学号_____20165004_____ 密级_____

东北大学本科毕业论文

基于深度图卷积网络的结点分类算法的研究与实现

学 院 名 称：软件学院

专 业 名 称：软件工程

学 生 姓 名：刘唐

指 导 教 师：张伟 副教授

黄增峰 副教授

2020 年 6 月

基于深度图卷积网络的结点分类算法的研究与实现

作者姓名： 刘唐

校内指导教师： 张伟 副教授

校外指导教师： 黄增峰 副教授

单位名称： 软件学院

专业名称： 软件工程

东 北 大 学

2020 年 6 月

Research and Implementation of Deep Graph Convolutional Networks on Node Classification

by Liu Tang

Supervisor:	Associate Professor	Zhang Wei
Associate Supervisor:	Associate Professor	Huang Zengfeng

Northeastern University

June 2020

郑 重 声 明

本人呈交的学位论文，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确的方式标明。本学位论文的知识产权归属于培养单位。

本人签名： 日期：

摘　要

图卷积神经网络是近年来深度学习领域新兴起的方向，在图的半监督结点分类等任务上表现突出。深度学习的成功在于深层网络架构，然而实验研究表明，随着模型层数增加，图卷积神经网络的性能会急剧下降。

本文首先分析了深度图卷积神经网络的研究现状，对这些模型所存在的主要问题进行了阐述。接着重点从三个方面展开了研究，分别是过拟合、梯度消失和过平滑。过拟合和梯度消失是传统深度神经网络面临的问题，而过平滑则是深度图卷积神经网络特有的问题。针对过拟合问题，在图卷积神经网络上引入了三种正则化算法，分别是权重衰减、提前终止和丢弃法。针对梯度消失问题，也在图卷积神经网络上引入了三种传统算法，分别是 Xavier 初始化、梯度修剪和批量归一化。实验研究表明，过拟合和梯度消失都不是限制图卷积神经网络加深的主要原因，已有的几种传统算法足以缓解这些问题，但是无法阻止随着层数增加模型性能的骤降。针对过平滑问题，本文不仅从理论角度进行了分析，也精心设计了实验对理论进行验证。不同于以往用 GCN 研究过平滑问题，本文采用 SGC 进行实验研究，避免了过拟合和梯度消失的干扰。本文基于理论分析和已有模型，从四个角度提出了缓解算法。从图数据预处理角度，本文基于结点相似度对 DropEdge 进行了改进。从控制邻居权重角度，本文基于余弦相似度对 GAT 进行了改进。从平衡局部全局角度，本文引入了残差连接和密集连接，并提出了带权重的残差连接。从增强自身特征角度，本文提出了一种新的网络结构，在每一层引入输入层的带权重跳接。实验研究表明，过平滑是限制图卷积神经网络加深的主要原因，本文提出的几种算法大大缓解了该问题，在多个数据集上取得了显著的效果。

本文系统地对图卷积神经网络无法加深这一问题进行了研究，并针对性地引入或提出了多种行之有效的算法。但是由于图神经网络的基准数据集规模比较小，因此实验结果对模型表现的区分性还不够。

关键词：深度图卷积神经网络；半监督结点分类任务；过拟合；梯度消失；过平滑

ABSTRACT

Graph convolutional networks are emerging directions in the field of deep learning in recent years.They are outstanding in tasks such as semi-supervised node classification of graphs. The success of deep learning lies in the deep network architecture. However, experimental research shows that as the number of model layers increases, the performance of graph convolutional networks will decrease sharply.

This paper first analyzes the research status of deep graph convolutional neural networks, and explains the main problems of these models. Then focus on research from three aspects, namely overfitting, vanishing gradient and oversmoothing. Overfitting and vanishing gradient are problems faced by traditional deep neural networks, while oversmoothing is a unique problem of deep graph convolutional networks. For the problem of overfitting, three regularization methods are introduced on the graph convolutional network, which are weight decay, early stopping and dropout. For the problem of vanishing gradient, three traditional methods are also introduced on graph convolutional neural networks, which are Xavier initialization, gradient clipping and batch normalization. Experimental studies have shown that neither overfitting nor vanishing gradient are the main reasons for limiting the depth of graph convolutional networks. There are several traditional methods that are sufficient to alleviate these problems, but they cannot prevent the model performance from dropping as the number of layer increases. Aiming at the problem of over-smoothness, we not only analyze from a theoretical perspective, but also carefully designs experiments to verify the theory. Different from previous research, we use SGC for experimental research to avoid the interference of overfitting and vanishing gradient. Based on theoretical analysis and existing models, we propose methods from four perspectives. From the perspective of graph data preprocessing, we improve DropEdge based on node similarity. From the perspective of controlling neighbor weights, we improve GAT based on cosine similarity. From the perspective of balancing local and global information, we introduce residual connections and dense connections, and propose weighted residual connections. From the perspective of enhancing its own characteristics, we propose a new network structure that introduces weighted jumpers from input layer to each layer. Experimental research shows that oversmoothing is the main reason for limiting the depth of graph convolutional networks.

Several methods proposed in this paper greatly alleviate this problem and have achieved significant results on multiple data sets.

This paper systematically studies the problem that graph convolutional networks cannot deepen, and introduces or proposes a variety of effective methods. However, because the scale of the benchmark data set of the graph neural network is relatively small, the experimental results are not sufficient to distinguish the model performance.

Key words: Deep GCN; Semi-supervised Node Classification; Overfitting; Vanishing Gradient; Oversmoothing

目 录

第 1 章 绪 论

本章首先介绍深度图卷积神经网络和图上的半监督结点分类的研究背景，接着分析近年来国内外研究现状，然后介绍本文的研究内容和主要贡献，最后给出该论文的组织结构。

1.1 研究背景

随着训练数据的大量增长和计算资源的快速发展，深度学习在语音识别、目标检测、自然语言处理等方面取得了巨大成功。这归功于深度学习能从欧式数据如语音、文本、图像等中提取有效的特征表示。但是越来越多的任务要求对非欧式数据，如引用网络、社交网络、蛋白质结构等图数据进行处理。然而由于图的不规则、异质性、大规模等特点，传统的神经网络 CNN、RNN 等无法胜任。近年来，研究人员相继提出了图递归神经网络 GRNNs、图卷积神经网络 GCNs、图自编码器 GAEs、图强化学习 GRL 等模型，在图数据处理上取得了优越的效果。

图分类、结点分类、链路预测是常见的图上的学习任务。其中结点分类一般指半监督结点分类任务：给定包含结点信息和结构信息的图数据集，带有标签的部分结点作为训练集，预测剩余结点的标签类别。有研究者运用近似技巧从谱图卷积推导出图卷积神经网络的逐层传播公式，使得图像处理中的卷积操作能够被简单应用到图结构数据处理中，在图的半监督结点分类任务上取得了不错的表现[1]。

深度学习的成功在于深层网络架构，该架构具有更高的模型复杂度，因此也具有更强的学习能力。此外，加深网络相比加宽网络具有逐层处理、特征变换等优点。在图像分类任务中，杰出的 ResNet 具有 152 层[2]。然而研究表明，随着层数增加，GCN 的性能会急剧下降。目前对该问题的研究还较少，为什么性能会下降，如何才能加深 GCN，是 GCN 发展面临的两个挑战[1]。

通过将关系数据自然地建模为图结构数据，GCN 等基于图的深度学习模型被广泛应用于其他学科，如计算机视觉、推荐系统、自然语言处理、疾病或药物预测、基于图的 NP 问题等。对如何加深 GCN 的研究，能够提升模型的性能，从而促进更深入地挖掘现有图数据的丰富价值。

1.2 研究现状

图神经网络是近年来新兴起的研究热点，对深度图卷积神经网络的研究也刚起步不久。

理论方面，研究者们揭示了 K 层 GCN 与 K 步随机游走的关系[3]；证明了图卷积是一种特殊形式的拉普拉斯平滑[4]；分析了在图同构测试任务上 GNN 性能的上限[5]。

模型方面，PPNP 将神经网络与传播算法分离，融入 Personalized PageRank 算法，在聚合时可以获取更大范围的邻居信息[6]；JK-Net 以层级聚合的方式自适应地融合不同层的信息，从而平衡不同邻域的结点的局部与全局信息[3]；Cluster-GCN 通过变换邻接矩阵并添加正则化，在考虑权重的同时强化邻近邻居结点的信息[7]；N-GCN 在不同尺度下进行图卷积操作，最后融合所有卷积结果得到结点的特征表示，通过对不同尺寸感受野的组合提高模型的表征能力[8]；RGCN 基于第 K 层捕获了 K-Hop 邻居结点信息，这些相邻层之间存在依赖关系，使用 RNN（GRU，LSTM）对层间的长期依赖建模[9]；DeepGCN 借鉴 CNN 的成功经验，基于梯度消失/爆炸的问题，引入残差连接、密集连接和空洞卷积，在点云语义分割任务上进行了实验[10]；Dropedge 在每轮训练中从图中随机删除一定比例的边，从数据增强角度缓解了过拟合，从减缓传播角度缓解了过平滑[11]；Snowball 和 Truncated Krylov 均利用了多尺度信息，在一定条件下两种网络结构是等价的，是谱图卷积和深度 GCN 在块 Krylov 空间下的推广形式[12]；PairNorm 通过引入正则化项改进目标函数，既保证了同一类簇的结点信息趋于一致，又促进了不同类簇的结点信息差异扩大[13]。

这些模型都增强了 GCN 的学习能力，但是各自也存在着一些不足之处，在几个引用数据集上性能提升有限，多数模型在超过两层后性能仍然会下降。其中，PPNP 的传播部分借鉴的是 Personalized PageRank，它是基于经验假设设计的，直接进行量化计算而不需要参数学习过程，但是也失去了神经网络的优势；JK-Net 只在最后一层对所有层进行融合，层之间的传播方式没有改变，较深层产生的输出仍然存在不同类簇间的结点混合问题，RGCN 虽然有一些改进，但是仍然存在相似的问题；Cluster-GCN 是基于矩阵的改进，当数据集庞大时，就会面临内存限制问题，而大数据集恰恰最需要更深的 GCN；N-GCN、Snowball 和 Truncated Krylov 的思路都类似于 Inception，从“宽度”上对网络进行拓展，在同一层级上运行多个不同尺寸的卷积核，但是大尺寸的卷积核不可避免地会引起过平滑问题；DeepGCN 的动机在于解决梯度消失/爆炸，但是它不是阻碍 GCN 加深的主要原因，同时，DeepGCN 只在点云数据集上进行了实验，该任务属于图层次的分类，每张图之间不连通，不存在过平滑问题；DropEdge 采用随机割边的算法，在稀疏连接图数据上有一定效果，在密集连接图数据上却有反作用；PairNorm 的正则化项扩大的是所有不相连结点对间的差异，总体来说对于缓解过平滑问题效果有限。

