作综述。介绍知识蒸馏和图神经网络的相关工作。

第三章为模型设计。详细介绍相关符号和预备知识，具体描述本文的研究问题，阐述本文提出的模型和实现。

第四章为实验。具体展示本文提出模型的实验结果，并与之前其他学者所提出的模型效果进行对比，验证本文模型的有效性和优越性。

第五章为总结与展望。总结了本文的工作，并对未来工作进行展望。

2 相关工作

本章将介绍与本文研究内容相关的一些工作，内容包括图神经网络的发展、常见的框架、参数的学习、模型自身的局限性，以及基于图的 GNNs 的知识蒸馏研究和学生模型为 MLPs 的 GNNs 知识蒸馏。

2.1 图神经网络

1) **GNN 的发展**

Sperduti 等人（1997 年）[25]最早将神经网络应用在有向无环图上，激发了图神经网络的早期研究。后来，Gori 等人（2005 年）[26]最初提出了图神经网络的概念。并在 Scarselli（2009 年）[27]和 Gallicchio 等人（2010 年）[28]的研究中得到进一步的阐述。这些早期的研究都属于递归神经网络（Recurrent Graph Neural Networks，RecGNNs），它们通过迭代传播邻居节点的信息学习目标节点的表示，直到收敛。但是这个过程计算量很大。

随后，受卷积神经网络（Convolutional Neural Networks, CNNs）[29]在计算机视觉领域成功的鼓舞，涌现了大量重新定义图数据卷积概念的方法。卷积神经网络在图领域得到推广，它拓展了现有的神经网络，用于处理在图领域中所表示的数据。后来由 GCN[30]将其简化为基于消息传递的神经网络（MPNN）[31]。现在大部分的图神经网络都是基于消息传递机制。GAT[32]网络采用了注意力机制。PPNP[33]采用了个性化的页面排名，DeeperGCN[34]则采用剩余连接和密集连接。

2) **GNN 的框架**

图神经网络的定义首次在 2008 年[27]被提出。在图中，每个节点都由其特征和相关节点自然定义。GNN 的目标是学习一个包含节点领域信息的状态嵌入 $h_v \in R^s$。其中状态嵌入 h_v 是节点 v 的一个 s 维的向量，用于训练模型得到输出标签 o_v。设 f 是一个参数函数，为局部转移函数，它被所有节点共享，并根据邻节点输入更新节点的状态。设 g 是描述如何产生输出的局部输出函数。f 和 g 的计算可以被解释为前馈神经网络。h_v 和 o_v 的关系如下：

$$h_v = f(X_v, X_{co[v]}, h_{ne[v]}, X_{ne[v]}), \tag{2.1}$$

$$o_v = g(h_v, x_v)\,, \tag{2.2}$$

其中，x_v 是节点 v 的特征，$x_{co[v]}$ 是与节点 v 相连接的边的特征，$h_{ne[v]}$ 是节点 v 的状态，$x_{ne[v]}$ 是节点 v 的邻节点的特征。

令 H 为叠加所有状态后构建的向量，O 为所有输出叠加后的向量，X 为所有特征叠加后的向量，X_N 为所有节点特征叠加后的向量。它们之间的数学关系为

$$H = F(H, X)\,, \tag{2.3}$$

其中 F 为全局转移函数，它是 f 的叠加版本。H 的值是固定不变的，F 为一个缩放映射。

$$O = G(H, X_N)\,, \tag{2.4}$$

G 为全局输出函数，它是 g 的叠加版本。

根据 Banach 的不动点理论[35]，GNN 使用以下经典的迭代策略来计算状态：

$$H^{t+1} = F(H^t, X)\,, \tag{2.5}$$

其中 H^t 为 H 的第 t 次迭代。对任意初始的 $H(0)$，更新过程以指数速度快速收敛。

3) **GNN 的参数学习**

在目标信息的监督下，损失函数可以定义为：

$$loss = \sum_{i=1}^{p}(t_i - o_i)\,, \tag{2.6}$$

其中 p 为被监督的节点个数。算法基于梯度下降进行更新，主要由以下几个部分组成。

(a) 状态 h_v^t 通过公式2.1迭代更新，知道 T 次迭代后，达到公式2.3的收敛点，此时有 $H(T) \approx H$.

(b) 权重 W 的梯度由 $loss$ 计算得到。

(c) 权重 W 的更新根据最后一步计算得到的梯度进行更新。

4) **图神经网络的局限**

虽然大量的研究已经表明，图神经网络在处理结构化数据表现出优越的性能，但是传统的图神经网络仍然具有一些局限性。传统图神经网络的局限性主要如下[36] [37]：

1）迭代更新不动点的隐藏层状态是低效的；

2）无法有效捕捉图中边的信息特征。例如，在知识图中，不同的边具有不同关系类别，在标签传播时，应该根据不同类型的边进行不同的传播；

3）如何对边的隐藏层状态进行学习也是目前图神经网络的有待解决的问题；

4）现有的 GNNs 大多数是半监督学习，其性能很大程度上依赖于高质量的标注数据；

5）随着图数据规模发展越来越大，图神经网络的模型设计也越来越复杂，这给模型的存储带来了新的困难和挑战。

2.2 基于图的 GNNs 的知识蒸馏

基于图的图神经网络知识蒸馏方法（GKD）[37]从图的卷积中间层或输出层中提取知识，为了进一步分析输入图中节点之间的关系，GNNs 可以利用所构建的图，深入挖掘图的拓扑结构和节点关系信息。GKD 的工作框架如图 2.1图所示：

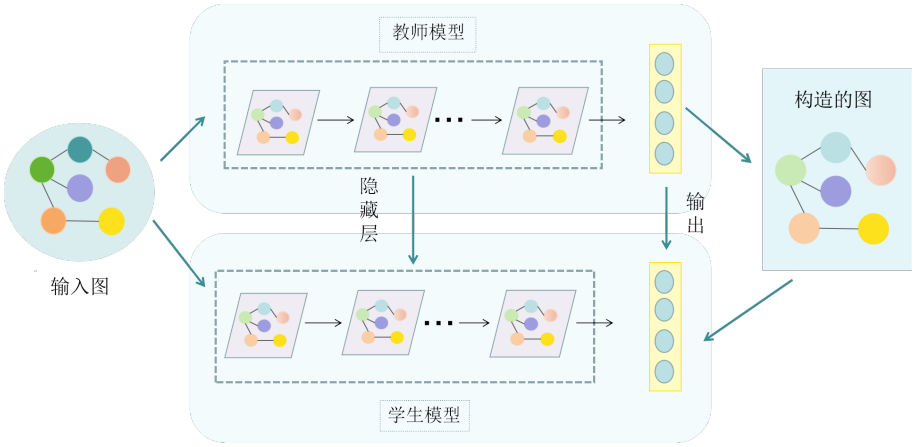

图 2.1 GKD 的工作框架

一般基于图的图神经网络知识蒸馏分为以下四个步骤[37]。

1）将输入图送入教师模型和学生模型进行训练，得到各自的中间层特征和节点关系；

2）用相似度计算函数计算内部节点间的相关性；

3）利用距离测量函数计算教师模型和学生模型各自内部节点嵌入间的差值；

4）通过积累知识蒸馏的损失，将教师模型学习到的拓扑知识和节点关系知识传授给学生模型。

最终的基于图的图神经网络知识蒸馏损失函数表示为：

$$\mathcal{L}_G = \sum_{l \in L} \sum_{(x,x') \in x^2} D_G(S(x_s,^l, x_s'^l), S(x_t^l, x_t'^l)) , \tag{2.7}$$

其中，x_s^l 和 $x_s'^l$ 表示 GNN 第 l 层的学生模型中的两个节点，同样地，x_t^l 和 $x_t'^l$ 表示第 l 层教师模型中的两个节点。S 为 GNN 卷积层和输出层节点的相似性计算函数，D_G 为使学生模型和教师模型的构造图距离最小的度量函数，常见的有 KL 散度，MSE，Huber，MAE 等。

2.3 学生模型为 MLP 的 GNNs 知识蒸馏

在学生模型为 MLP 的 GNNs 知识蒸馏模型 GLNNs[2] 中，教师模型 GNN 会被离线训练，得到数据的软标签。学生模型 MLP 也会被离线训练，但是学生模型 MLP 不仅被要求输出标签要与节点真实标签差距最小，还被要求输出的软标签要与教师模型输出的软标签尽可能一致。这是通过重新定义训练学生模型的损失函数实现的。最后，训练完后的 MLP 会被用于新数据的预测，这样实际进行部署的模型就由 GNN 变为轻量级模型 MLP。学生模型为 MLP 的 GNNs 知识蒸馏的工作框架如图 2.2图所示：

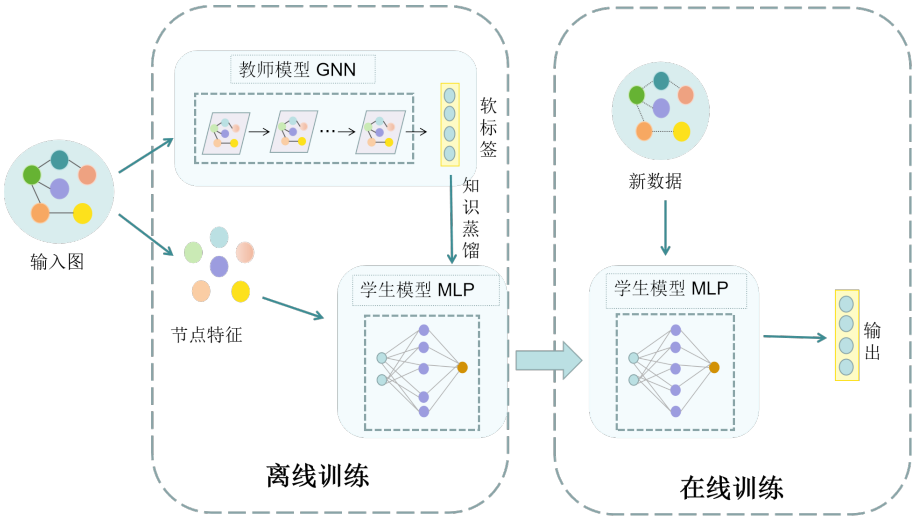

图 2.2 学生模型为 MLP 的 GNN 知识蒸馏的工作框架

和 GNNs 作为学生模型的损失函数设计类似，MLPs 的损失函数定义为

$$\mathcal{L} = \lambda \sum_{v \in \mathcal{V}^L} \mathcal{L}_{label}(\hat{y}_v, y_v) + (1-\lambda) \sum_{v \in \mathcal{V}} \mathcal{L}_{teacher}(\hat{y}_v, z_v) , \tag{2.8}$$

其中，$\hat{y}$ 是学生模型 MLP 的预测结果，y_v 是节点的真实标签，z_v 是教师模型 GNN

的预测软标签，λ 是权重，用于衡量学生模型的预测结果与真实标签和软标签之间的损失。$\mathcal{L}_{label}$ 是真实标签 y_v 和学生预测结果 $\hat{y}_v$ 的交叉熵。$\mathcal{L}_{teacher}$ 是 KL 散度计算函数。

在多分类任务下，交叉熵损失函数通常表示为：

$$L = \frac{1}{N}\sum_{i} L_i = -\frac{1}{N}\sum_{i}\sum_{c=1}^{M} y_{ic} log(p_{ic}) , \tag{2.9}$$

其中，M 为样本类别的数量，y_{ic} 为符号函数，取值为 0 或 1，如果样本 i 的类别为 c，则 $y_{ic} = 1$，否则 $y_{ic} = 0$。p_{ic} 为样本 i 属于类别 c 的预测概率。