1.3 研究内容

文献方面，本文从理论和模型两个角度，对当前深度 GCN 研究的最新进展进行了总结归纳，确定了过平滑问题是限制 GCN 层数加深的主要原因。

算法方面，本文从三个角度展开了研究：

针对过拟合问题，在 GCN 上研究并实验了几种传统算法：权重衰减 weight decay、提前终止 Early Stopping 和丢弃法 Dropout。

针对梯度消失/爆炸问题，在 GCN 上研究并实验了几种传统算法：Xavier 初始化、梯度修剪和批量归一化 Batch Norm。

针对过平滑的问题，从图数据预处理的角度，在 DropEdge 的基础上做了两种改进，分别是基于结点度数的 DegreeDrop 和基于特征相似 DistanceDrop；从控制邻居权重的角度，利用特征的余弦相似度，经过 Softmax 归一化处理，作为结点与邻居间的权重；从平衡局部全局的角度，将残差连接引入 GCN 网络，并提出了带可学习权重的残差连接，同时进一步引入并尝试了密集连接；从增强结点自身的角度，在每层引入输入层的跳接，并在初始特征和当前特征间赋予平衡权重。

实验方面，本文额外引入了几个密集连接的图数据集，使得实验结果更能区分模型的学习能力；设计了对过平滑理论分析的验证实验，有力地佐证了过平滑是阻碍 GCN 加深的主要问题；规范了对过平滑问题的模型的实验设置，排除了过拟合和梯度消失/爆炸对实验的混合影响。

本文的创新之处主要体现在算法和实验方面，一方面基于理论分析从不同角度计了多个有效缓解过平滑问题的有效算法，另一方面通过改进和规范实验设置与流程，使得对深度 GCN 的实验研究更具有说明力。

1.4 组织结构

本文的主要研究内容为探索限制图卷积神经网络在结点分类任务上的深度的因素，并且提出相应的缓解算法。因此以深度图卷积神经网络面临的问题以及相应的缓解算法为导向进行论文的组织结构。

第 1 章 绪论。介绍深度图卷积神经网络和图上的半监督结点分类的研究背景，分析近年来国内外研究现状，阐述本文的研究内容和主要贡献，最后介绍该论文的组织结构。

第 2 章 相关工作。介绍图卷积神经网络的理论基础，以及图上的半监督结点分类任务的定义，介绍了几种主要的深度图卷积神经网络模型，分析了各自的优缺点以及本

文所做的改进。

第 3 章 实验规范。介绍实验中用到的 9 个开源数据集，并说明后几章通用的一些实验设置，如数据集划分、损失函数选择，评价指标选择等。

第 4 章 面向过拟合的算法。从理论角度分析了过拟合，接着引入了 3 种正则化算法：权重衰减、提前终止和丢弃法，最后在引用数据集上进行了实验。

第 5 章 面向梯度消失的算法。从理论角度分析了梯度消失，接着引入了 3 种传统算法：Xavier 初始化、梯度修剪和批量归一化，最后在引用数据集上进行了实验。

第 6 章 面向过平滑的算法。从理论角度分析了过平滑，接着设计了实验对理论进行验证，然后从四个角度提出了缓解算法：基于图数据预处理的算法、基于控制邻居权重的算法、基于平衡局部全局的算法和基于增强自身特征的算法，最后在 9 个数据集上进行了实验。

第 7 章 总结与展望。对本文进行了总结，并说明了下一步工作。

第 2 章 相关工作

图卷积神经网络是近年来深度学习领域新兴起的方向。本章首先介绍图卷积神经网络的理论基础，以及图上的半监督结点分类任务的定义，接着介绍了深度图卷积神经网络的主要模型，并分析了各自的优缺点以及本文所做的改进。

2.1 图卷积神经网络

图卷积神经网络 GCN 由 ChebNet 近似推导而来，是一种简单有效的层式传播模型，是深度图卷积神经网络的基础[1]。

2.1.1 问题定义

图上的半监督结点分类任务是指：给定包含结点信息和结构信息的图数据集，将带标签的部分结点作为训练集，预测剩余结点的标签类别。对该任务可采取的学习策略见公式（2.1）-（2.2）表述。

$$\mathcal{L} = \mathcal{L}_0 + \lambda \mathcal{L}_{reg} \tag{2.1}$$

$$\mathcal{L}_{reg} = \sum_{i,j} A_{ij} \left\| f(X_i) - f\left(X_j\right) \right\|^2 = f(X)^T \Delta f(X) \tag{2.2}$$

其中$\mathcal{L}_0$表示带标签的结点的监督损失，$\mathcal{L}_{reg}$表示图结构信息引入的损失，$f(X)$表示类似神经网络的可微分函数，λ是权重系数，X是结点特征矩阵，$\Delta = D - A$表示无向图$\mathcal{G} = (\mathcal{V}, \mathcal{E})$的拉普拉斯矩阵，$A$是邻接矩阵，D 是度矩阵，$D_{ii} = \sum_j A_{ij}$。

正则化项$\mathcal{L}_{reg}$基于相邻结点更加相似的假设，然而该假设可能会限制模型的能力。GCN 使用神经网络模型$f(X, A)$直接编码图结构信息，回避了损失函数中的正则化项$\mathcal{L}_{reg}$的使用。

2.1.2 谱图卷积

经过对称归一化后拉普拉斯矩阵为$L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} = U \Lambda U^T$，其中$U$是$L$的特征向量矩阵，$\Lambda$是$L$的特征值矩阵。给定输入信号$x \in R^N$，傅里叶域的滤波器$g_\theta = diag(\theta), \theta \in R^N$，谱图卷积见公式（2.3）表述。

$$g_\theta * x = U g_\theta U^T x \tag{2.3}$$

其中$U^T x$是对 x 做图上的傅里叶变换，g_θ可视作L的特征值的函数$g_\theta(\Lambda)$。然而，大图上L的特征分解很低效，特征矩阵乘法的时间复杂度也较高。这里可以用切比雪夫多项式$T_k(x)$近似$g_\theta(\Lambda)$，见公式（2.4）表述。

$$g_{\theta'}(\Lambda) \approx \sum_{k=0}^{K} \theta'_k T_k(\widetilde{\Lambda}) \tag{2.4}$$

这里$\widetilde{\Lambda} = \frac{2}{\lambda_{max}}\Lambda - I_N \cdot \lambda_{max}$是$L$的最大特征值，$\theta' \in R^k$是切比雪夫因子。切比雪夫多项式由递推公式$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$定义，其中$T_0(x) = 1, T_1(x) = x$。将切比雪夫近似公式代入谱图卷积公式，见公式（2.5）表述。其中$\tilde{L} = \frac{2}{\lambda_{max}}L - I_N$。

$$g_{\theta'} * x \approx \sum_{k=0}^{K} \theta'_k T_k\left(\tilde{\tilde{L}}\right)x \tag{2.5}$$

2.1.3 传播公式

在公式（2.5）中，令$K = 1$，谱图卷积近似为关于L的线性函数，我们可以堆叠多层获得卷积能力；令$\lambda_{max} \approx 2$，我们期望神经网络的参数在训练过程中自适应该变化。在这些条件下近似结果见公式（2.6）表述。

$$g_{\theta'} * x \approx \theta'_0 x + \theta'_1(L - I_N)x = \theta'_0 x - \theta'_1 D^{-\frac{1}{2}}AD^{-\frac{1}{2}}x \tag{2.6}$$

其中参数θ'_0和θ'_1可以共享于所有结点的计算。我们引入$\theta = \theta'_0 = -\theta'_1$进一步近似，见公式（2.7）表述。这在一定程度上可以缓解过拟合，并减少计算量。

$$g_{\theta'} * x \approx \theta\left(I_N + D^{-\frac{1}{2}}AD^{-\frac{1}{2}}\right)x \tag{2.7}$$

这里$I_N + D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$的特征值范围是[0, 2]，重复该操作会导致数值不稳定等问题。为此我们引入再正则化技巧$I_N + D^{-\frac{1}{2}}AD^{-\frac{1}{2}} \to \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$，其中$\tilde{A} = A + I_N, \tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$。

我们将公式（2.7）进一步泛化，给定输入信号$X \in R^{N \times C}$，X有 C 个通道（即 C 维特征），卷积包含 F 个滤波器，见公式（2.8）表述。

$$Z = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}X\theta \tag{2.8}$$

其中$\theta \in R^{C \times F}$是滤波器的参数矩阵，$Z \in R^{N \times F}$是卷积操作后的信号矩阵。在多层神经网络中，习惯上把变换后的X记做H，表示结点在隐藏层的嵌入。此外，习惯上把神经网络的参数记做W而非θ，当前层的输出Z会作为下一层输入。经过这些符号替换后，得到 GCN 逐层传播公式，见公式（2.9）表述。

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) \tag{2.9}$$

2.1.4 结点分类

经过 GCN 处理后的输出特征可以用于下游任务，两层的用于半监督结点分类任务的 GCN 见公式（2.10）表述。其中$\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$。

$$Z = f(X, A) = softmax(\hat{A}\, ReLU\left(\hat{A}XW^{(0)}\right)W^{(1)} \tag{2.10}$$

这里$W^{(0)} \in R^{H \times C}$是权重矩阵，在输入层和隐藏层间做线性变换，$W^{(1)} \in R^{H \times F}$也是

权重矩阵，在隐藏层和输出层间做线性变换。变换结果经过 softmax 激活函数输出作为分类结果。对于半监督结点分类任务，在带标签的样本上评估交叉熵，见公式（2.11）表述。

$$\mathcal{L} = -\sum_{l\in y_L}\sum_{f=1}^{F} Y_{lf}\ ln\ Z_{lf} \tag{2.11}$$

其中y_L表示所有带标签的结点的集合。通过梯度下降法我们可以训练神经网络的权重。

2.2 深度图卷积神经网络

研究发现，超过 2-3 层后，随着层数增加，GCN 的性能会急剧下降。研究者在加深 GCN 上做了一些尝试，提出了一些深度 GCN 模型。

2.2.1 PPNP

传统的 GCN 模型在每一层变换时包括特征变换和 1 阶邻居的聚合，通常只能使用有限的邻居结点信息并且难以扩展，但是边缘结点，稀疏结点等需要更多的邻居结点信息。然而，简单地堆叠层以获取更多邻居结点信息会带来两个问题：一是聚合次数过多会导致过平滑，丧失了结点的局部特性；二是堆叠层数过多会导致参数量过大，有可能造成过拟合。