KL 散度的计算将在下一小节中详细介绍。

3 模型设计

本章节将对本文要解决的问题进行描述，展示一些问题的数学描述，以及介绍与本文提出的模型有关的前置知识。

3.1 符号与预备知识

3.1.1 图的符号表示

图可根据边是否有方向，分为有向图和无向图，也可根据边的权值分为有权图和无权图。一般使用 $G(V,E)$ 表示一个图，其中 V 表示节点集合 $v_1, v_2, v_3, ..., v_n$，E 表示边的集合。$n = |V|$ 为图 G 中的节点个数，$m = |E|$ 为边数。对于一个具有 n 个节点的图 G，可以用一个 $n \times n$ 的权重矩阵 W 表示，矩阵 W 也称为图的邻接矩阵。如果从节点 i 到节点 j 有一条边，则 w_{ij} 为从顶点 i 到顶点 j 的边的权重。如果从顶点 i 到顶点 j 不存在边，则 w_{ij} 可根据需求设为 0 或-1。在无权图中，若从顶点 i 到顶点 j 的边存在，则 w_{ij} 为 1，否则为 0。在无向图中，矩阵 W 是一个对称矩阵。

由图的定义，我们容易得到无权图的度矩阵 D。矩阵 D 是一个对角矩阵，对角线上的元素 d_{ii} 表示节点 v_i 的度。其中

$$d_{ii} = \sum_j w_{ij} \ . \tag{3.1}$$

3.1.2 拉普拉斯算子

多元函数 $f(x_1, ..., x_n)$ 的拉普拉斯算子是所有自变量的非混合二阶偏导数之和，

$$\Delta f = \sum_{i=1}^{n} \frac{\partial^2 f}{\partial x_i^2} \ . \tag{3.2}$$

比如，对于三元函数，拉普拉斯算子为

$$\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} \ . \tag{3.3}$$

一阶导数的计算可以用单侧差分公式近似：

$$f^{'}(x) \approx \frac{f(x+\Delta x)-f(x)}{\Delta x}, \Delta x \to 0 \,. \tag{3.4}$$

对于二阶导数，有

$$\begin{aligned} f^{''}(x) &\approx \frac{f^{'}(x)-f^{'}(x-\Delta x)}{\Delta x} \\ &\approx \frac{\frac{f(x+\Delta x)-f(x)}{\Delta x}-\frac{f(x)-f(x-\Delta x)}{\Delta x}}{\Delta x} \\ &= \frac{f(x+\Delta x)+f(x-\Delta x)-2f(x)}{(\Delta x)^2} \,. \end{aligned} \tag{3.5}$$

对于二元函数 $f(x,y)$，其拉普拉斯算子可以用下面的公式近似：

$$\begin{aligned} \Delta f &= \frac{\partial^2 f}{\partial x^2}+\frac{\partial^2 f}{\partial y^2} \\ &\approx \frac{f(x+\Delta x,y)+f(x-\Delta x,y)-2f(x,y)}{(\partial x)^2}+\frac{f(x,y+\Delta y)+f(x,y-\Delta y)-2f(x,y)}{(\Delta y)^2} \,. \end{aligned} \tag{3.6}$$

对函数值进行离散化采样，得到一系列点值矩阵，如下：

$$\begin{bmatrix} f(x_1,y_1) & f(x_2,y_1) & \dots & f(x_n,y_1) \\ f(x_1,y_2) & f(x_2,y_2) & \dots & f(x_n,y_2) \\ \dots & \dots & \dots & \dots \\ f(x_1,y_n) & f(x_2,y_n) & \dots & f(x_n,y_n) \end{bmatrix} \,. \tag{3.7}$$

为了简化，假设 x，y 的增量都为 1，即

$$\Delta x = x_{i+1}-x_i = 1, \Delta y = y_{i+1}-y_i = 1 \,. \tag{3.8}$$

则点 (x_i, y_i) 处的拉普拉斯算子为

$$\begin{aligned}\Delta f(x_i, y_j) &= \frac{f(x_i + \Delta x, y_j) + f(x_i - \Delta x, y_j) - 2f(x_i, y_j)}{(\Delta x)^2} \\ &\quad + \frac{f(x_i, y_j + \Delta y) + f(x_i, y_j - \Delta y) - 2f(x_i, y_j)}{(\Delta y)^2} \\ &= \frac{f(x_i + \Delta x, y_j) + f(x_i - \Delta x, y_j) - 2f(x_i, y_j)}{1^2} \\ &\quad + \frac{f(x_i, y_j + \Delta y) + f(x_i, y_j - \Delta y) - 2f(x_i, y_j)}{1^2} \\ &= f(x_{i+1}, y_j) + f(x_{i-1}, y_j) + f(x_i, y_{j+1}) + f(x_i, y_{j-1}) - 4f(x_i, y_j)\ .\end{aligned} \tag{3.9}$$

从式3.9中可以看到，点 (x_i, y_j) 处的拉普拉斯算子等于其周围四个点的函数值之和减去 4 倍该点的函数值。因此，拉普拉斯算子也可以表示为

$$\begin{aligned}\Delta f &= f(x_{i+1}, y_j) - f(x_i, y_j) + f(x_{i-1}, y_j) - f(x_i, y_j) \\ &\quad + f(x_i, y_{j+1}) - f(x_i, y_j) + f(x_i, y_{j-1}) - f(x_i, y_j) \\ &= \sum_{(x_k, y_l) \in N(x_i, y_j)} (f(x_k, y_l) - f(x_i, y_j))\ ,\end{aligned} \tag{3.10}$$

其中，(x_k, y_l) 是点 (x_i, y_j) 的邻居节点，$N(x_i, y_j)$ 为点 (x_i, y_j) 的邻居节点的集合。

3.1.3 拉普拉斯矩阵

将拉普拉斯算子推广到图领域，由于图的边具有权重，因此可以在计算拉普拉斯算子时，使用权重信息。在顶点 i 处的拉普拉斯算子可以表示为

$$\Delta f_i = \sum_{j \in N_i} w_{ij}(f_i - f_j)\ . \tag{3.11}$$

这里将 f_i 和 f_j 的位置进行了互换，与前面介绍的拉普拉斯算子相比，加上负号后与之等价。

在图中，当两个节点 i 和 j 之间没有边连接时，$w_{ij} = 0$，因此式子3.11可以表示为

$$\begin{aligned}\Delta f_i &= \sum_{j \in V} w_{ij}(f_i - f_j) \\ &= \sum_{j \in V} w_{ij} f_i - \sum_{j \in V} w_{ij} f_i \\ &= d_i f_i - w_i f\ ,\end{aligned} \tag{3.12}$$

其中，d_i 为节点 i 的加权度，w_i 为邻接矩阵 W 的第 i 行，f 为由所有顶点的值构成的列向量。

对图中所有顶点，有

$$\begin{aligned}\Delta f &= \begin{bmatrix}\Delta f_1\\ \dots\\ \Delta f_n\end{bmatrix} = \begin{bmatrix}d_1 f_1 - \mathrm{w_1 f}\\ \dots\\ d_n f_n - \mathrm{w_n f}\end{bmatrix}\\ &= \begin{bmatrix}d_1 & \dots & \dots\\ \dots & \dots & \dots\\ \dots & \dots & d_n\end{bmatrix}\begin{bmatrix}f_1\\ \dots\\ f_n\end{bmatrix} - \begin{bmatrix}\mathrm{w_1}\\ \dots\\ \mathrm{w_n}\end{bmatrix}\begin{bmatrix}f_1\\ \dots\\ f_n\end{bmatrix}\\ &= (\mathrm{D} - \mathrm{W})\mathrm{f}\ .\end{aligned} \tag{3.13}$$

由此，假设无向图 G 有 n 个顶点，邻接矩阵为 W，加权度矩阵为 D，则可定义拉普拉斯矩阵为

$$L = D - W\ . \tag{3.14}$$

由上面的介绍易知，拉普拉斯矩阵的实际含义为图的二阶导数。

3.1.4 KL 散度

KL 散度（Kullback-Leibler divergence, KL divergence）是用于描述两个概率分布 $Q(x)$ 和 $P(x)$ 相似度的一种度量，记作 $D(Q||P)$。对离散的随机变量，KL 散度定义为

$$D(Q||P) = \sum_i Q(i) log \frac{Q(i)}{P(i)}\ . \tag{3.15}$$

对连续的随机变量，KL 散度定义为

$$D(Q||P) = \int Q(x) log \frac{Q(x)}{P(x)} dx\ . \tag{3.16}$$

利用 Jensen 不等式可以知道

$$\begin{aligned}-D(Q||P) &= \int Q(x) log \frac{P(x)}{Q(x)} dx\\ &\le log \int Q(x) \frac{P(x)}{Q(x)} dx\\ &= log \int P(x) dx = 0\ .\end{aligned} \tag{3.17}$$

因此 KL 散度具有性质：$D(Q||P) \geq 0$，当且仅当 $Q = P$ 时，$D(Q||P) = 0$ 成立。

3.1.5 Frobenius 范数

Frobenius 范数，简称 F-范数，是一种矩阵范数，记为 $||\cdot||_F$。F-范数经常被用于比较真实矩阵和估计矩阵之间的相似性。具体地，设 $A = [a_{ij}]_{m\times n}$，是一个 $m \times n$ 的矩阵，则 A 的 Frobenius 范数定义为

$$||A||_F = \sqrt{tr(A^T A)} = \sqrt{\sum_{i,j} a_{ij}^2}\,. \tag{3.18}$$

若想让矩阵 B 近似矩阵 A，则可以令

$$B = \underset{B}{argmin}\, ||A - B||_F\,. \tag{3.19}$$

3.1.6 均方误差

数理统计中，均方误差（mean-square error, MSE）指参数预测值和参数真实值之差平方和的均值。通常用于衡量模型预测值与真实值的匹配程度，是回归问题中常见的损失函数。计算公式为

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2\,, \tag{3.20}$$

其中，y_i 为第 i 个数据的真实值，$\hat{y}_i$ 为第 i 个数据的预测值。

3.2 问题描述

虽然图形神经网络 (GNNs) 在处理非欧几里德结构数据方面能够取得良好的效果，但是由于多跳数据依赖性所带来的可扩展性限制，它们难以应用于实际的大规模数据。为了加速图神经网络的推理速度，出现了剪枝等优化方法，这些方法在一定程度上加速了模型的推理速度，但是并没有从根本上解决图神经网络依赖多跳邻居节点信息进行推理的问题，使得改善力度十分有限。为了突破此瓶颈，有学者提出用知识蒸馏的方法将图神经网络学习到的知识蒸馏给推理速度较快，即不依赖于多跳邻接节点信息的轻型模型。现有的方法试图通过使用从教师 GNNs 推理出的软标签，在节点内容特征上训练学生模型多层感知机（MLPs)，让学生模型推理出的软标签拟合教师模型推理出的软标签，从而使学生模型学习到教师模

型的知识。然后使用训练好的多层感知机进行推理，进而解决图神经网络模型的可扩展性问题。然而，经过训练的 MLP 对噪声表现敏感，缺乏有效性和鲁棒性。在本文中，我们总结了两个导致模型有效性和鲁棒性的缺乏的原因:

1) 模型只捕捉到了图的局部结构信息，没有捕捉到节点的位置信息，导致模型无法区分具有相同局部结构信息，不同位置信息的节点。
2) 由于 MLP 只对教师模型的软标签进行硬匹配，实际上无法学习到 GNNs 对图结构信息的捕捉能力，导致 MLP 对节点特征噪声比 GNNs 敏感。