受 Personalized PageRank 启发，研究者提出一种新的传播算法，该算法可以平衡局部性和对更大范围邻居信息的需求，从而缓解过平滑的问题。此外将神经网络与传播过程分离，神经网络的深度完全独立于传播过程，从而回避过拟合的问题[6]。普通的 PageRank 见公式（2.12）表述。

$$\pi_{pr} = A_{rw}\pi_{pr}, A_{rw} = AD^{-1} \tag{2.12}$$

Personalized PageRank 考虑了根节点i_x，见公式（2.13），其中$\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$是添加了自循环的对称归一化邻接矩阵。

$$\pi_{ppr}(i_x) = (1-\alpha)\hat{A}\pi_{ppr}(i_x) + \alpha i_x \tag{2.13}$$

求解公式（2.13）并矩阵化后见公式（2.14）表述。它的每个元素(yx)表示结点 x 对 y 的影响分数大小。

$$\prod_{ppr} = \alpha\left(I_n - (1-\alpha)\hat{A}\right)^{-1} \tag{2.14}$$

为了利用上述 Personalized PageRank 影响分数，我们将该分数与高层特征一起用于生成每个结点的类别概率分布，见公式（2.15）表述。

$$Z_{PPNP} = softmax\left(\alpha\left(I_n - (1-\alpha)\hat{A}\right)^{-1}H\right), H_{i,:} = f_\theta\left(X_{i,:}\right) \tag{2.15}$$

其中X是结点的输入特征矩阵，$H \in R^{n\times c}$是结点的隐层特征矩阵，f_θ表示任意特征映射函数，如神经网络，f_θ独立地对每个结点进行变换，该过程中不涉及聚合操作。

直接计算矩阵$\prod_{ppr}$的时间复杂度和空间复杂度都很高，我们可以用迭代的算法近似求解，见公式（2.16）-（2.18）表述。

$$Z^{(0)} = H = f_\theta(X) \tag{2.16}$$

$$Z^{(k+1)} = (1-\alpha)\hat{A}Z^{(k)} + \alpha H \tag{2.17}$$

$$Z^{(K)} = softmax\left((1-\alpha)\hat{A}Z^{(k-1)} + \alpha H\right) \tag{2.18}$$

其中K是超参数，表示迭代轮数。我们可以通过设置转移概率α控制邻居结点的范围，对于不同类型的图和任务选择不同的转移概率α。

PPNP 的传播部分借鉴的是 Personalized PageRank，在该过程中聚合图的结构信息，它是基于经验假设设计的，直接进行量化计算而不需要参数学习过程，但是也失去了神经网络的优势。本文在增强结点自身的角度，将 Personalized PageRank 引入了 GCN，可以同时自动学习结构信息和特征信息。

2.2.2 JK-Net

虽然 GCN 能适应不同结构的图数据，但是 GCN 固定的层级结构无法满足不同邻域结构的结点对平滑范围的要求。K 步随机游走在不同邻域结构的结点上的效果见图 2.1，从左往右分别是中心结点的 4 步随机游走，边缘结点的 4 步随机游走和 5 步随机游走。可以看到，位于连接紧密的中心结点，平滑范围扩散过快；位于连接稀疏的边缘结点，平滑范围扩散过慢，但是一旦触及中心，平滑范围就会陡增。

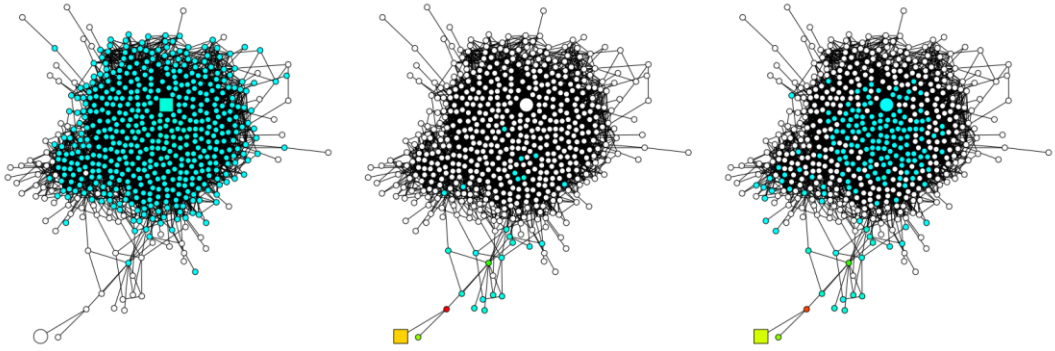

图 2.1 K 步随机游走在不同邻域结构的结点上的效果

底层信息更具局部性，高层信息更具全局性，JK-Net 以层级聚合的方式自适应的融合不同层的信息，从而平衡不同邻域结构结点的局部与全局信息[3]，见图 2.2。具体的融合方式有 Concatenation、Max-pooling 和 LSTM-attention。

Concatenation 将各层输出拼接，之后作线性变换用于分类；Max-pooling 将各层输出聚在一起做元素级别的最大池化操作。

LSTM-attention 是最复杂的融合方式，为每层学习注意力系数，该系数表示各层的重要程度。将各层输入依次送入一个双向 LSTM，将每层的前向表达和后向表达拼接后作线性变换得到一个系数，对该系数作 softmax 归一化得到最终的注意力系数，最后对各层输出依据注意力系数加权求和。

加入融合机制后，GCN 就存在两种聚合方式，横向的邻居聚合学习结构信息，纵向的层级聚合促使模型有选择地学习结构信息，从而使得 GCN 模型能够堆叠更多层。

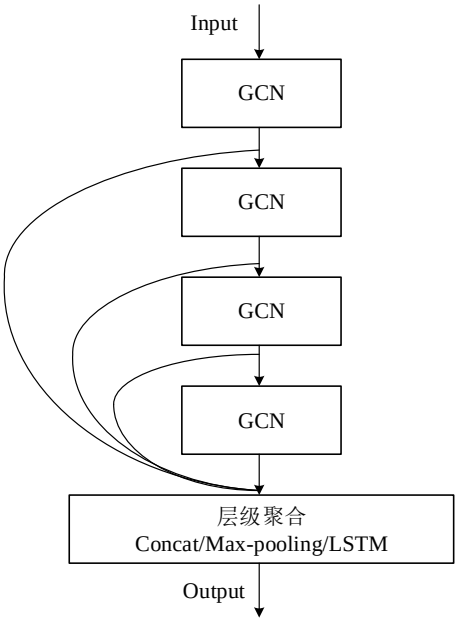

图 2.2 JK-Net 网络结构

JK-Net 能自适应地根据结点的邻域结构组合不同层的信息，但是由于 JK-Net 只在最后一层对所有层进行融合，层之间的传播方式没有改变，较深层产生的输出仍然存在不同类簇间的结点混合问题。本文引入 DenseNet 对此作了改进。

2.2.3 Cluster-GCN

近距离的邻居结点比远距离的邻居结点贡献更大，通过放大 GCN 中邻接矩阵的对角部分，可以在每层的聚合中对上一层的表达施加更多权重[7]，见公式（2.19）表述。

$$X^{(l+1)} = \sigma\left(\left(\hat{A} + I\right)X^{(l)}W^{(l)}\right) \tag{2.19}$$

然而该改进存在两个弊端：一是对所有结点使用相同的权重可能是不合理的；二是随着层数增加可能会导致数值不稳定。我们可以为原始矩阵添加自循环后再标准化，从而规避数值不稳定问题，见公式（2.20）表述。

$$\tilde{A} = (D + I)^{-1}(A + I) \tag{2.20}$$

接着考虑到结点的权重添加正则化，见公式（2.21）表述。

$$X^{(l+1)} = \sigma(\left(\tilde{A} + \lambda diag\left(\tilde{A}\right)\right)X^{(l)}W^{(l)} \tag{2.21}$$

基于矩阵的实现，当数据集庞大时，就会面临内存限制问题，而大数据集恰恰最需要更深的 GCN。本文基于 DGL 框架实现模型，采用消息发送与接收的方式进行邻居聚合和结点更新。

2.2.4 N-GCN

受 Inception 的启发，我们可以在不同尺度下进行卷积，最后融合所有卷积结果得到结点的特征表示，通过组合不同尺寸感受野来提高模型的表征能力[8]。N-GCN 的原理见公式（2.22）表述。

$$N - GCN_{fc}\left(\hat{A}, A; W_{fc}, \theta\right) = softmax([GCN(\hat{A}^0, X; \theta^{(0)}, GCN\left(\hat{A}^1, X; \theta^{(1)}, \ldots\right]W_{fc}) \tag{2.22}$$

N-GCN 相当于采用 Concatenation 融合方式的 JK-Net，较深层产生的输出仍然存在不同类簇间的结点混合问题。

2.2.5 RGCN

对于一个 n 层的 GCN，第 i 层捕获了 i-hop 邻居结点的信息，相邻层之间存在依赖关系，我们可以用 RNN 对各层之间的长期依赖建模。RGCN 的原理见公式（2.23）-（2.24）表述。

$$H^{l+1} = RNN(GNN(H^l, A; \theta^l), H^l), l \geq 0 \tag{2.23}$$

$$H^0 = RNN(W_i X + b_i, 0) \tag{2.24}$$

通过引入门控机制控制信息的累积速度，从而改善循环神经网络的长程依赖问题，这就是基于门控的循环神经网络。具体地，包含两个步骤，一是有选择地加入新的信息，二是有选择地遗忘旧的信息。长短期记忆网络和门控循环单元网络是常见的两种。

长短期记忆网络 LSTM 是循环神经网络的一个变体，可以有效地解决简单循环神经网络的梯度消失/爆炸问题。将 LSTM 引入 GCN，RGCN-LSTM 的原理见公式（2.25）-（2.31）表述。

$$X^{l+1} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}\Theta^{(l)} \tag{2.25}$$

$$I^{l+1} = \sigma(X^{l+1}W_i + H^l U_i + [b_i]_N) \tag{2.26}$$

$$F^{l+1} = \sigma(X^{l+1}W_f + H^l U_f + \left[b_f\right]_N) \tag{2.27}$$

$$O^{l+1} = \sigma(X^{l+1}W_o + H^l U_o + [b_o]_N) \tag{2.28}$$

$$\hat{C}^{l+1} = tanh(X^{l+1}W_c + H^l U_c + [b_c]_N) \tag{2.29}$$

$$C^{l+1} = F^{l+1} \circ C^{l} + I^{l+1} \circ \hat{C}^{l+1} \tag{2.30}$$

$$H^{l+1} = O^{l+1} \circ \tanh\left(C^{l+1}\right) \tag{2.31}$$

门控循环单元网络 GRU 是一种比 LSTM 网络更加简单的循环神经网络。GRU 网络引入门控机制来控制信息更新的方式。和 LSTM 不同，GRU 不引入额外的记忆单元。GRU 引入一个更新门来控制当前状态需要从历史状态中保留多少信息，以及需要从候选状态中接受多少信息。将 GRU 引入 GCN，RGCN-GRU 的原理见公式（2.32）-（2.36）表述。

$$X^{l+1} = \widetilde{D}^{-\frac{1}{2}}\tilde{A}\widetilde{D}^{-\frac{1}{2}}H^{(l)}\Theta^{(l)} \tag{2.32}$$

$$Z^{l+1} = \sigma(X^{l+1}W_z + H^{l}U_z + [b_z]_N) \tag{2.33}$$

$$R^{l+1} = \sigma(X^{l+1}W_r + H^{l}U_r + [b_r]_N) \tag{2.34}$$

$$\widehat{H}^{l+1} = tanh(X^{l+1}W_h + U_h(R^{l+1} \circ H^{l}) + [b_h]_N) \tag{2.35}$$

$$H^{l+1} = (1 - Z^{l+1}) \circ H^{l} + Z^{l+1} \circ \widehat{H}^{l+1} \tag{2.36}$$

与 JK-Net 在最后一层使用 LSTM 融合各层信息不同，RGCN 使用 RNN 对各层之间的长期依赖建模，缓解了更深的层不同类簇结点混合的问题。

2.2.6 DeepGCN

借鉴深度 CNN 的经验，我们可以将残差连接、密集连接引入 GCN 解决由于网络加深导致的梯度消失/爆炸问题，将空洞卷积引入 GCN 解决由于池化操作导致的空间信息丢失问题[10]。

残差网络 ResNet 通过给卷积层增加残差连接，从而提高信息在网络中的传播效率。将残差连接引入 GCN，ResGCN 的原理见公式（2.37）描述。

$$G_{l+1} = \mathcal{H}(G_l, W_l) = \mathcal{F}(G_l, W_l) + G_l \tag{2.37}$$

密集网络 DenseNet 通过密集连接来改进信息流并重用层之间的特征。将密集连接引入 GCN，以利用不同层的信息流，DenseGCN 的原理见公式（2.38）描述。

$$\begin{aligned} G_{l+1} &= \mathcal{H}(G_l, W_l) \\ &= \mathcal{T}(\mathcal{F}(G_l, W_l), G_l) \\ &= \mathcal{T}(\mathcal{F}(G_l, W_l), \dots, \mathcal{F}(G_0, W_0), G_0) \end{aligned} \tag{2.38}$$

空洞卷积是一种不增加参数数量，同时增加输出单元感受野的算法，也称为膨胀卷积。空洞卷积通过在卷积核中插入“空洞”来扩大其感受野尺寸。在特征空间上使用 L2 距离，根据与目标结点的距离将邻居结点排序，见公式（2.39）表述。

$$u_1, u_2, \dots, u_{k \times d} \tag{2.39}$$

给定空洞系数d，目标结点的邻居结点见公式（2.40）表述。

$$\mathcal{N}^{(d)}(v) = u_1, u_{1+d}, u_{1+2d} \ldots, u_{1+(k-1)d} \tag{2.40}$$

DeepGCN 的动机在于解决梯度消失/爆炸，但是它不是阻碍 GCN 加深的主要原因，同时，DeepGCN 只在点云数据集上进行了实验，该任务属于图层次的分类，每张图之间不连通，不存在过平滑问题。本文在引用数据集等多个图数据集上实验了 ResGCN、DenseGCN，并提出了带可学习权重的 ResGCN。

2.2.7 DropEdge

DropEdge 在每轮训练中随机删除图数据集中一定数量的边，在验证集和测试集上不使用 DropEdge 机制[11]。具体而言，在邻接矩阵A中随机选取VP个非零元素置零，其中V是原始图的总边数，P是删除概率，最后得到邻接矩阵A_{drop}，见公式（2.41）描述。

$$A_{drop} = A - A' \tag{2.41}$$

接着对A_{drop}添加自循环并做对称归一化，将得到的结果$\hat{A}_{drop}$代替 GCN 中的$\hat{A}$。我们可以让所有层共享同一个$\hat{A}_{drop}$，也可以在每一层进行 DropEdge，在第l层得到邻接矩阵$\hat{A}^{(l)}{}_{drop}$，这样可以赋予原始数据更多随机性。

从数据增强角度，DropEdge 在训练中不断随机删除原始图的边，增强了输入数据的随机性和多样性，从而缓解了过拟合问题。从消息传递角度，GCN 中结点间通过连边进行消息传递，随机删除一些边可以使得结点连接变稀疏，从而缓解了过平滑问题。

在基于结点采样的算法 DropNode 中，删除某个结点相当于删除了与该结点相连的所有边，可以视为 DropEdge 的特殊形式。与 DropNode 相比，DropEdge 是面向边的，保留了所有结点的特征，更具灵活性。此外 DropEdge 对所有边的采样是并行的，更具高效性。

Dropout 是一种正则化算法，在训练中随机丢弃一部分神经元，即随机将特征向量的部分维度置零，与 DropEdge 相比，它可以缓解过拟合但是不能缓解过平滑。图稀疏性通过复杂的优化算法删掉部分边来压缩图，与 DropEdge 相比时间复杂度往往很高。

DropEdge 是一种简单而高效的算法，但是实验发现，在引用数据集等稀疏图上表现良好，在其他几个密集图数据集上却起到了反作用，这可能是因为相当一部分有效边被随机删除了。本文基于 DropEdge 做了一些改进，分别是基于结点度数的 DropEdge 和基于特征相似 DropEdge。

2.2.8 PairNorm

给定$\bar{X} \in R^{n \times d}$表示结点的新特征矩阵，其中$\bar{x}_i \in R^d$表示特征矩阵$\bar{X}$的第$i$行，图上的正则化最小平方 GRLS 优化问题见公式（2.42）描述[14]。

$$\min_{\bar{X}} \sum_{i\in\mathcal{V}} \|\bar{x}_i - x_i\|_{\tilde{D}}^2 + \sum_{(i,j)\in\mathcal{E}} \left\|\bar{x}_i - \bar{x}_j\right\|_2^2 \tag{2.42}$$

这里运算$\|z_i\|_{\tilde{D}}^2 = z_i^T \tilde{D} z_i$，第一项可以看做带度数权重的最小平方，第二项表示图结构上新特征之间的差异。该优化问题的目标在于保证新特征与原特征相似，同时促使新特征在图上更平滑。

GRLS 问题有解析解$\bar{X} = (2I - \tilde{A}_{rw})^{-1}X$，其中$\tilde{A}_{rw} = \tilde{D}^{-1}\tilde{A}$，$\tilde{A}_{rw}X$是一阶泰勒近似，即$\tilde{A}_{rw}X \approx \bar{X}$。用$\tilde{A}_{sym} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$代替$\tilde{A}_{rw}$，也就是$\tilde{X} = \tilde{A}_{sym}X \approx \bar{X}$。因此，图卷积是 GRLS 问题的近似解。

理想情况下，我们希望同一类簇类更平滑，同时不同类簇间不平滑，但是公式（2.42）只能确保前者。为了同时实现两个目标，我们可以在公式（2.42）中增加一项，该项表示不相连结点对之间的距离总和，见公式（2.43）描述。

$$\min_{\bar{X}} \sum_{i\in\mathcal{V}} \|\bar{x}_i - x_i\|_{\tilde{D}}^2 + \sum_{(i,j)\in\mathcal{E}} \left\|\bar{x}_i - \bar{x}_j\right\|_2^2 - \lambda \sum_{(i,j)\notin\mathcal{E}} \left\|\bar{x}_i - \bar{x}_j\right\|_2^2 \tag{2.43}$$

其中系数λ用于平衡两个目标的重要程度。我们可以求出公式（2.43）的解析解，并用图卷积近似，从而得到带系数λ的新的图卷积操作，但是这样不具备通用性。

给定图卷积的输出$\tilde{X}$作为 PairNorm 层的输入，$\dot{X}$是 PairNorm 层的输出。图卷积$\tilde{X} = \tilde{A}_{sym}X$只实现了第一个目标，PairNorm 作为正则化层，通过增加不相连结点对间的总距离来实现第二个目标。结点对间的总距离记做 TPSD，PairNorm 确保$TPSD(\dot{X}) = TPSD(X)$，见公式（2.44）描述。

$$\sum_{(i,j)\in\mathcal{E}} \left\|\dot{x}_i - \dot{x}_j\right\|_2^2 + \sum_{(i,j)\notin\mathcal{E}} \left\|\dot{x}_i - \dot{x}_j\right\|_2^2 = \sum_{(i,j)\in\mathcal{E}} \left\|x_i - x_j\right\|_2^2 + \sum_{(i,j)\notin\mathcal{E}} \left\|x_i - x_j\right\|_2^2 \tag{2.44}$$

随着图卷积操作的不断平滑，$\sum_{(i,j)\in\mathcal{E}}\left\|\dot{x}_i - \dot{x}_j\right\|_2^2$和$\sum_{(i,j)\in\mathcal{E}}\left\|x_i - x_j\right\|_2^2$越来越接近，因此$\sum_{(i,j)\notin\mathcal{E}}\left\|\dot{x}_i - \dot{x}_j\right\|_2^2$和$\sum_{(i,j)\notin\mathcal{E}}\left\|x_i - x_j\right\|_2^2$也越来越接近。$TPSD(X)$是与数据集特性相关的常数，记做超参数 C。

为了对$\tilde{X}$进行正则化，我们需要计算$TPSD(\tilde{X})$，然而对于大数据集直接计算时间复杂度很高。$TPSD(\tilde{X})$的等价形式见公式（2.45）描述。

$$\begin{aligned} TPSD(\tilde{X}) &= \sum_{i,j\in[n]} \left\|\tilde{x}_i - \tilde{x}_j\right\|_2^2 \\ &= 2n^2\left(\frac{1}{n}\sum_{i=1}^{n}\|\tilde{x}_i\|_2^2 - \left\|\frac{1}{n}\sum_{i=1}^{n}\tilde{x}_i\right\|_2^2\right) \end{aligned} \tag{2.45}$$

为进一步简化，将$\tilde{x}_i$中心化，第二项为 0，TPSD 的值不变。具体地，PairNorm 分为两步，中心化见公式（2.46）表述[13]，缩放化见公式（2.47）表述。

$$\tilde{x}_i^c = \tilde{x}_i - \frac{1}{n}\sum_{i=1}^{n}\tilde{x}_i \tag{2.46}$$

$$\dot{x}_i = s \cdot \frac{\tilde{x}_i^c}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left\|\tilde{x}_i^c\right\|_2^2}} = s\sqrt{n} \cdot \frac{\tilde{x}_i^c}{\sqrt{\left\|\tilde{X}^c\right\|_F^2}} \tag{2.47}$$

PairNorm 具有坚实的理论基础，但是由于扩大的是所有不相连结点对间的差异，总体来说对于缓解过平滑问题效果有限，在受过平滑较严重的密集连接图数据集上表现不佳。本文提出的基于 DropEdge 的改进算法直接针对过平滑问题，取得了较好的效果。