3.3 模型框架

本文的方法在 GLNNs 模型框架的基础上，针对上述分析出的两点原因，对模型进行了创新，使经过知识蒸馏后的 MLP 在的有效性和鲁棒性上表现更胜一筹。我们对模型进行了两点创新，分别是

1) 提取节点位置信息将其作为节点特征的补充，帮助 MLP 捕捉节点的位置特征；
2) 同时将 GNNs 模型的隐藏层特征和输出的软标签蒸馏给 MLP，使 MLP 能学习到 GNNs 的内部表征能力。

3.3.1 提取节点位置特征的模型框架

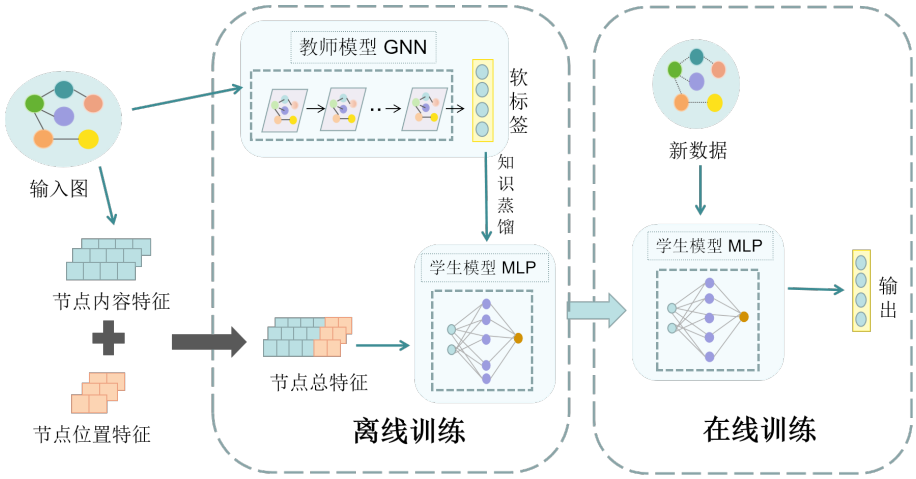

图 3.1 提取节点位置特征的模型框架

用拉普拉斯矩阵特征值分解的方法，提取节点的位置信息，将其作为节点特征的一部分，与节点内容特征向量拼接在一起，组成一个更能表征节点的向量。训练模型时，离线训练 GNN 模型，训练时输入的特征是节点的内容特征，然后提取 GNN 模型输出的软标签。在教师模型 GNN 软标签的指导下，离线训练学生

模型 MLP，输入的特征是合并节点内容特征和位置特征后的总特征。模型框架如图3.1所示。

3.3.2 蒸馏 GNNs 隐藏层特征的模型框架

从训练好的 GNN 模型中提取出节点的隐藏层特征表示，在训练学生模型时蒸馏给模型 MLP。我们定义了一个隐层特征相似度蒸馏损失（Hidden feature Similarity Distillation，HSD）$\mathcal{L}_{HSD}$，用于最小化 GNN 模型和 MLP 模型的隐层特征的距离。加入隐藏层特征蒸馏后的模型框架如图3.2所示。对 $\mathcal{L}_{HSD}$ 的定义将在下一小节中给出。

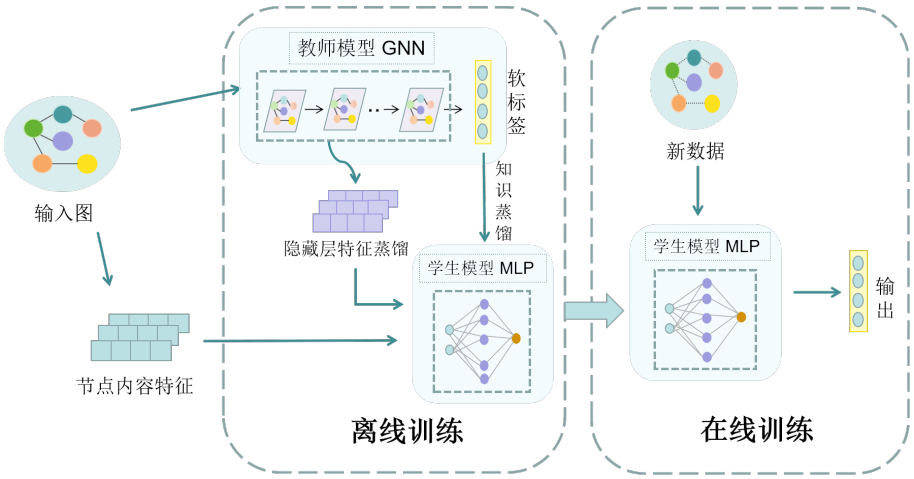

图 3.2 蒸馏 GNNs 隐藏层特征的模型框架

3.3.3 整体模型

1) **损失函数**

训练学生模型时，损失函数由三部分组成，一部分是学生模型的输出结果与节点的实际标签的交叉熵组成损失 $\mathcal{L}_{label}$，第二部分由学生模型的输出结果与教师模型输出的软标签的 KL 散度组成损失 $\mathcal{L}_{teacher}$。第三部分是由教师模型的节点隐藏层特征与学生模型的节点隐藏层特征的损失 $\mathcal{L}_{HSD}$。

用权值参数 λ 和 α 权衡损失 $\mathcal{L}_{teacher}$ 和损失 $\mathcal{L}_{HSD}$。假设 $\hat{y}_v$ 为学生模型的输出，y_v 为节点的真实标签，z_v 为教师模型输出的软标签，$\mathcal{V}^L$ 为带有真实标签的节点的集合，则模型的损失函数可以表示为式3.21：

$$\begin{aligned}\mathcal{L} &= \mathcal{L}_{label} + \lambda\mathcal{L}_{teacher} + \alpha\mathcal{L}_{HSD} \\ &= \sum_{v\in\mathcal{V}^L}\mathcal{L}_{label}(\hat{y}_v, y_v) + \lambda\sum_{v\in\mathcal{V}}\mathcal{L}_{teacher}(\hat{y}_v, z_v) + \alpha\mathcal{L}_{HSD},\end{aligned} \tag{3.21}$$

其中 $\mathcal{L}_{HSD}$ 将在下一小节中详细介绍。

2) **整体模型框架**

提取节点位置信息的同时对模型隐藏层特征进行蒸馏，模型的框架如图3.3所示。

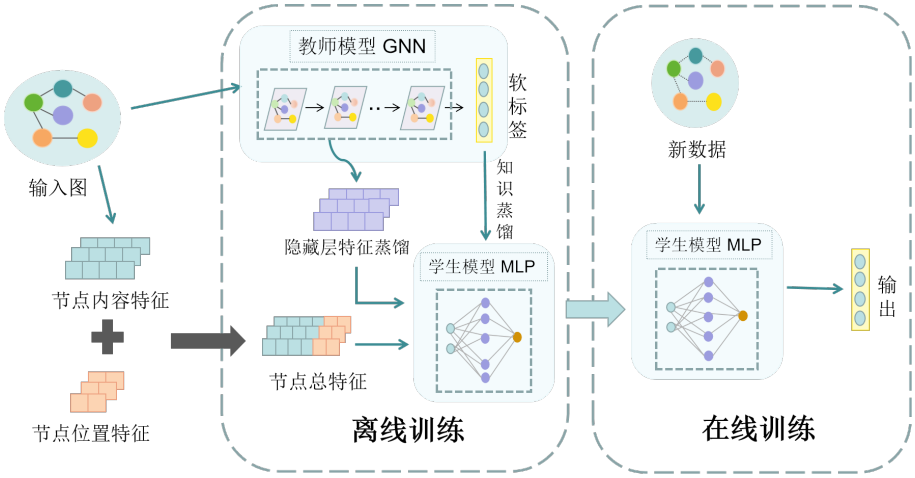

图 3.3 模型整体框架

3) **算法流程**

下面，我们将通过伪代码3.1的形式展示我们模型的训练过程。训练学生模型的时候，我们先将教师模型训练好，并将其软标签和隐藏层特征保存在文件中。

算法 3.1: 提取节点位置特征并蒸馏 GNN 隐藏层特征训练 MLP 的方法

输入: 图 $G=(V,E)$，$||V||=n$，$||E||=m$，节点标签 Y。参数 λ 和 α，以及系数矩阵 W。训练模式 mode（tran 或 ind）

计算 G 的拉普拉斯矩阵 L
提取 L 最大的 100 个特征值对应的特征向量组 V_{max}
提取 L 最小的 100 个特征值对应的特征向量组 V_{min}
$V \leftarrow [V_{max}\ V_{min}]$
如果 $mode = tran$ **则**
    数据集按直推式划分
否则
    数据集按归纳式划分
从文件中读取教师模型的软标签 Z 和隐藏层特征 H
对于每个 *epoch* **进行**
    使用真实标签训练:
        将 Y，S 和 V 作为模型输入，$V' \leftarrow C \cdot V$，$F \leftarrow [S\ V']$
        利用 F 进行前馈计算，得到输出 $\hat{Y}$
        计算误差 $L \leftarrow L_{label}(\hat{y}, y)$
        梯度置 0，误差反向传播，更新参数
    软标签蒸馏:
        将 Z，S 和 V 作为模型输入，$V' \leftarrow C \cdot V$，$F \leftarrow [S\ V']$
        利用 F 进行前馈计算，得到输出 $\hat{Y}$
        计算误差 $L \leftarrow \lambda L_{teacher}(\hat{y}, z)$
        梯度置 0，误差反向传播，更新参数
    隐藏层特征蒸馏:
        将 H，S 和 V 作为模型输入，$V' \leftarrow C \cdot V$，$F \leftarrow [S\ V']$
        利用 F 进行前馈计算，得到输出层前一层的隐藏层特征 $\hat{H}$
        $H' \leftarrow W \cdot \hat{H}$
        $S_{MLP} \leftarrow H' \cdot (H')^T$
        $S_{GNN} \leftarrow H \cdot (H)^T$
        计算误差 $L \leftarrow \alpha L_{HSD}(S_{MLP}, S_{GNN})$
        梯度置 0，误差反向传播，更新参数
    计算验证集在模型上的准确率
    如果 在验证集上的准确率在 *50* 个 *epochs* 内没有提高 **则**
        停止模型的训练
    如果 *epoch* 超过设置的最大 *epochs* **则**
        停止模型的训练
计算测试集在模型上的准确率和误差
输出结果

3.4 具体实现

在本小节中，我们将介绍上一小节中提出的新的模型框架的实现细节，包括节点位置特征的提取和隐藏层特征的蒸馏。

3.4.1 位置特征的提取

通过对图的拉普拉斯矩阵进行特征值分解，提取最大的 100 个特征值和最小的 100 个特征值对应的特征向量，并为每个向量分别赋予一个可学习的权重 w_i，作为节点的位置特征。