2.3 本章小结

本章首先介绍了图卷积神经网络的理论基础和模型详情，以及 GCN 在图的半监督结点分类任务上的应用，接着介绍了 8 种主要的深度图卷积神经网络模型，讨论分析了它们各自的优缺点以及本文的主要工作以及在已有工作上的改进。

第 3 章 实验规范

本章介绍了实验中用到的 9 个开源数据集，并说明了后几章通用的一些实验设置，如数据集划分、损失函数选择，评价指标选择等。

3.1 实验数据

本文使用 9 个开源图数据集验证提出的算法和模型[15]。这些数据集的详细信息如表 3.1 所示。

表 3.1 图数据集

Dataset	Nodes	Edges	Features	Classes
Cora	2708	5429	1433	7
Cite.	3327	4732	3703	6
Pubm.	19717	44338	500	3
Cham.	2277	36101	2325	4
Squi.	5201	217073	2089	4
Actor	7600	33544	931	4
Corn.	183	295	1703	5
Texa.	183	309	1703	5
Wisc.	251	499	1703	5

Citiation networks：Cora、Citeseer 和 Pubmed 是 3 个标准的引用网络基准数据集。在引用网络中，结点表示论文，边表示论文之间的引用关系。结点特征是论文的词袋模型表示，结点标签是论文的学术主题。

WebKB：WebKB 是从各大学计算机系收集的网页数据集，我们使用了它的 3 个子集：Cornell、Texas 和 Wisconsin。在 WebKB 数据集中，结点表示网页，边表示网页之间的超链接关系。结点特征是网页的词袋模型表示。网页被人为分成 5 类：student、project、course、staff 和 faculty。

Actor co-occurrence network：该数据集是电影-导演-演员-编剧网络的诱导子图，只包含了演员。结点表示演员，边表示演员在同一维基百科页面的共现关系。结点特征表示维基百科页面的某些关键词。我们人为将其分为 4 类。

Wikipedia network：Chameleon 和 Squirrel 是维基百科中特定主题下的 page-page 网络。结点表示网页，边表示网页之间的相互链接关系。结点特征是维基百科页面中的一些信息量丰富的名词。我们人为将其分为 4 类。

可以看到，Cora、Citeseer 和 Pubmed 边比较少，连接比较稀疏。而 Chameleon、Squirrel 和 Actor 边比较多，连接比较密集，因此过平滑问题也更严重。

3.2 实验设置

(1) 数据集划分

对于所有图数据集，我们按照 60%、20%、20%的比例将其划分为训练集、验证集和测试集。其中训练集用于训练模型，验证集用于超参数寻优、测试集用于评估模型。

对于过拟合和梯度消失/爆炸问题，我们只使用引用数据集验证模型。对于过平滑问题，我们使用所有数据集验证模型。

(2) 损失函数

我们用交叉熵损失函数来训练模型。交叉熵损失函数一般用于分类问题。假设样本的标签$y \in \{1, \ldots, C\}$为离散的类别，模型$f(x;\theta) \in [0,1]^C$的输出类别为类别标签的条件概率分布，即$p(y = c|x;\theta) = f_c(x;\theta)$，并满足$f_c(x;\theta) \in [0,1], \sum_{c=1}^{C} f_c(x;\theta) = 1$。我们可以用一个 C 维的 one-hot 向量 y 来表示样本标签。给定样本标签 k，则标签向量 y 只有第 k 个维度值是 1，其余维度的值都是 0。标签向量 y 相当于样本标签的真实条件概率分布$p_r(y|x)$，即第 c 维是类别为 c 的真实条件概率。对于两个概率分布，一般可以用交叉熵来衡量它们的差异，标签的真实分布 y 和模型预测分布$f(x;\theta)$之间的交叉熵见公式（3.1）描述。

$$\begin{aligned} \mathcal{L}\big(y, f(x;\theta)\big) &= -y^T log f(x;\theta) \\ &= -\sum_{c=1}^{C} y_c log f_c(x;\theta) \end{aligned} \tag{3.1}$$

(3) 评价指标

准确率、精确率、召回率和 F 值等是分类任务常用到的评价指标。给定测试集$\mathcal{T} = \{\big(x^{(1)}, y^{(1)}\big), \ldots, \big(x^{(N)}, y^{(N)}\big)\}$，假设标签$y^{(n)} \in \{1, \ldots, C\}$，用学习好的模型$f(x;\theta^*)$对测试集中的每一个样本进行预测，结果为$\{\hat{y}^{(1)}, \ldots, \hat{y}^{(N)}\}$。我们用准确率来评价模型，见公式（3.2）描述，其中$I(\cdot)$为指示函数。

$$\mathcal{A} = \frac{1}{N}\sum_{n=1}^{N} I(y^{(n)} = \hat{y}^{(n)} \tag{3.2}$$

(4) 超参数寻优

我们用网格搜索进行超参数寻优。网格搜索通过尝试超参数的所有组合，从而找到一组合适的超参数配置。假设总共有K个超参数，第k个超参数可以取 m_k个值，那么总共的配置组合数量为$m_1 \times m_2 \times \ldots \times m_K$。网格搜索在不同的超参数组合上分别训练模型，接着测试它们在验证集上的性能来选取一组最好的配置。

(5) 激活函数

常用的非线性激活函数有 Sigmoid、ReLU 等，原始的 GCN 采用 ReLU 作为激活函数。然而研究表明，由于过平滑问题，随着层数加深，在 GCN 中 Tanh 函数更有利于保持特征列之间的线性无关性[12]，效果比 ReLU 要好，因此本文也采用 Tanh 作为激活函数。

(6) 优化算法

常用的优化算法有动量法、Nesterov 加速梯度、RMSprop 算法、Adam 算法等[16]。Adam 算法是动量法和 RMSprop 算法的结合，不仅使用动量作为参数更新方向，而且可以自适应调整学习率。本文统一采用 Adam 优化算法，初始学习率设置为 1e-2。

第 4 章　面向过拟合的算法

过拟合是限制传统神经网络加深的问题，本章首先对该问题进行了理论分析，接着引入了 3 种正则化算法：权重衰减、提前终止和丢弃法，最后在引用数据集上进行了实验。

4.1　问题定义

过拟合可以形式化地定义为：给定一个假设空间$\mathcal{F}$，一个假设f属于$\mathcal{F}$，如果存在其他的假设f'也属于$\mathcal{F}$，使得在训练集上f的损失比f'的损失小，但在整个样本空间上f'的损失比f的损失小，那么就说假设f过度拟合训练数据[17]。

我们可以用期望风险$\mathcal{R}(\theta)$衡量模型$f(x;\theta)$的好坏，见公式（4.1）表述。其中$\mathcal{L}(y,f(x;\theta))$是损失函数，描述了两个变量的差异。$p_r(x,y)$是数据的真实分布。

$$\mathcal{R}(\theta) = E_{(x,y)\sim p_r(x,y)}\left[\mathcal{L}\left(y,f(x;\theta)\right)\right] \tag{4.1}$$

一般来说，期望风险越小，表示模型$f(x;\theta)$越优秀。但是我们无法获悉数据的真实分布和映射函数，因此也无法计算模型的期望风险$\mathcal{R}(\theta)$。给定一个训练集$\mathcal{D} = \{\left(x^{(n)},y^{(n)}\right)\}_{n=1}^{N}$，我们可以计算训练集上的平均损失，也就是经验风险，见公式（4.2）表述。

$$\mathcal{R}_{\mathcal{D}}^{emp}(\theta) = \frac{1}{N}\sum_{n=1}^{N}\mathcal{L}\left(y^{(n)},f\left(x^{(n)};\theta\right)\right) \tag{4.2}$$

因此，我们可以采用经验风险最小化原则作为学习准则，也就是找到一组使得经验风险最小的参数θ^*，见公式（4.3）表述。

$$\theta^* = arg\min_{\theta}\mathcal{R}_{\mathcal{D}}^{emp}(\theta) \tag{4.3}$$

根据大数定理理论，经验风险会随着训练集规模趋向无穷大而趋向于期望风险。但是实际情况是，我们一般无法获取足够数量的样本。训练样本往往是从真实数据采样的一个非常小的子集，并且该子集中通常会包含一些噪声数据。因此训练样本不能很好的反映数据的真实分布。经验风险最小化常常会导致过拟合，即在训练集上的准确率很高，但是在测试集也就是未知数据上的准确率很低。

导致过拟合问题的原因有训练数据较少，数据包含噪声，模型能力过强等。正则化算法通过限制模型复杂度来提高模型的泛化能力，从而缓解在训练集上过拟合。常用的正则化算法有权重衰减、提前终止、丢弃法等。

4.2 权重衰减

权重衰减通过在每次更新参数时引入一个衰减系数限制模型复杂度，从而缓解过拟合问题，是一种有效的正则化算法，见公式（4.4）表述。其中β是权重衰减系数，通常取比较小的值。α为学习率，g_t为第t步更新时的梯度。

$$\theta_t \leftarrow (1-\beta)\theta_{t-1} - \alpha g_t \tag{4.4}$$

在标准的随机梯度下降中，权重衰减正则化相当于$\mathcal{L}_2$正则化。因此，在一些深度学习框架中权重衰减通过$\mathcal{L}_2$正则化来实现。$\mathcal{L}_1$和$\mathcal{L}_2$正则化是两种最常用的正则化算法，通过限制参数的$\mathcal{L}_1$和$\mathcal{L}_2$范数，从而缓解模型在训集上的过拟合问题。通过加入$\mathcal{L}_1$和$\mathcal{L}_2$正则化，优化问题见公式（4.5）描述。

$$\theta^* = arg\min_{\theta} \frac{1}{N}\sum_{n=1}^{N} \mathcal{L}\left(y^{(n)}, f\left(x^{(n)};\theta\right)\right) + \lambda \mathcal{L}_p(\theta) \tag{4.5}$$

这里N是训练样本的数量，$\mathcal{L}(\cdot)$是损失函数，$f(\cdot)$为待学习的模型，θ是它的参数，$\mathcal{L}_p$是范数函数，p表示范数的类型，取值为{1,2}时表示$\mathcal{L}_1$和$\mathcal{L}_2$范数，λ是正则化系数。带正则化的优化问题可以转化为带约束条件的优化问题，见公式（4.6）-（4.7）表述。

$$\theta^* = arg\min_{\theta} \frac{1}{N}\sum_{n=1}^{N} \mathcal{L}\left(y^{(n)}, f\left(x^{(n)};\theta\right)\right) \tag{4.6}$$

$$\mathcal{L}_p(\theta) \leq 1 \tag{4.7}$$

对于给定的特征向量$\mathrm{x} = [\mathrm{x}_1, x_2, \ldots, x_n]^T$，其$p$范数见公式（4.8）表述。

$$\|x\|_p = (|x_1|^p + |x_2|^p + \cdots + |x_n|^p)^{\frac{1}{p}} \tag{4.8}$$

4.3 提前终止

由于深度神经网络的学习能力很强，因此特别容易在训练集上过拟合。在梯度下降优化的过程中，我们可以用验证集上的错误代替期望错误，当验证集的错误率不再下降，就停止模型的迭代。验证集也叫开发集，常用于超参数寻优。验证集的错误率变化不一定是平缓曲线，可能会在某处先升高再降低，因此，我们需要根据实际任务进行优化，选取恰当的早停窗口。