假设向量 $\boldsymbol{v}_1$，$\boldsymbol{v}_2$，...，$\boldsymbol{v}_{100} \in \mathbb{R}^{n\times 1}$ 为拉普拉斯矩阵 L 最大的 100 个特征值对应的特征向量，向量 $\boldsymbol{u}_1$，$\boldsymbol{u}_2$，...，$\boldsymbol{u}_{100} \in \mathbb{R}^{n\times 1}$ 为拉普拉斯矩阵 L 最小的 100 个特征值对应的特征向量。$V = [\boldsymbol{v}_1\boldsymbol{v}_2...\boldsymbol{v}_{100}] \in \mathbb{R}^{n\times 100}$，$U = [\boldsymbol{u}_1\boldsymbol{u}_2...\boldsymbol{u}_{100}] \in \mathbb{R}^{n\times 100}$，$n$ 为图中的节点个数。w_1，w_2，...，w_{200} 分别为 v_i 和 u_j 的可学习权重，其中 $i = 1, 2, ..., 100$，$j = 1, 2, ..., 100$。经过可学习权重的调整后，最终节点位置特征矩阵 P 表示为

$$\boldsymbol{P} = \begin{bmatrix} w_1\boldsymbol{v}_1 & w_2\boldsymbol{v}_2 & \dots & w_{100}\boldsymbol{v}_{100} & w_{101}\boldsymbol{u}_1 & w_{102}\boldsymbol{u}_2 & \dots & w_{200}\boldsymbol{u}_{200} \end{bmatrix} . \tag{3.22}$$

假设节点的内容特征为 $\boldsymbol{c}_1$，$\boldsymbol{c}_2$，...，$\boldsymbol{c_n}$，$c_i \in \mathbb{R}^{1\times m}$，

$$C = \begin{bmatrix} c_1 \\ c_2 \\ \dots \\ c_n \end{bmatrix} , \tag{3.23}$$

m 为节点内容特征的维度。节点 i 的内容特征与位置特征拼接后的总特征 $\boldsymbol{f_i} \in \mathbb{R}^{1\times(m+200)}$，可以表示为

$$\boldsymbol{f_i} = \begin{bmatrix} \boldsymbol{c_i}\boldsymbol{p_i} \end{bmatrix} . \tag{3.24}$$

3.4.2 隐藏层特征的蒸馏

从训练好的 GNN 模型中提取度数最大的 $p\%$ 的节点的隐藏层特征表示 $H_G \in \mathbb{R}^{N'\times d_G}$，在训练 MLP 过程中，也提取出 MLP 模型中度数最大的 $p\%$ 的节点的隐藏层特征表示 $H_M \in \mathcal{R}^{N'\times d_M}$，其中 $N' = \lfloor N \times p\% \rfloor$，$d_G$ 表示 GNN 模型中节点隐藏层特征的维度，d_M 表示 MLP 模型中节点隐藏层特征的维度。定义 GNN 的节点内

隐层特征相似度矩阵 S_{GNN} 为

$$S_{GNN} = H_G \cdot (H_G)^T \ . \tag{3.25}$$

定义 MLP 的节点内隐层特征相似度矩阵 H_{MLP} 为

$$S_{MLP} = H_M^{'} \cdot (H_M^{'})^T \ , \tag{3.26}$$

其中

$$H_M^{'} = \sigma(W_M \cdot H_M) \ , \tag{3.27}$$

W_M 为系数矩阵，用于调整 H_G 和 $H_M^{'}$ 的数据保持在同一个数量级上。
最后我们定义 $\mathcal{L}_{HSD}$ 如下：

$$\mathcal{L}_{HSD} = ||S_{GNN} - S_{MLP}||_F^2 \ , \tag{3.28}$$

其中，$||\cdot||_F$ 为 Frobenius 范数。

4 实验

本章节将介绍实验的相关细节，包括实验设置和实验结果展示。在实验设置中我们将具体介绍我们实验所用的数据集、评估指标和数据划分方式，以及我们实验的设计和实现。在实验结果展示中，我们对实验结果进行了探讨和分析。

4.1 实验设置

在本小节中，我们将详细介绍实验部分用到的数据集、数据处理和数据划分方式，以及模型评估的方法。

4.1.1 数据集

本文使用了在图研究领域常用的五个数据集，分别是被广泛使用的节点分类数据集 Cora、Citeseer 和 Pubmed，以及 A-Computer 和 A-Photo 数据集。数据集的信息如下:

表 4.1 数据集信息

数据集	节点数	边数	特征数	类别数
Cora	2,485	5,069	1,433	7
Citeseer	2,110	3,668	3,703	6
Pubmed	19,717	44,324	500	3
A-Computer	13,381	245,778	767	10
A-Photo	7,487	119,043	745	8

Cora[38]是一个由机器学习论文组成的基准引用数据集，其中每个节点代表一个具有稀疏词袋特征向量的文档。边代表文献之间的引用，标签代表每篇论文的研究领域。

Citeseer[38]是另一个计算机科学出版物的基准引用数据集，与 Cora 的配置类似。在本文使用的五个数据集中，Citeseer 数据集具有最多的特征。

Pubmed[38]也是一个引用数据集，由 Pubmed 数据库中与糖尿病相关的文章组成。节点特征为 TF/IDF 加权词频，标签表示本文所讨论的糖尿病类型。

A-Computers 和 A-Photo[39]提取自 Amazon co-purchase graph，其中节点表示产品，边表示两种产品是否经常共同购买，特征表示用 bag-of-words 编码的产品评论，标签是预定义的产品类别。

4.1.2 评估指标

对于本文中的实验，本文运行通过多次随机种子的结果，得到平均值和标准偏差。模型性能以准确性来衡量，展示了测试集数据的结果，并使用了验证集的数据选择最佳模型。

4.1.3 两种数据集划分方式

同 GLNNs[2], 给定图 $\mathcal{G}$, 节点内容特征 X 和节点标签 Y^L, 我们使用两种划分方式对数据集进行划分。第一种是 Transductive（Tran）划分方式，第二种是 Inductive（Ind）划分方式。

我们使用 X 表示节点特征，Y 表示节点标签，上标 U 表示没有标签的数据，L 表示有标签的数据。则有，$X = X^U \cup X^L$ 和 $Y = Y^U \cup Y^L$。下标 obs 表示观察到的数据。将数据 $\mathcal{V}^U$ 划分为两个不相交的部分 $\mathcal{V}^U_{obs}$ 和 $\mathcal{V}^U_{ind}$，即 $\mathcal{V}^U = \mathcal{V}^U_{obs} \cup \mathcal{V}^U_{ind}$。拿出 $v \in \mathcal{V}^U_{ind}$ 和任意与节点 $v \in \mathcal{V}^U_{ind}$ 相连的边，将整个图划分为没有共同顶点和边的两个不相交的图，即 $\mathcal{G} = \mathcal{G}_{obs} \cup \mathcal{G}_{ind}$。同理，节点特征和节点标签也被分成了三个不相交的集合，即 $X = x^L \cup X^U_{obs} \cup X^U_{ind}$ 和 $Y = Y^L \cup Y^U_{obs} \cup Y^U_{ind}$。

两种数据划分方式的定义如下：

1) **Transductive**

   在 $\mathcal{G}$，X 和 Y^L 上对模型进行训练，在 (X^U, Y^U) 上进行测试。知识蒸馏中使用的软标签 z_v 中 $v \in \mathcal{V}$。

2) **Inductive**

   在 $\mathcal{G}_{obs}$，X^L，X^U_{obs} 和 Y^L 上对模型进行训练，在 (X^U_{ind}, Y^U_{ind}) 上对模型进行测试。知识蒸馏中使用的软标签 z_v 中 $v \in \mathcal{V}^L \cup \mathcal{V}^U_{obs}$。

4.1.4 实现

1) **实验环境**

   本文的模型通过 PyTorch 实现，并使用 Adam 优化器训练模型。

2) **初步验证**

   对节点位置特征的提取，我们首先在 Cora、Citeseer、Pubmed 三个数据集上，用 SAGE 和 GCN 模型，分别采取了最小特征值对应的的特征向量 10、20、50、100、200 个和最大特征值对应的特征向量 10、50、100 个，初步验证对图的拉普拉斯矩阵进行特征分解，提取其特征向量作为节点的位置特征补充节点特征，对提升学生模型 MLPs 性能具有积极影响。

3) **改进实验**

在初步验证实验效果后，我们改进了实验。我们提取了拉普拉斯矩阵最大的 100 个特征值对应的特征向量和最小的 100 个特征值对应的特征向量，并为每个特征向量分配一个权重，让模型通过自适应的方式学习每个权重参数。模型收敛时，更能表示节点位置信息的特征向量将获得更大的权重，包含节点位置信息少的特征向量将获得较小的权重。权重采用两种初始化方式。分别是权重全部初始化为 1，以及符合标准正态分布的初始化。

在改进实验后，我们设计了两组对照实验，证明本文的改进是有效的。

模型内的结果对比

我们分别在教师模型 GCN，学生模型 MLP 和知识蒸馏后的 MLP 模型上进行单独的对照实验，比较每个模型在加入节点位置特征前和加入节点位置特征后的结果。验证加入节点位置特征对提高模型预测性能的有效性。

改进前和改进后的对比

我们针对不同训练集的不同训练模式，选取了模型改进前的最好结果和模型改进后的最好结果，以及没有加入节点位置特征时模型的结果进行对比，验证实验改进后的可行性。

4) **鲁棒性分析**

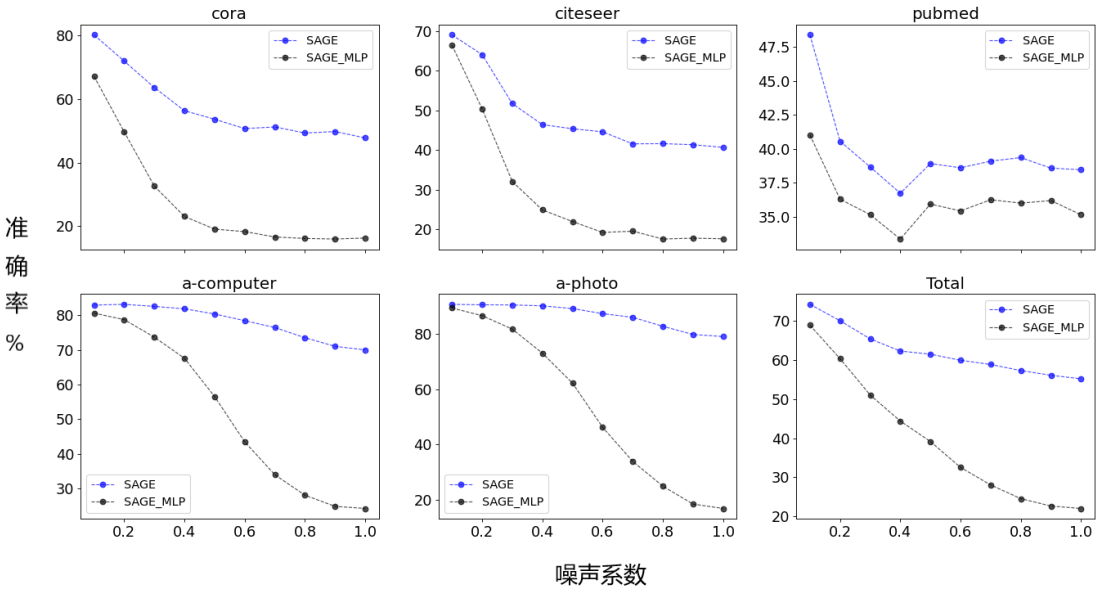

图 4.1 GNN 与知识蒸馏后的 MLP 鲁棒性对比

图4.1中，每个子图上面的标题为数据集名称，横坐标为噪声系数，取值范围属于 $[0, 1]$，系数越大，噪声干扰越多，纵坐标是模型在归纳式下，测试集的

准确率。

我们通过对输入数据增加不同程度的白噪声，观察模型对测试集的预测准确率，对模型的鲁棒性进行分析。如图4.1所示, 我们发现经过知识蒸馏后的MLP 在噪声的干扰下，性能下降严重，从五个数据集的平均情况看，经过知识蒸馏训练后的 MLP 性能在噪声系数提高到 1.0 的时候，模型预测准确率只有不到原 GNN 教师模型的 40%。与教师模型 GNN 相比，经过知识蒸馏后的 MLP 的鲁棒性较差。