具体而言，提前终止主要有三种停止标准。

第一类停止标准是指，当泛化损失超过指定阈值时停止训练，泛化损失见公式（4.9）表述，它表示的是当前迭代周期中，泛化误差相对目前最小误差的增长率。其中$E_{va}(t)$表示第t次迭代时验证集的误差，描述的是泛化误差，$E_{opt}(t) \coloneqq \min \mathrm{E}_{\mathrm{va}}(t'), t' \leq t$表示迭代$t$次后取得的最小的验证集误差。

$$GL(t) = 100 \cdot \left(\frac{E_{va}(t)}{E_{opt}(t)} - 1 \right) \tag{4.9}$$

第二类停止标准基于一个假设：过拟合出现在训练集误差降低很慢的时候。也就是训练集误差依然下降很快时，泛化误差可能会在未来被修正。给定一个周期k，度量进展见公式（4.10）描述。

$$P_k(t) = 1000 \cdot \left(\frac{\sum_{t'=t-k+1}^{t} E_{tr}(t')}{k \cdot min_{t'=t-k+1}^{t} E_{tr}(t')} - 1 \right) \tag{4.10}$$

度量进展描述了在某段时间内训练集误差的平均下降情况。当训练过程不稳定时，该变量的值可能会很大。训练了较长时间后，该变量会趋向于 0。因此，引入第二类停止标准，泛化损失和度量进展的比值超过指定阈值时停止训练，该比值见公式（4.11）描述。

$$PQ_\alpha = \frac{GL(t)}{P_k(t)} \tag{4.11}$$

第三类停止标准完全基于泛化误差的变化，在连续k个周期内泛化误差持续增长时停止训练。该停止标准可以用作剪枝算法。

4.4 丢弃法

通过随机丢弃一定比例的神经元，从而缓解深度神经网络在训练集上过拟合，这就是丢弃法[18]。给定神经网络层$y = f(Wx + b)$，我们可以引入掩蔽函数$mask(\cdot)$，得到$y = f(Wmask(x) + b)$。掩蔽函数$mask(\cdot)$的定义见公式（4.12）描述。

$$mask(x) = \begin{cases} m \odot x & \text{训练阶段} \\ px & \text{测试阶段} \end{cases} \tag{4.12}$$

这里$m \in \{0,1\}^D$是通过概率为p的的零一分布生成的丢弃掩码。在训练阶段，激活的神经元的平均数量只有原来的比例p。在测试阶段，所有神经元都被激活。因此训练和测试阶段神经网络的输出不一致。我们可以在测试阶段将神经层输入x乘上p，从而缓解该问题。我们可以用验证集来选取一个最优的保留率p。一般来说，将隐藏层的保留率设置为$p = 0.5$最好，这适用于大多数网络和任务。当$p = 0.5$时，在训练阶段丢弃了一半的神经元，还可以激活一半的神经元，相当于不同的神经网络的平均，更具多样性。对于输入层的神经元，通常将保留率设置为近似 1 的值，这样输入的变化不会太大。丢弃输入层的神经元，相当于在数据中增加噪声，训练出的模型更具鲁棒性。

从集成学习的角度，每丢弃一次神经元，相当于从原始网络采样一个子网络。对于一个有n个神经元的网络，一共可以采样出2^n个子网络，这些子网络共享原始网络的参数，最终训练得到的网络相当于指数级别数量的网络的组合模型[17]。

4.5 实验分析

(1) 权重衰减

实验中采用了$\mathcal{L}_2$正则化，层数为[1,8]时正则化系数设置为10^{-3}，层数为[9,16]时正则化系数设置为10^{-2}。[2,4,8,16]层时 GCN 的准确率见表 4.1。其中 GCN(WD)表示使用了权重衰减的 GCN。

表 4.1 权重衰减算法的实验结果

数据集	模型	2 层	4 层	8 层	16 层
Cora	GCN	87.15	85.94	86.35	26.51
	GCN(WD)	85.94	87.55	85.94	42.17
Citeseer	GCN	76.42	75.94	72.64	63.21
	GCN(WD)	76.42	75	75	72.64
Pubmed	GCN	86.87	85.6	84.43	59.99
	GCN(WD)	85.55	86.11	84.69	70.59

可以看到，当层数为 2 时，权重衰减算法会起到反作用，此时会造成模型轻微的欠拟合。当层数为[4,8,16]时，模型的参数增多，学习能力变强，产生了过拟合问题。此时权重衰减算法降低了模型的复杂度，缓解了过拟合问题。特别地，当层数达到 16 层时，权重衰减算法的作用非常明显。

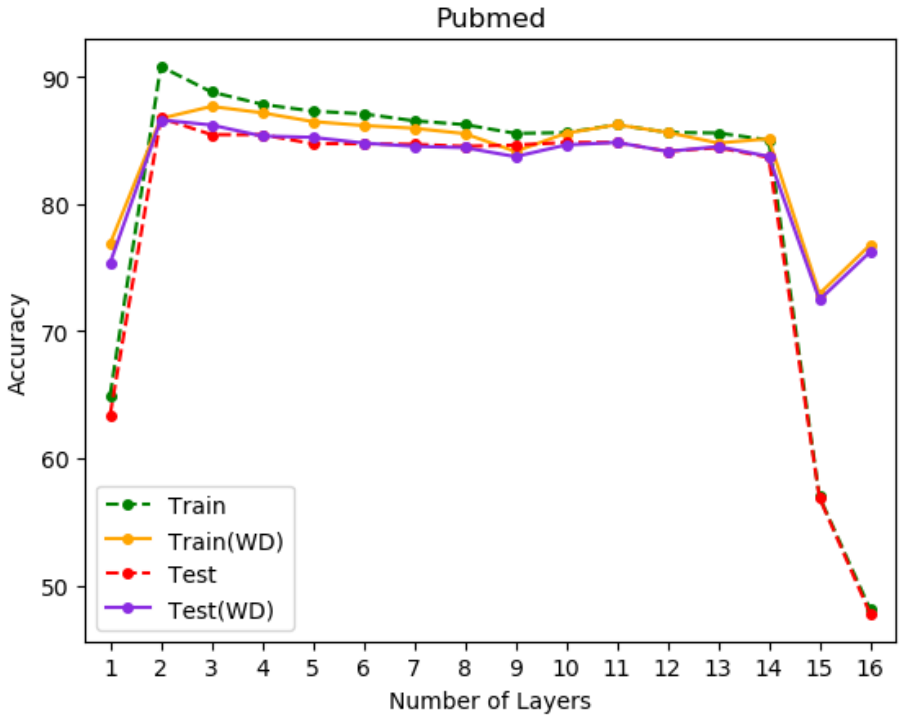

图 4.1 Pubmed 上权重衰减算法的实验结果

Pubmed 的数据规模最大，更能区分模型的有效性。我们在该数据集上实验了[1,16]层的 GCN 采用权重衰减算法的效果，见图 4.1。可以看到，当不使用权重衰减算法时，

随着层数加深性能下降，使用权重衰减算法后得到缓和。值得注意的是，当层数超过 14 层后，训练集和测试集的准确率都会骤降，这说明还有其他因素阻碍着 GCN 的加深，这就是后面会讲到的过平滑问题。

表 4.2 提前终止算法的实验结果

数据集	模型	2 层	4 层	8 层	16 层
Cora	GCN	85.14	83.94	85.14	26.51
	GCN(ES)	87.15	85.94	86.35	26.51
Citeseer	GCN	69.81	67.92	68.4	62.81
	GCN(ES)	76.42	75.94	72.64	63.21
Pubmed	GCN	86.46	84.94	84.08	40.37
	GCN(ES)	86.87	85.6	84.43	59.99

(2) 提前终止

实验中采用了第三类停止标准，以准确率作为早停指标。在其他算法的实验中，我们用该提前终止作为剪枝算法，以节省不必要的训练开销。不使用提前终止算法时，训练轮数设置为 400 轮。使用提前终止算法时，训练轮数设置为 400 轮，变化窗口设置为 50 轮。[2,4,8,16]层时 GCN 的准确率见表 4.2。其中 GCN(ES)表示采用了提前终止算法的 GCN。

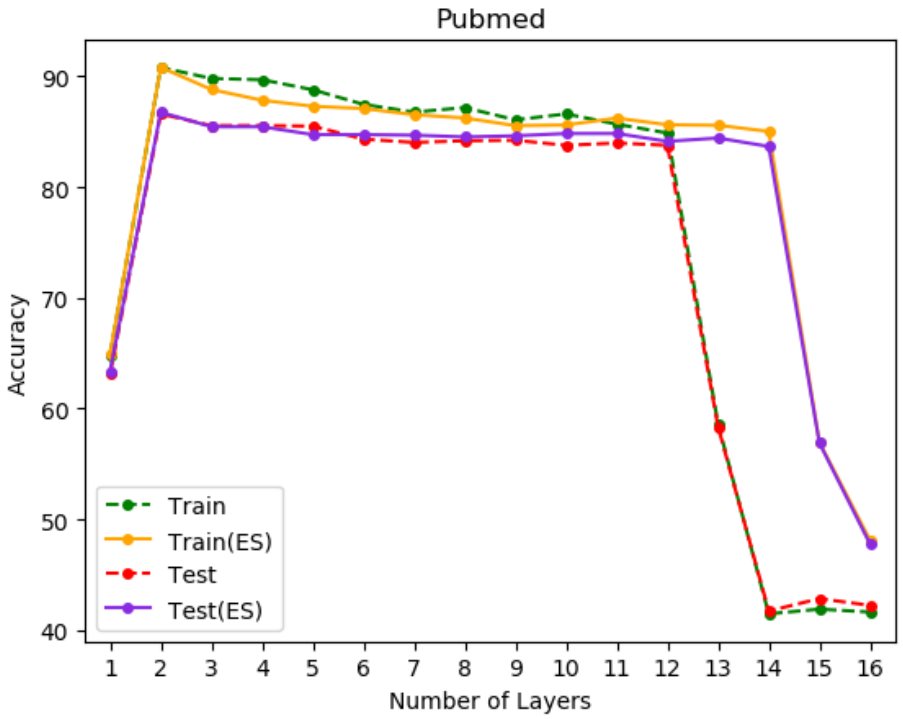

图 4.2 Pubmed 上提前终止算法的实验结果

可以看到，在所有数据集上，在任意层数 GCN 上，采用了提前终止算法都会有一定的性能提升。即使在两层的 GCN 上，如果不采用正则化算法，随着训练轮数的不断

增加，最终也会产生过拟合问题。由于 Pubmed 的结点的特征向量的维数比较小，当层数过多时过拟合会更明显。

同样地，我们在规模最大的 Pubmed 数据集上实验了[1,16]层的 GCN，见图 4.2。准确率随层数增加的整体走势与图 4.1 相似。由于在进行其他算法的实验时，都采用了提前终止作为辅助手段，所以图 4.2 的整体准确率不高。

表 4.3 丢弃法的实验结果

数据集	模型	2 层	4 层	8 层	16 层
Cora	GCN	87.15	85.94	86.35	26.51
	GCN(DO)	88.76	87.15	87.95	33.33
Citeseer	GCN	76.42	75.94	72.64	63.21
	GCN(DO)	75.94	75.94	74.53	76.42
Pubmed	GCN	86.87	85.6	84.43	59.99
	GCN(DO)	88.44	85.85	84.94	82.25

(3) 提前终止

实验中的保留率统一设置为 0.5。层数为[2,3,8,16]的 GCN 的实验结果见表 4.3。其中 GCN（DO）表示采用了丢弃法的 GCN。可以看到，相比较其他两种正则化算法，丢弃法的效果最好，GCN 的性能有较大提升。

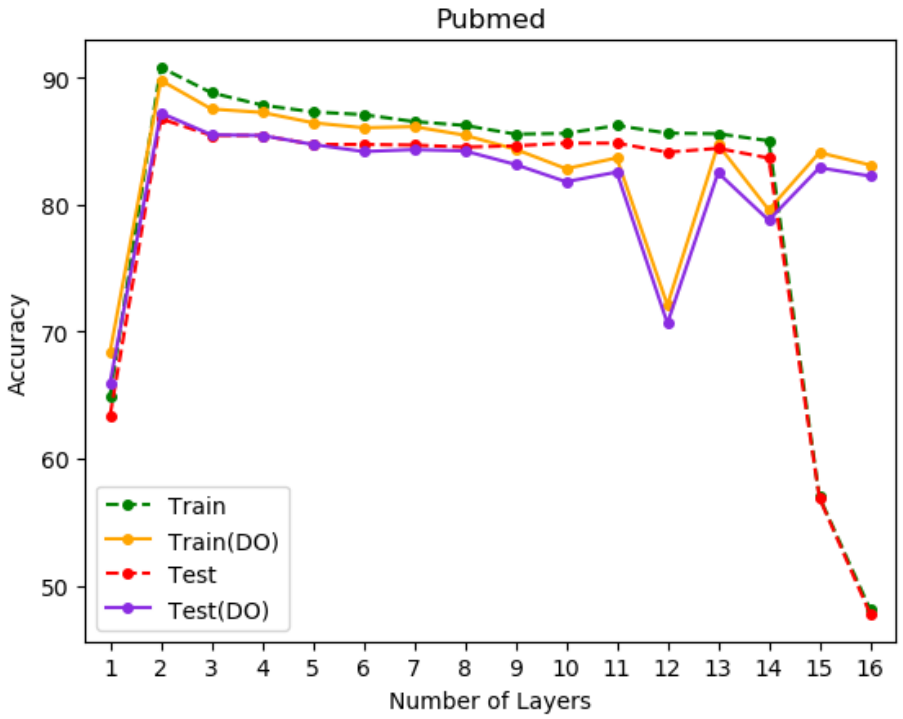

图 4.3 Pubmed 上丢弃法的实验结果

我们也在 Pubmed 数据集上实验了[1,16]层的 GCN，实验结果见图 4.3。与表 4.3 显示的情况有所出入，采用了丢弃法的 GCN 在层数为[10,15]时性能出现了波动，在 12

层时甚至大幅下降，这可能是因为丢弃法涉及了随机过程。通过设置较大的早停窗口，增加实验次数取均值等算法，我们可以获得更加准确的实验结果。

4.6 本章小结

本章首先介绍了过拟合问题的具体含义，接着介绍了三种常用的正则化算法：权重衰减、提前终止和丢弃法，并在 GCN 上做了一些实验。实验表明，过拟合问题也是限制 GCN 加深的一个因素，传统的正则化算法可以用于缓解该问题。但是实验发现，即使缓解了过拟合，当 GCN 层数超过 14 层后，训练集和测试集的准确率都会骤降，这与过拟合现象不符，限制 GCN 加深的关键因素不是过拟合。

第 5 章 面向梯度消失的算法

梯度消失是限制传统神经网络加深的问题，本章首先对该问题进行了理论分析，接着引入了 3 种传统算法：Xavier 初始化、梯度修剪和批量归一化，最后在引用数据集上进行了实验。

5.1 问题定义

在神经网络中误差反向传播的迭代公式见公式（5.1）表述。

$$\delta^{(l)} = f_l'\left(z^{(l)}\right) \odot (W^{l+1})^T \delta^{(l+1)} \tag{5.1}$$

误差在反向传播过程中，需要乘以每一层的激活函数的导数。当我们使用 Sigmoid 型函数，Logistic 函数$\sigma(x)$或$Tanh$函数时，其导数见公式（5.2）-（5.3）表述。

$$\sigma'(x) = \sigma(x)\left(1 - \sigma(x)\right) \in [0, 0.25] \tag{5.2}$$

$$tanh'(x) = 1 - (\tanh(x))^2 \in [0,1] \tag{5.3}$$

Sigmoid 型函数的导数的值域都不大于 1。此外 Sigmoid 型函数饱和区的导数近似 0。这样，误差经过每一层传播都会不断衰减。随着网络层数的加深，梯度会不断衰减乃至消失，因此使训练变得困难。这就是梯度消失问题。除了激活函数的导数，神经网络的参数的初始值也会导致梯度消失问题。类似地，还有梯度爆炸问题，统称梯度消失/爆炸问题。

5.2 Xavier 初始化

给定神经网络第l层的神经元$a^{(l)}$，其输出值见公式（5.4）表述。其中$a_i^{l-1}, 1 \le i \le M_{l-1}$为前一层$M_{l-1}$个神经元的输出 。$f(\cdot)$是激活函数，$w_i^l$是学习参数。

$$a^{(l)} = f(\sum_{i=1}^{M_{l-1}} w_i^{(l)} a_i^{(l-1)} \tag{5.4}$$

假设$f(\cdot)$为恒等激活函数，$w_i^{(l)}$和$a_i^{(l-1)}$相互独立且均值为 0，那么$a^{(l)}$的均值见公式（5.5）描述。

$$\begin{aligned} E\left(a^{(l)}\right) &= E[\sum_{i=1}^{M_{l-1}} w_i^{(l)} a_i^{(l-1)}] \\ &= \sum_{i=1}^{M_{l-1}} E\left[w_i^{(l)}\right] E\left[a_i^{(l-1)}\right] \\ &= 0 \end{aligned} \tag{5.5}$$

同样地，我们可以推导出$a^{(l)}$的方差，见公式（5.6）描述。

$$\begin{aligned} var\left(a^{(l)}\right) &= var\left(\sum_{i=1}^{M_{l-1}} w_i^{(l)} a_i^{(l-1)}\right) \\ &= \sum_{i-1}^{M_{l-1}} var\left(w_i^{(l)}\right) var\left(a_i^{(l-1)}\right) \\ &= M_{l-1} var\left(w_i^{(l)}\right) var\left(a_i^{(l-1)}\right) \end{aligned} \tag{5.6}$$

可以看到，输入信号的方差被神经元缩放了$M_{l-1}var\left(w_i^{(l)}\right)$倍。通过使每个神经元的输入与输出的方差尽可能保持一致，确保输入信号在经过许多层网络后不被过分缩放[19]。我们可以将$M_{l-1}var\left(w_i^{(l)}\right)$设置为 1，见公式（5.7）表述。

$$var\left(w_i^{(l)}\right) = \frac{1}{M_{l-1}} \tag{5.7}$$

在反向传播过程中，误差信号也会被缩放，为此我们可以采用同样的算法，见公式（5.8）表述。

$$var\left(w_i^{(l)}\right) = \frac{1}{M_l} \tag{5.8}$$

同时考虑前向传播和反向传播过程中信号的缩放，见公式（5.9）表述。

$$var\left(w_i^{(l)}\right) = \frac{2}{M_{l-1} + M_l} \tag{5.9}$$

计算出参数的约束方差后，我们可以通过均匀分布或正态分布对其进行随机初始化。如果采用正态分布，可以按$\mathcal{N}(0,2/(M_{l-1}+M_l))$进行初始化。如果采用均匀分布，可以按$[-r,r]$进行初始化，其中$r$的取值见公式（5.10）表述。上述算法就是 Xavier 初始化。

$$r = \sqrt{\frac{6}{M_{l-1} + M_l}} \tag{5.10}$$

神经元的参数和输入的绝对值一般比较小，处于 Logistic 函数和 Tanh 函数的线性区间，此时他们可以近似为线性函数，也可以使用 Xavier 初始化。在实际使用中，根据使用的激活函数，通常将方差$var(w_i^{(l)})$乘以一个缩放因子ρ。

5.3 梯度修剪

梯度修剪主要用于缓解梯度爆炸问题。在梯度下降中，如果梯度骤增，用大梯度更新参数会使得其远离最优点。梯度修剪通过将梯度的模限制在一个区间内来缓解该问题。主要有两类修剪方式[17]。

一类是按值修剪。给定区间$[a,b]$，如果参数的梯度超过b，将其设置为b；如果参数的梯度小于a，将其设置为a，见公式（5.11）表述，其中g_t是第t次迭代时参数的梯度。

$$g_t = \max(\min(g_t, b), a) \tag{5.11}$$

一类是按模修剪。按模修剪通过将梯度的模限制为一个给定的阈值b来缓解该问题，见公式（5.12）表述。

$$g_t = \begin{cases} g_t, & \|g_t\|^2 \le b \\ \dfrac{b}{\|g_t\|} g_t, & \|g_t\|^2 > b \end{cases} \tag{5.12}$$

阈值b是超参数，一般设置为一个较小的值就可以取得不错的结果。

5.4 批量归一化

批量归一化是逐层归一化算法的一种。逐层归一化是传统机器学习中的一种数据归一化算法，通过对隐藏层的输入进行归一化，从而使网络的训练更加容易[20]。

给定激活函数$f(\cdot)$，可学习参数W和b，第l层的净输入$z^{(l)}$，第l层神经元的输出见公式（5.13）表述。

$$a^{(l)} = f\left(z^{(l)}\right) = f\left(Wa^{(l-1)} + b\right) \tag{5.13}$$

通过保持净输入$z^{(l)}$的分布一致，我们可以提高优化的效率，例如将$z^{(l)}$归一化为标准正态分布。在实践中，一般在仿射变换后，激活函数前进行归一化操作。我们可以使用标准化将$z^{(l)}$的每个维度归一化为标准正态分布，见公式（5.14）表述。

$$\hat{z}^{(l)} = \frac{z^{(l)} - E\left[z^{(l)}\right]}{\sqrt{var(z^{(l)}) + \varepsilon}} \tag{5.14}$$