我们使用 SAGE、GCN、GAT 和 APPNP 四个 GNN 模型，每次运行五次随机种子，记录了归纳式下测试集的准确率，分别在 Cora 数据集和 Pubmed 数据集上验证隐藏层蒸馏对模型鲁棒性的提升效果。此外，我们也对比了增加节点位置特征前后，模型在噪声干扰下的性能，验证节点位置特征对模型鲁棒性的积极作用。

5) **模型效率分析**

为了分析模型的效率，我们在 Pubmed 数据集上，对比了 SAGE 模型、GLNN 模型和本文提出的模型的推理时间和预测准确率，对模型的推理速度进行了分析。在实际部署中，模型对数据的推理是在线训练的，因此我们记录了模型在测试集上的推理时间，即在线训练时的推理时间对模型进行分析。

4.2 实验结果分析

在本节中，本文进行了对比实验，并对实验结果进行了进一步的分析和总结。

1) **初步验证的结果**

表4.2使用的教师模型是 GCN，学生模型是 MLP，None 表示通过 GCN 蒸

Dataset	None	SM+10	SM+20	SM+50	SM+100	SM+200	LM+10	LM+50	LM+100
Cora tran	78.69	79.25 ↑	77.47	76.35	**79.58** ↑	79.44 ↑	76.58	79.53 ↑	78.45
Cora ind	74.00	74.00	74.47 ↑	74.24 ↑	74.71 ↑	**75.41** ↑	74.47 ↑	73.77	73.07
Citeseer tran	71.33	72.65 ↑	71.05	70.83	73.20 ↑	72.10 ↑	69.94	72.60 ↑	**73.26** ↑
Citeseer ind	70.44	71.27 ↑	72.65 ↑	**73.20** ↑	72.10 ↑	72.93 ↑	71.55 ↑	71.27 ↑	71.27 ↑
Pubmed tran	80.19	80.08	79.83	80.26 ↑	**80.43** ↑	80.28 ↑	79.86	79.92	80.40 ↑
Pubmed ind	78.00	78.38 ↑	78.30 ↑	78.38 ↑	78.35 ↑	**78.48** ↑	78.02 ↑	78.20 ↑	77.97

表 4.2 初步验证的结果

馏的学生模型 MLP 在没有使用特征向量，也没有蒸馏隐藏层特征下的表现，SM+n 表示选取了 n 个最小特征值对应的特征向量下模型的表现，LM+n 表示选取了 n 个最大特征值对应特征向量下模型的表现。对每个数据集，我们

分别在 tran 模式和 ind 模式下进行了实验。其中 ind 模式下的结果采用的是模型在 V_{ind}^{U} 数据下的预测准确率，tran 模式下的结果采用的是模型在 V_{obs}^{U} 的预测准确率。

从表 4.2 中可以看到，不同数目的特征向量，对模型的影响效果不一样。在上面的特征向量选取中，大部分对特征向量的选取会引导模型表现更好。其中在 Cora 数据集在 tran 模式下，最多可以提高 0.89%，在 ind 模式下，最多可以提高 1.41%。Citeseer 数据集在 tran 模式下，最多可以提高 1.93%，在 ind 模式下，最多可以提高 2.76%。Pubmed 数据集在 tran 模式下最多可以提高 0.24%，在 ind 模式下，最多可以提高 0.48%。

2) **改进实验的结果**

模型内的结果对比

表4.3取两种权重初始化方式中的最好结果作为模型加入位置特征 Pos200 后的结果。ind 模式下的结果采用的是模型在 V_{ind}^{U} 数据下的预测准确率，tran 模式下的结果采用的是模型在 V_{obs}^{U} 的预测准确率。

在这次对比实验中，我们使用了五个数据集（Cora，Citeseer，Pubmed，A-computer 和 A-photo）。表4.3中，在加入位置特征后，模型在大部分数据集的表现得到了改善。对于 MLP 和经过知识蒸馏后的 MLP，在五个数据集中的性能均得到了提升。其中在 Pubmed 数据集的两种模式中 MLP 预测的准确率提升了 3.6% 以上。经过知识蒸馏后的 MLP，在 Cora、Citeseer、Pubmed 和 A-photo 数据集的两种模式性能均得到了改善，其中在 tran 模式下，模型性能在数据集 Cora 上提升最明显，准确率提升了 1.12%，在 ind 模式下，模型

Dataset	GCN		KD-MLP		MLP	
	None	Pos200	None	Pos200	None	Pos200
Cora tran	83.00	83.04 ↑	78.69	79.81 ↑	57.61	57.33
Cora ind	83.14	82.44	74.00	74.94 ↑	58.31	58.55 ↑
Citeseer tran	70.99	73.59 ↑	71.33	72.10 ↑	55.03	56.41 ↑
Citeseer ind	72.65	71.55	70.44	71.55 ↑	55.52	56.63 ↑
Pubmed tran	78.23	77.97	80.19	80.38 ↑	67.85	71.52 ↑
Pubmed ind	77.08	78.46 ↑	78.00	79.56 ↑	67.19	70.94 ↑
A-computer tran	80.47	84.04 ↑	91.77	84.02	65.62	68.05 ↑
A-computer ind	83.81	82.07	80.43	79.04	66.38	68.91 ↑
A-photo tran	91.69	91.41	91.77	92.83 ↑	78.81	80.34 ↑
A-photo ind	90.83	91.46 ↑	89.84	91.04 ↑	78.26	80.73 ↑

表 4.3 模型内的结果

性能在 A-photo 数据集下提升最明显，准确率提升了 1.56%。对于模型 GCN，模型的性能提升效果不如 MLP 和经过知识蒸馏后的 MLP，我们认为原因是，作为图神经网络的 GCN，本身具有提取图的拓扑信息的能力，而这正是 MLP 模型所欠缺的，加入节点位置信息后，能帮助 MLP 获取部分拓扑信息，因此对 MLP 模型性能的提升效果大于图神经网络 GCN。

改进前后实验对比

针对初步验证实验中不同特征向量的选取，此处选择性能最好的结果和改

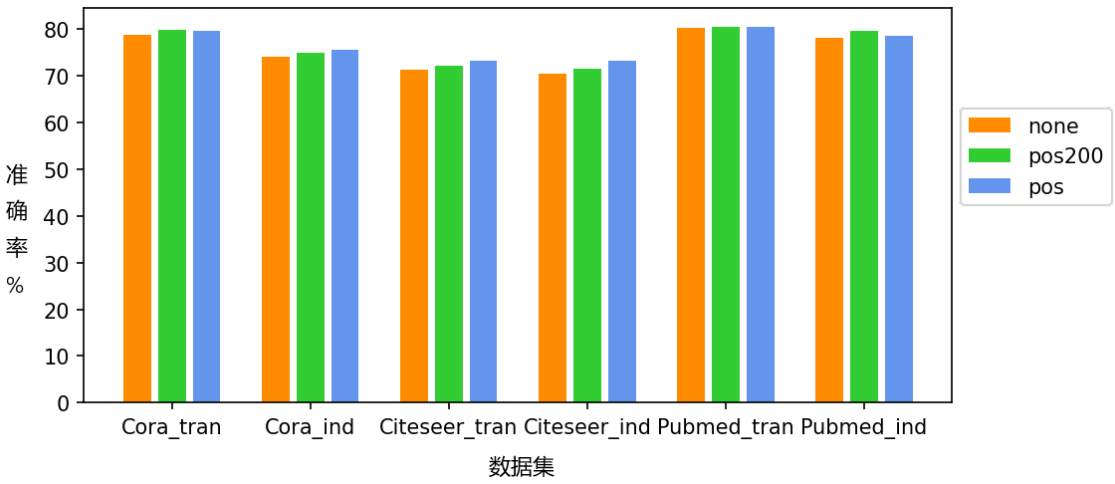

图 4.2 改进前后实验对比的结果

进后的结果进行对比。改进实验中，对不同的权重初始化方式，采取两种初始化方式中结果最好的进行比较和分析。

从图4.2可以看到，实验改善后，在没有人为选择特征向量数后，对所有数据集，模型表现均优于没有添加节点位置特征进行训练的模型的表现，说明权重参数的自适应学习是成功的。另外，在数据集 Cora 的 tran 模式和数据集 Pubmed 的 ind 模式下，改进后的模型性能优于改进前的模型性能，结果准确率分别比改进前的模型高 0.23% 和 1.08%。

3) **鲁棒性分析的结果**

Cora 数据集

表4.4中记录了 Cora 数据集在四个 GNN 模型下，五次随机种子结果的均值和方差。对于隐藏层特征的蒸馏，我们只选取度数最大前的 20% 的节点进行蒸馏。

可以看到，无论是对 GNN 模型的隐藏层进行蒸馏还是添加节点的位置特征，都能在一定程度上增强模型的鲁棒性。对于教师模型 SAGE、GCN 和 GAT，蒸馏其隐藏层特征或加入节点位置特征后，即使在噪声干扰严重（0.6，0.8，

Model	none	noise 0.1	noise 0.2	noise 0.4	noise 0.6	noise 0.8	noise 1.0
SAGE	80.87±1.67	79.04±1.93	72.55± 2.09	59.93 ± 2.61	55.15 ± 1.54	55.01 ± 2.77	52.63 ± 1.79
SAGE-MLP	73.07 ± 0.94	65.71±2.16	47.78±1.90	21.92±1.17	16.44±1.18	15.55±0.75	16.21±2.16
SAGE-hidden-MLP	72.88±1.28	65.67±1.21	**49.51 ±1.27**	**22.25±1.35**	**17.56±1.29**	**16.39±0.71**	**16.63±1.734**
SAGE-pos-MLP	72.51±2.00	**65.71±2.13**	47.45±1.43	**22.58±0.94**	**18.13±0.97**	**15.60±0.79**	**16.25±2.14**
SAGE-hidden-pos-MLP	72.56±1.34	**66.09±1.61**	**48.34±1.82**	**23.33±1.19**	**16.58±1.16**	**16.25±1.13**	**16.25±3.37**
GCN	81.59±1.95	80.05±1.53	72.69±3.37	56.07±1.81	50.87±1.32	48.62±1.30	47.87±1.47
GCN-MLP	73.35±0.87	66.56±1.12	47.40±3.36	22.95±2.37	17.10±0.85	16.11±1.06	16.81±1.61
GCN-hidden-MLP	**74.47±1.22**	**67.07±0.53**	**48.57±3.17**	22.62±2.65	**18.74±1.68**	**16.72 ±1.24**	**17.38±0.75**
GCN-pos-MLP	**74.05±0.62**	**66.89±1.23**	46.70±2.52	22.53±1.45	**17.42±1.66**	**17.75±1.34**	**16.81±1.30**
GCN-hidden-pos-MLP	**73.91±0.97**	65.85±1.67	**48.43±2.44**	**23.51±2.52**	**17.28±0.72**	**16.72±1.72**	**17.28±0.91**
GAT	83.51±1.17	81.45±1.98	74.99±3.48	64.73±1.38	60.52±0.78	57.61±3.91	56.91±1.32
GAT-MLP	73.35±0.87	66.56±1.12	47.40±3.36	22.95±2.37	17.10±0.85	16.11±1.06	16.81±1.61
GAT-hidden-MLP	**74.47±1.22**	**67.07±0.53**	**48.57±3.17**	22.62±2.65	**18.74±1.68**	**16.72±1.24**	**17.38±0.75**
GAT-pos-MLP	**73.91 ±1.43**	65.25±1.46	47.12±1.58	**23.14±1.68**	**18.50±1.18**	**16.63±2.08**	**16.96±1.25**
GAT-hidden-pos-MLP	**74.24±1.77**	65.57±1.07	**47.68±2.52**	**23.42±0.97**	**18.92±1.44**	**17.52±1.25**	**17.94±0.60**
APPNP	83.42±0.80	82.06±1.76	79.81±2.50	69.32±2.01	64.26±1.57	65.29±1.81	63.28±0.88
APPNP-MLP	72.79±1.56	65.90±2.01	50.59±2.32	23.61±1.40	17.33±1.73	16.63±1.42	17.28±1.14
APPNP-hidden-MLP	**73.35±1.34**	**66.56±1.14**	**51.15±2.62**	**25.15±1.51**	**17.75±1.72**	16.11±1.48	16.86±1.77
APPNP-pos-MLP	**74.00±1.60**	**66.84±1.08**	50.12±1.95	**23.61±2.79**	**17.75±2.99**	**17.28±0.98**	17.10±1.60
APPNP-hidden-pos-MLP	**73.86±0.87**	**66.93±1.70**	**50.63±1.79**	23.23±2.28	**18.08±3.01**	16.35±1.38	16.96±1.30

表 4.4 Cora 数据集鲁棒性分析的结果

1.0）的情况下，其性能也比改进前的模型好。对教师模型 APPNP，增加节点位置特征使学生模型 MLP 在大多数噪声的干扰下表现更优，对隐藏层特征进行蒸馏后，在噪声干扰不强的情况下，模型性能普遍得到提升，最优可以提高 1.54%。

Pubmed 数据集

图4.3是 Pubmed 数据集在四个 GNN 模型下取平均的结果，其中横坐标表示

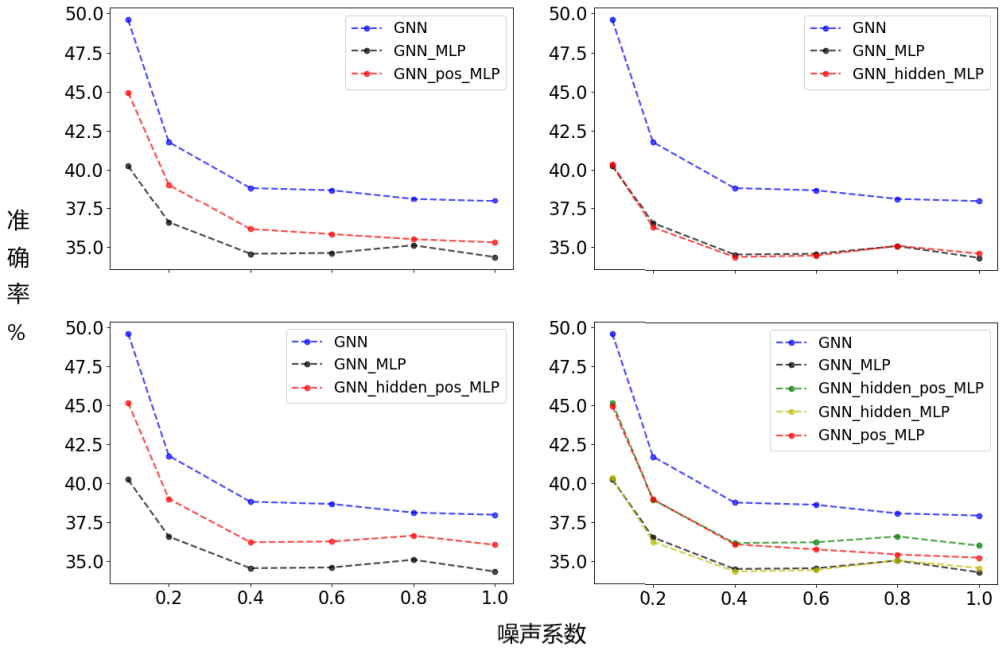

图 4.3 Pubmed 数据集鲁棒性分析的结果

噪声系数，纵坐标表示归纳式下模型在测试集上的预测结果的准确率。

可以看到对于 Pubmed 数据集，对节点位置特征进行提取，总体上能够给蒸

馏后的 MLP 性能带来提升，随着噪声系数的增大，提升效果减小，其中在噪声点 0.8 处，提升最为轻微，但总体上，模型在每个噪声点处的性能都得到提高；对隐藏层特征进行蒸馏，总体上给蒸馏后的 MLP 带来的性能提升效果不大，在噪声系数较大（0.8，1.0）的点处，能带来轻微提升；对同时提取节点位置特征和蒸馏隐藏层特征，模型的性能在每个噪声点处都提升显著且平稳。对此，我们认为在提取节点位置信息的时候，对 GNN 模型的隐藏层特征进行蒸馏能使 MLP 的性能提升在保持显著的同时更平稳。

隐藏层蒸馏对模型鲁棒性的影响分析

表4.5为 Cora 数据集在四个 GNN 模型下，蒸馏了隐藏层特征的 GNN-hidden-

Model	none	noise 0.1	noise 0.2	noise 0.4	noise 0.6	noise 0.8	noise 1.0	mean
Δ SAGE-hidden-MLP	-0.19	-0.04	1.73	0.33	1.12	0.84	0.42	0.60
Δ GCN-hidden-MLP	1.12	0.51	1.17	-0.33	1.64	0.61	0.57	0.76
Δ GAT-hidden-MLP	1.12	0.51	1.17	-0.33	1.64	0.61	0.57	0.76
Δ APPNP-hidden-MLP	0.56	0.66	0.56	1.54	0.42	-0.52	-0.42	0.4

表 4.5 Cora 数据集上蒸馏隐藏层特征后模型的鲁棒性分析

MLP 与原始的没有蒸馏隐藏层特征的学生模型 GNN-MLP 的表现对比。其中 $-a$ 表示 GNN-hidden-MLP 的预测准确率比 GNN-MLP 的性能降低了 $a\%$，b 表示 GNN-hidden-MLP 的预测准确率比 GNN-MLP 提高了 $b\%$。

从表4.5中,我们可以发现对不同的 GNN 教师模型,在大多数噪声点下,GNN-hidden-MLP 表现比 GNN-MLP 模型优秀，对不同 GNN 教师模型，模型预测准确率最好均可以提高 1.12 + %。而只在少数的噪声点处，表现稍差。对四个不同的 GNN 模型，GNN-hidden-MLP 的表现平均而言比 GNN-MLP 好，GNN-hidden-MLP 模型在测试集上的预测准确率比 GNN-MLP 模型的预测准确率可以高 0.4% 到 0.76%。这说明在 Cora 数据集下，蒸馏隐藏层特征对提升模型的鲁棒性具有积极的推动作用。

图4.4为 Pubmed 数据集下，蒸馏了隐藏层特征后模型的表现。其中横坐标表示噪声系数，纵坐标表示归纳式下模型在测试集上预测结果的准确率。

从图4.4中，可以看到，在四个 GNN 教师模型下，蒸馏隐藏层特征只在个别噪声点处给 MLP 带来了性能的提升，但总体上，模型表现变化不大。

节点位置特征对模型鲁棒性的影响分析

表4.6为 Cora 数据集在四个 GNN 教师模型下，使用了节点位置信息的模型 GNN-pos-MLP 与未使用节点位置信息的模型 GNN-MLP 的实验结果进行对

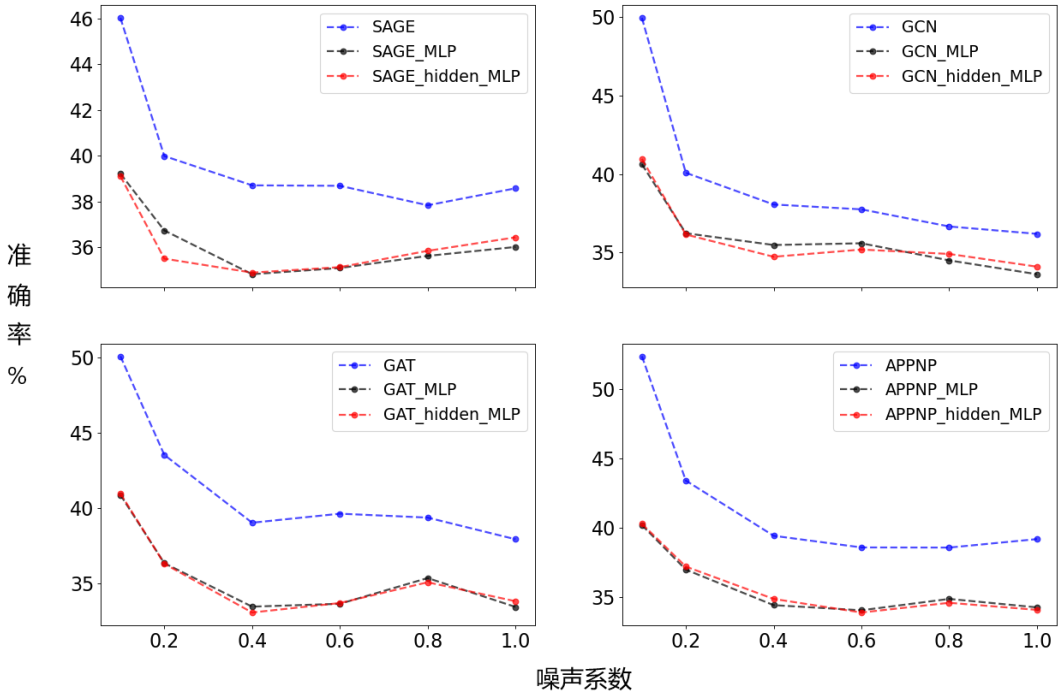

图 4.4 Pubmed 数据集上蒸馏隐藏层特征后模型的鲁棒性分析

Model	none	noise 0.1	noise 0.2	noise 0.4	noise 0.6	noise 0.8	noise 1.0	mean
Δ SAGE-pos-MLP	-0.56	0	-0.33	0.66	1.69	0.05	0.04	0.22
Δ GCN-pos-MLP	0.7	0.33	-0.7	-0.42	0.32	1.64	0	0.27
Δ GAT-pos-MLP	0.56	-1.31	-0.28	0.19	1.4	0.52	0.15	0.18
Δ APPNP-pos-MLP	1.21	0.94	-0.47	0	0.42	0.65	-0.18	0.37

表 4.6 Cora 数据集上加入节点位置信息后模型的鲁棒性分析

比。其中 $-a$ 表示 GNN-pos-MLP 的预测准确率比 GNN-MLP 的性能降低了 $a\%$，b 表示 GNN-pos-MLP 的预测准确率比 GNN-MLP 提高了 $b\%$。

如表4.6所示，可以看到在大部分噪声点处，GNN-pos-MLP 模型的预测准确率有所提升，在小部分噪声点处，模型的性能有所下降，下降幅度与提升幅度相当。但总体而言，GNN-pos-MLP 的性能在不同噪声点的平均性能是有所提升的，但是提升幅度没有模型 GNN-hidden-MLP 大。因此，在 Cora 数据集上，节点位置特征对提升模型的鲁棒性也是具有积极的推动作用的，但是作用没有隐藏层蒸馏大。

图4.5为在 Pubmed 数据集上，加入节点位置特征后模型的表现。其中横坐标表示噪声系数，纵坐标表示归纳式下模型在测试集上预测结果的准确率。

从图4.5中，可以看到对四个教师模型，加入节点位置信息后，GNN-pos-MLP 的表现均比 GNN-MLP 好。其中在 SAGE、GAT 和 APPNP 三个教师模型中，在噪声点小时，GNN-pos-MLP 的性能提升更为明显，但随着噪声点增

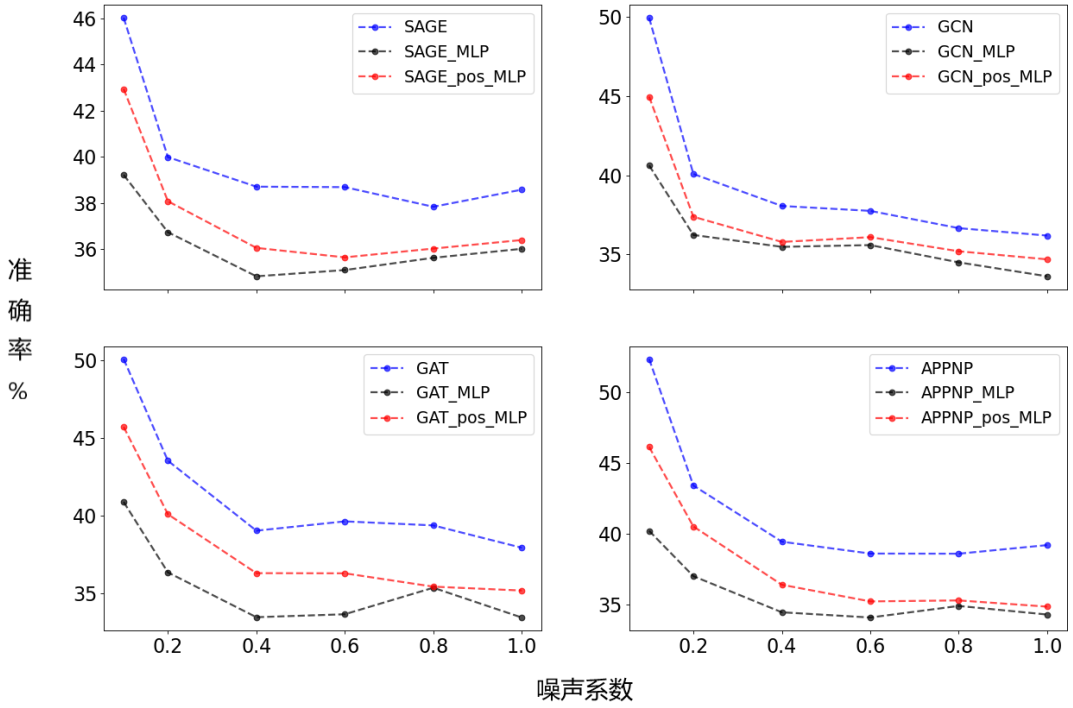

图 4.5 Pubmed 数据集上加入节点位置信息后模型的鲁棒性分析

大，GNN-pos-MLP 的性能提升幅度减小，如 APPNP 教师模型中，在噪声系数为 0.1 时，GNN-pos-MLP 预测准确率比 GNN-MLP 提升了 5.93%，但在噪声系数为 1.0 时，GNN-pos-MLP 的预测准确率仅比 GNN-MLP 提升了 0.56%。说明加入节点位置信息后，确实能对 GNN-MLP 的性能带来大幅度的提升，但是随着噪声系数的增加，提升的幅度并不能保持稳定。

本模型整体的鲁棒性分析

表4.7为 Cora 数据集在四个 GNN 教师模型下，增加了隐藏层蒸馏与节点位置信息的完整的模型 GNN-hidden-pos-MLP 与 GNN-MLP 的实验结果进行对比。其中 $-a$ 表示 GNN-hidden-pos-MLP 的预测准确率比 GNN-MLP 的性能降低了 $a\%$，b 表示 GNN-pos-MLP 的预测准确率比 GNN-MLP 提高了 $b\%$。

如表4.7所示，可以看见，除了 APPNP-hidden-pos-MLP 模型外，其他 GNN-

Model	none	noise 0.1	noise 0.2	noise 0.4	noise 0.6	noise 0.8	noise 1.0	mean
Δ SAGE-hidden-pos-MLP	-0.51	0.38	0.56	1.41	0.14	0.7	0.04	0.39
Δ GCN-hidden-pos-MLP	0.56	-0.71	1.03	0.56	0.18	0.61	0.47	0.39
Δ GAT-hidden-pos-MLP	0.89	-0.99	0.28	0.47	1.82	1.41	1.13	0.72
Δ APPNP-hidden-pos-MLP	1.07	1.03	0.04	-0.38	0.75	-0.28	-0.32	0.27

表 4.7 Cora 数据集上模型整体的鲁棒性分析

hidden-pos-MLP 模型在噪声系数增加的情况下，表现均比 GNN-MLP 好。但

APPNP-hidden-pos-MLP 模型在噪声系数小的情况下，也能获得性能的提升，且提升幅度高达 1.0±%，而在噪声系数大的情况下，模型性能有所下降，但是下降幅度不大，保持在 0.38% 内。另外，GNN-hidden-pos-MLP 在所有噪声点处的平均表现也有所提升，表现最好的模型为 GAT-hidden-pos-MLP，性能提升幅度高达 0.72%。这表明，在 Cora 数据集上，我们的模型能够在一定程度上提升传统的 GNN-MLP 模型的鲁棒性。

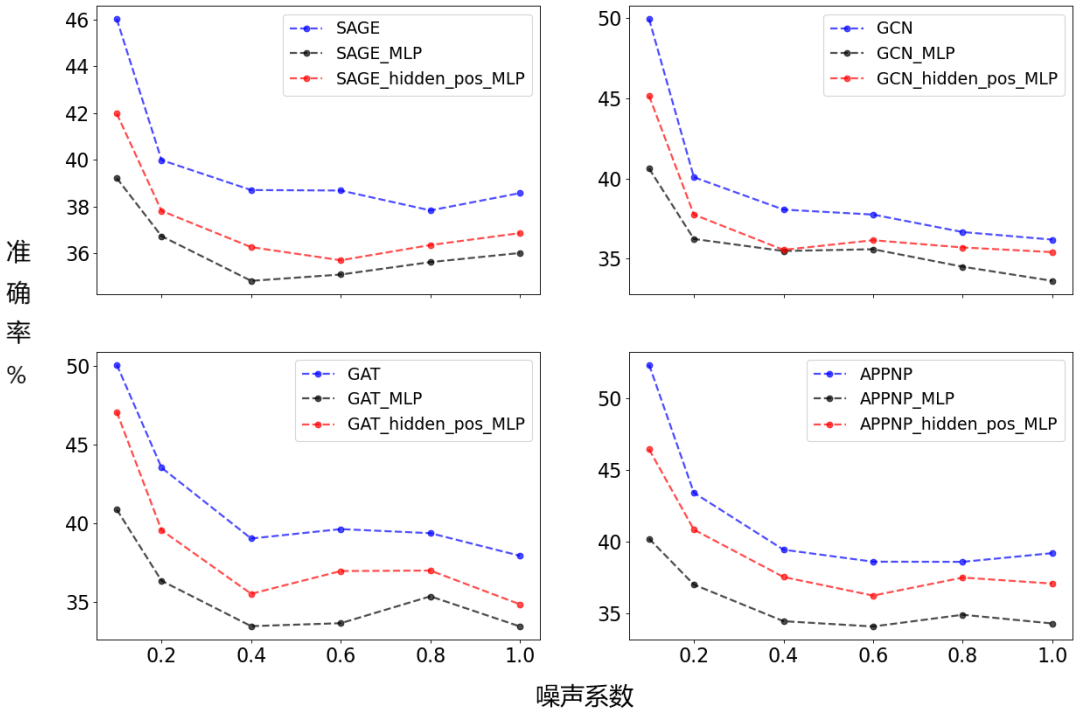

图 4.6 Pubmed 数据集上模型整体的鲁棒性分析

图4.6为在 Pubmed 数据集上，同时加入节点位置特征和蒸馏隐藏层特征后模型的表现。其中横坐标表示噪声系数，纵坐标表示归纳式下模型在测试集上预测结果的准确率。

从图4.6中，可以看到对四个教师模型，同时加入节点位置信息和蒸馏隐藏层特征后，GNN-hidden-pos-MLP 的性能相比 GNN-MLP 得到大幅度的提升，并且随着噪声系数的增加，提升的幅度不会发生下降的趋势。从教师模型 APPNP 中可以看到，APPNP-hidden-pos-MLP 的性能已经更接近于 APPNP 模型的性能。由此说明在 Pubmed 数据集上，同时加入节点位置信息和蒸馏隐藏层特征能显著提高模型的性能，并使模型性能的提升幅度在噪声系数增大的情况下，不会出现急剧下降的情况。

4) **模型效率分析的结果**

图4.7中记录了层数分别为 1，2，3 的 SAGE 模型、GLNN 模型以及本文提

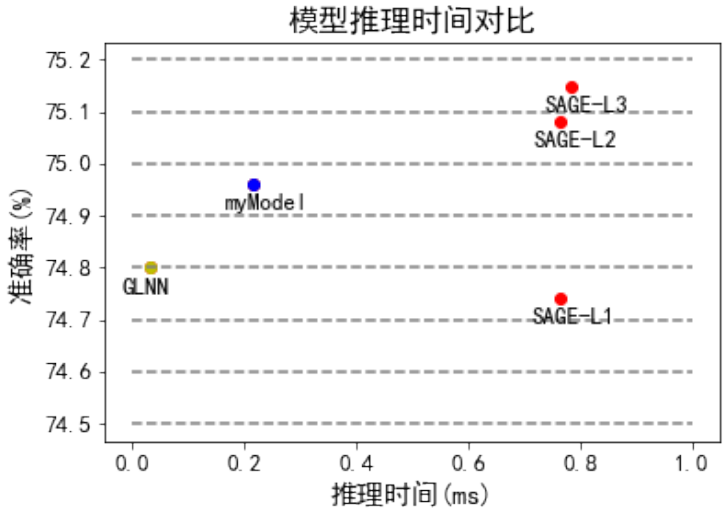

图 4.7 模型的效率分析

出的模型的表现，其中横纵标为模型的推理时间，单位为 ms，纵坐标为模型预测的准确率，单位为 %。

从图4.7中可以看到，本文模型的推理时间明显比 SAGE 模型的推理时间短，预测准确率接近于 2 层 SAGE 模型，远超 1 层 SAGE 模型。与 GLNN 模型相比，本文模型的推理时间稍慢，但是预测准确率较 GLNN 模型的预测准确率高 1.6%。可以看出本文模型在保证了模型快速的推理时间的同时，提高了模型预测准确率。

4.3 本章小结

在本章中，我们在多个不同的数据集上进行加入位置特征后的图神经网络知识蒸馏实验。通过加入节点位置特征前后模型性能的对比，我们验证了节点位置特征能够帮助 MLP 捕捉节点的位置信息，提升 MLP 在图知识蒸馏上的性能。通过对比手动选择节点位置特征向量和本文提出的用权重自适应学习的方法提取节点位置特征的实验结果，我们验证了本文提出的用权重自适应学习方法提取节点位置特征的有效性和合理性。

除此之外，我们发现经过知识蒸馏后的 MLP 鲁棒性差，针对此问题，我们对我们模型的鲁棒性进行了系统的分析，通过与原来 GNN-MLP 模型的性能进行对比，我们发现在 Cora 数据集上对隐藏层特征进行蒸馏能够帮助模型的鲁棒性得到明显的提升，对节点位置特征进行提取，能轻微提升模型的鲁棒性，添加隐藏层

蒸馏和节点位置特征后，整体模型性能的鲁棒性得到一定程度的改善。在 Pubmed 数据集上，对隐藏层特征进行蒸馏对提高模型的鲁棒性帮助不大，对节点位置特征进行提取能有效提高模型的鲁棒性，但随着噪声系数的增加，提升效果越来越不显著，同时添加隐藏层蒸馏和节点位置特征后，整体模型性能的鲁棒性得到显著提高，且不会出现噪声系数增加的情况下，提升效果下降的现象，即在噪声系数增加的情况下，整体模型性能仍然能得到显著的提高。由此，我们可以得出同时添加隐藏层蒸馏和节点位置特征后，我们模型的鲁棒性是有所提高的。

最后，我们分析了模型的运行效率，证明了模型能够保持比 GNN 高的推理速度，取得接近 GNN 模型的性能。

5 总结与展望

本章将对本文的工作进行简单的总结，并针对其中的不足，提出未来可以改进的方向。

5.1 总结

图神经网络因其鲁棒性和在各种图分析应用中的优异性能而得到了广泛的关注。然而，GNNs 的有效性严重依赖于足够的数据标签和复杂的网络模型，前者的获取具有挑战性，后者需要昂贵的计算资源。为了解决 GNNs 的标记数据稀缺性和高复杂性问题，引入了知识蒸馏来增强现有的 GNNs。该技术是在保持预测性能的同时，将庞大的教师模型的软标签转移到轻型的学生模型中。

关注到 MLP 的推理速度快，且推理过程不依赖于图结构，但是其无法很好捕捉图的拓扑信息，这与 GNNs 的功能是互补的，因此有学者提出将 GNNs 的知识蒸馏到 MLP 中，本文针对现有的 MLP 在图领域的知识蒸馏方法的不足，提出了相应的改善方案。

本文探究了 MLP 在图领域的知识蒸馏中性能受到限制的原因是 MLP 无法捕捉节点的位置信息，对教师模型 GNN 的软标签的硬匹配并不能从根本上让 MLP 具有捕捉图拓扑信息的能力。基于此原因，本文提出，提取节点位置特征，并将其与节点的内容特征进行拼接，作为 MLP 在图领域的知识蒸馏中的输入特征。

对节点位置特征的提取，本文提出了一个权重自适应学习的方法，为每个特征向量分配一个权重，使模型性能达到最优。本文进行了两组对照实验，我们验证了本文提出的方法的有效性。

通过提取多种不同位置特征的向量组合作为节点位置特征的表征，对比加入节点位置特征前面的模型表现，我们验证了节点位置特征对提升模型性能的积极作用。

通过对比手动提取特征向量和权重自适应学习方法提取特征向量对模型性能的提升效果，我们验证了本文提出的权重自适应学习方法的可行性。

另外，我们发现了经过知识蒸馏后的 MLP 的鲁棒性远差于教师模型 GNN。为了让学生模型 MLP 能更好地学习教师模型 GNN 内部的表征能力，提高学生模型的鲁棒性，我们提出将教师模型 GNN 的隐藏层特征蒸馏给学生模型 MLP。

通过对比 GLNN 模型的性能，我们发现对隐藏层特征进行蒸馏对提高学生模

型 MLP 的鲁棒性具有较大的帮助，对节点位置特征的提取对提高模型的鲁棒性有轻微帮助，整体模型的鲁棒性优于改进前的模型的鲁棒性。

通过对比 SAGE 模型、GLNN 模型和本文模型的预测准确率和推理时间，我们验证了本文模型在保持较高推理速度的同时，测准确率较 GLNN 有所提高，模型鲁棒性更是有所增强。

5.2 未来工作展望

本文提出了一种改善 MLP 在图领域的知识蒸馏性能受到限制的方法，实验效果总体上达到了预期目标。但是在一些方面仍有改善空间。在未来的工作中，我们希望能将以下几点不足进行改善。

1) 对于验证节点位置特征对改善图领域知识蒸馏后的 MLP 的性能的有效性实验中，我们没有采取多个图神经网络模型对该方法进行验证，在未来的工作中可以完善相关实验。
2) 对节点位置特征的提取，我们的最终版本是对所有数据集都提取拉普拉斯矩阵最大的 100 个特征值对应的特征向量和最小的 100 个特征值对应的特征向量，这种方法的缺点是没有针对不同大小的数据集进行不同数量的特征向量的提取，在 A-computer 数据集上，节点位置特征的提取并没有使模型准确率得到提高，这是由于该数据集的节点数和边数均比较多，只提取 200 个特征向量对它来说很可能是不够的。因此更好的方案应该是根据数据集大小自适应调整需要提取的特征向量数。
3) 本文对于权重的初始化采取了正态分布初始化和全 1 初始化两种方式，但未深入研究该参数初始化的最佳选取标准，亦没有深入分析在模型可以自适应学习下，不同初始化方式导致模型性能不一样的原因。因此在后续的工作，我们可以考虑在这方面进行深入的探讨和研究。
4) 在以上实验中,我们只在较小的数据集上进行,没有拓展到大的 OPEN GRAPH BENCHMARK (OGB) 数据集[40]，现实生活中的应用，数据量要比我们的数据集大得多，因此在更大的数据集上验证我们模型的效果是我们后期有待完成的工作。

以上是本文的不足和缺陷，我们希望在未来的研究中可以完善这些不足。

参考文献

[1] 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.

[2] ZHANG S, LIU Y, SUN Y, et al. Graph-less neural networks: Teaching old mlps new tricks via distillation[A]. 2021.

[3] XU Y, ZHANG Y, GUO W, et al. Graphsail: Graph structure aware incremental learning for recommender systems[C]//Proceedings of the 29th ACM International Conference on Information & Knowledge Management. 2020: 2861-2868.

[4] HINTON G, VINYALS O, DEAN J. Distilling the knowledge in a neural network[A]. 2015.

[5] CHEN Y H, EMER J, SZE V. Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks[J]. ACM SIGARCH computer architecture news, 2016, 44(3): 367-379.

[6] JUDD P, ALBERICIO J, HETHERINGTON T, et al. Proteus: Exploiting numerical precision variability in deep neural networks[C]//Proceedings of the 2016 International Conference on Supercomputing. 2016: 1-12.

[7] HAN S, POOL J, TRAN J, et al. Learning both weights and connections for efficient neural networks[C]//NIPS'15: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1. Cambridge, MA, USA: MIT Press, 2015: 1135–1143.

[8] GUPTA S, AGRAWAL A, GOPALAKRISHNAN K, et al. Deep learning with limited numerical precision[C]//International conference on machine learning. PMLR, 2015: 1737-1746.

[9] HU Y, YOU H, WANG Z, et al. Graph-mlp: Node classification without message passing in graph[A]. 2021.

[10] YANG Y, QIU J, SONG M, et al. Distilling knowledge from graph convolutional networks[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 7074-7083.

[11] YAN B, WANG C, GUO G, et al. Tinygnn: Learning efficient graph neural networks[C]// Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2020: 1848-1856.

[12] YANG C, LIU J, SHI C. Extract the knowledge of graph neural networks and go beyond it: An effective knowledge distillation framework[C]//Proceedings of the web conference 2021. 2021: 1227-1237.

[13] DWIVEDI V P, LUU A T, LAURENT T, et al. Graph neural networks with learnable structural and positional representations[A]. 2021.

[14] TIAN Y, ZHANG C, GUO Z, et al. Learning MLPs on graphs: A unified view of effectiveness, robustness, and efficiency[C/OL]//The Eleventh International Conference on Learning Repre-

sentations. 2023: 1-12. https://openreview.net/forum?id=Cs3r5KLdoj.

[15] PEROZZI B, AL-RFOU R, SKIENA S. Deepwalk: Online learning of social representations [C]//Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. 2014: 701-710.

[16] PAN S J, YANG Q. A survey on transfer learning[J]. IEEE Transactions on knowledge and data engineering, 2010, 22(10): 1345-1359.

[17] 黄震华, 杨顺志, 林威, 等. 知识蒸馏研究综述[J]. 计算机学报, 2022, 45(3): 624-653.

[18] GOTMARE A, KESKAR N S, XIONG C, et al. A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation[A]. 2018.

[19] ROMERO A, BALLAS N, KAHOU S E, et al. Fitnets: Hints for thin deep nets[A]. 2014.

[20] YIM J, JOO D, BAE J, et al. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 4133-4141.

[21] SRINIVAS S, FLEURET F. Knowledge transfer with jacobian matching[C]//International Conference on Machine Learning. PMLR, 2018: 4723-4731.

[22] LEE S H, KIM D H, SONG B C. Self-supervised knowledge distillation using singular value decomposition[C]//Proceedings of the European Conference on Computer Vision (ECCV). 2018: 335-350.

[23] GOODFELLOW I, POUGET-ABADIE J, MIRZA M, et al. Generative adversarial networks[J]. Communications of the ACM, 2020, 63(11): 139-144.

[24] TIAN Y, KRISHNAN D, ISOLA P. Contrastive representation distillation[A]. 2019.

[25] SPERDUTI A, STARITA A. Supervised neural networks for the classification of structures[J]. IEEE Transactions on Neural Networks, 1997, 8(3): 714-735.

[26] GORI M, MONFARDINI G, SCARSELLI F. A new model for learning in graph domains[C]// Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.: volume 2. IEEE, 2005: 729-734.

[27] SCARSELLI F, GORI M, TSOI A C, et al. The graph neural network model[J]. IEEE transactions on neural networks, 2008, 20(1): 61-80.

[28] GALLICCHIO C, MICHELI A. Graph echo state networks[C]//The 2010 international joint conference on neural networks (IJCNN). IEEE, 2010: 1-8.

[29] LECUN Y, BENGIO Y, et al. Convolutional networks for images, speech, and time series[J]. The handbook of brain theory and neural networks, 1995, 3361(10): 1995.

[30] KIPF T N, WELLING M. Semi-supervised classification with graph convolutional networks[A]. 2016.

[31] GILMER J, SCHOENHOLZ S S, RILEY P F, et al. Neural message passing for quantum chemistry[C]//International conference on machine learning. PMLR, 2017: 1263-1272.

[32] VELIČKOVIĆ P, CUCURULL G, CASANOVA A, et al. Graph attention networks[A]. 2017.

[33] GASTEIGER J, BOJCHEVSKI A, GÜNNEMANN S. Predict then propagate: Graph neural

networks meet personalized pagerank[A]. 2018.

[34] LI G, XIONG C, THABET A, et al. Deepergcn: All you need to train deeper gcns[A]. 2020.

[35] KHAMSI M A, KIRK W A. An introduction to metric spaces and fixed point theory[M]. John Wiley & Sons, 2011.

[36] ZHOU J, CUI G, HU S, et al. Graph neural networks: A review of methods and applications[J]. AI open, 2020, 1: 57-81.

[37] LIU J, ZHENG T, ZHANG G, et al. Graph-based knowledge distillation: A survey and experimental evaluation[A]. 2023.

[38] YANG Z, COHEN W, SALAKHUDINOV R. Revisiting semi-supervised learning with graph embeddings[C]//International conference on machine learning. PMLR, 2016: 40-48.

[39] SHCHUR O, MUMME M, BOJCHEVSKI A, et al. Pitfalls of graph neural network evaluation [A]. 2018.

[40] HU W, FEY M, ZITNIK M, et al. Open graph benchmark: Datasets for machine learning on graphs[J]. Advances in neural information processing systems, 2020, 33: 22118-22133.