这里$E[z^{(l)}]$和$var(z^{(l)})$是指在当前参数下，在整个训练集上，$z^{(l)}$的每个维度的期望和方差。但是在小批量随机梯度下降法中，无法准确地计算$z^{(l)}$的期望和方差。因此，我们只能用小批量样本集近似估计，见公式（5.15）-（5.16）表述。其中K为小批量样本集合的容量，$z^{(k,l)}$为第l层神经元的净输入，$\mu_{\mathcal{B}}$和$\sigma_{\mathcal{B}}^2$为均值和方差。

$$\mu_{\mathcal{B}} = \frac{1}{K}\sum_{k=1}^{K} z^{(k,l)} \tag{5.15}$$

$$\sigma_{\mathcal{B}}^2 = \frac{1}{K}\sum_{k=1}^{K}\left(z^{(k,l)} - \mu_{\mathcal{B}}\right) \odot \left(z^{(k,l)} - \mu_{\mathcal{B}}\right) \tag{5.16}$$

经过标准归一化后，$z^{(l)}$的取值会集中在 0 附近，该取值区间是一些激活函数的近似线性变换区间，削弱了神经网络的非线性能力。因此，我们附加一个缩放和平移变化操作来修正取值区间，见公式（5.17）表述。

$$\hat{z}^{(l)} = \frac{z^{(l)} - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \varepsilon}} \odot \gamma + \beta \tag{5.17}$$

这里γ和β分别是缩放和平移参数。当$\gamma = \sqrt{\sigma_{\mathcal{B}}^2}, \beta = \mu_{\mathcal{B}}$时，$\hat{z}^{(l)} = z^{(l)}$。批量归一化可以作为一个神经层，作用在激活函数之前，见公式（5.18）表述。批量归一化包含了

平移变换，因此仿射变换不再需要偏置参数。

$$\alpha^{(l)} = f\left(BN_{\gamma,\beta}\left(z^{(l)}\right)\right) \tag{5.18}$$

需要注意的是，小批量样本的均值和方差是变化的，在计算梯度时需要考虑该影响。一般我们可以用移动平均代替计算。

批量归一化不仅可以提高优化效率，也能起到正则化算法的作用，使得模型不会在某个特定样本上过拟合。

5.5 实验分析

(1) Xavier 初始化

实验中 GCN 采用 tanh 激活函数，所以需要将方差乘以一个缩放因子。[2,4,6,8]层 GCN 的准确率见表 5.1，其中 GCN（Xa）表示使用了 Xavier 初始化的 GCN。可以看到，使用了 Xavier 初始化后，GCN 的性能有一定提升。即使是浅层的 GCN，也得益于恰当的初始值，分类性能有所增强。但是当层数达到 16 层时，GCN 的性能大幅下降，这时 Xavier 初始化起到的作用有限。

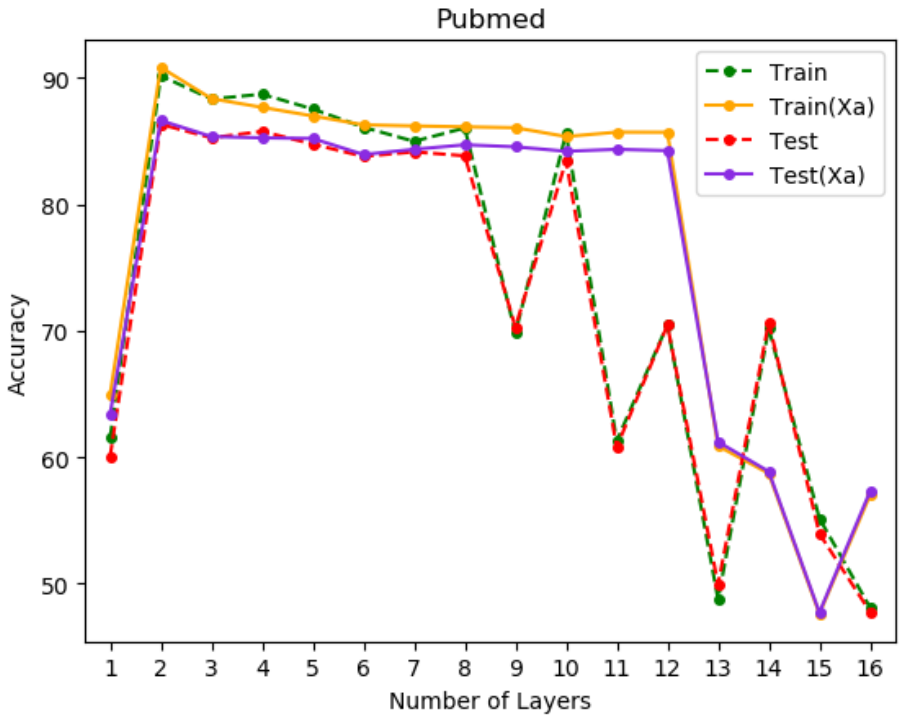

图 5.1 Pubmed 上 Xavier 初始化算法的实验结果

同样地，我们在规模最大的 Pubmed 数据集上实验了[1,16]层的 GCN，见图 5.1。总体而言，Xavier 初始化起到了一定的效果。但是注意到在[13,16]层，即使使用了 Xavier 初始化，GCN 的性能仍会骤降。并且在该区间内，GCN 的性能发生了抖动，这可能是

由于 Xavier 初始化本身的随机性经过了 GCN 多层放大。

Xavier 初始化是一种比较实用且常用的工程技巧，在其他实验中，我们同样采用它作为一种辅助手段。

表 5.1 Xavier 初始化算法的实验结果

数据集	模型	2 层	4 层	8 层	16 层
Cora	GCN	86.35	83.13	86.35	30.92
	GCN(Xa)	87.15	89.16	85.14	30.92
Citeseer	GCN	76.89	74.06	71.7	65.28
	GCN(Xa)	79.25	76.89	75.94	64.32
Pubmed	GCN	86.56	85.5	84.03	44.02
	GCN(Xa)	86.41	85.8	83.37	54.87

(2) 梯度修剪

实验中采用按模修剪，使用L_2范数，阈值b设置为 2。[2,4,6,8]层 GCN 的准确率见表 5.2，其中 GCN（GC）表示使用了梯度修剪的 GCN。可以看到，对于浅层 GCN，此时不存在梯度消失/爆炸问题，梯度修剪会导致信息损失，因此 GCN 的性能有所下降。当层数达到 16 层时，梯度消失/爆炸问题较明显，梯度修剪的增益性有所体现。

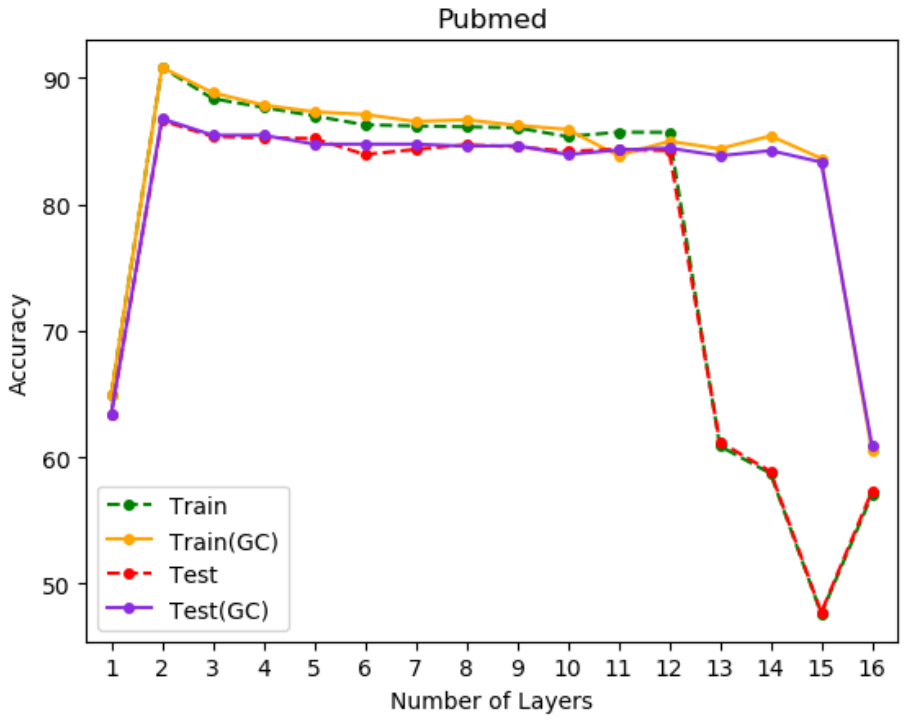

图 5.2 Pubmed 上梯度修剪算法的实验结果

同样地，我们在规模最大的 Pubmed 数据集上实验了[1,16]层的 GCN，见图 5.2。不使用梯度修剪时，层数达到 12 层时训练集和测试集的准确率都发生了骤降，此时可能发生了严重得梯度消失/爆炸问题。使用了梯度修剪后，GCN 的性能相对有了大

幅提升，进一步验证了梯度消失/爆炸问题的存在。GCN 层数较少时，该问题不明显，梯度修剪的作用有限。当层数为 16 层，即使使用了梯度修剪，GCN 的性能还是骤降，可能还存在其他因素限制着 GCN 加深。

表 5.2 梯度修剪算法的实验结果

数据集	模型	2 层	4 层	8 层	16 层
Cora	GCN	87.15	89.16	85.14	30.92
	GCN(GC)	87.15	85.94	85.54	69.08
Citeseer	GCN	79.25	76.89	75.94	64.32
	GCN(GC)	76.42	75.94	72.64	71.23
Pubmed	GCN	86.41	85.8	83.37	54.87
	GCN(GC)	86.87	85.6	84.43	60.9

(3) 批量归一化

实验中，ε的值设置为10^{-5}，移动平均的动量值设置为 0.1，缩放和平移变量为可学习参数。由于 GCN 的几个基准数据集规模都比较小，所以实际上计算的是整个训练集上的均值和方差。[2,4,6,8]层 GCN 的准确率见表 5.3，其中 GCN（BN）表示使用了批量归一化的 GCN。可以看到，在 Cora 和 Citeseer 数据集上，GCN 层数较少时，批量归一化产生了负面影响。在 Pubmed 数据集上，批量归一化的表现最好。当层数达到 16 层时，批量归一化在所有数据集上都对 GCN 有增益。

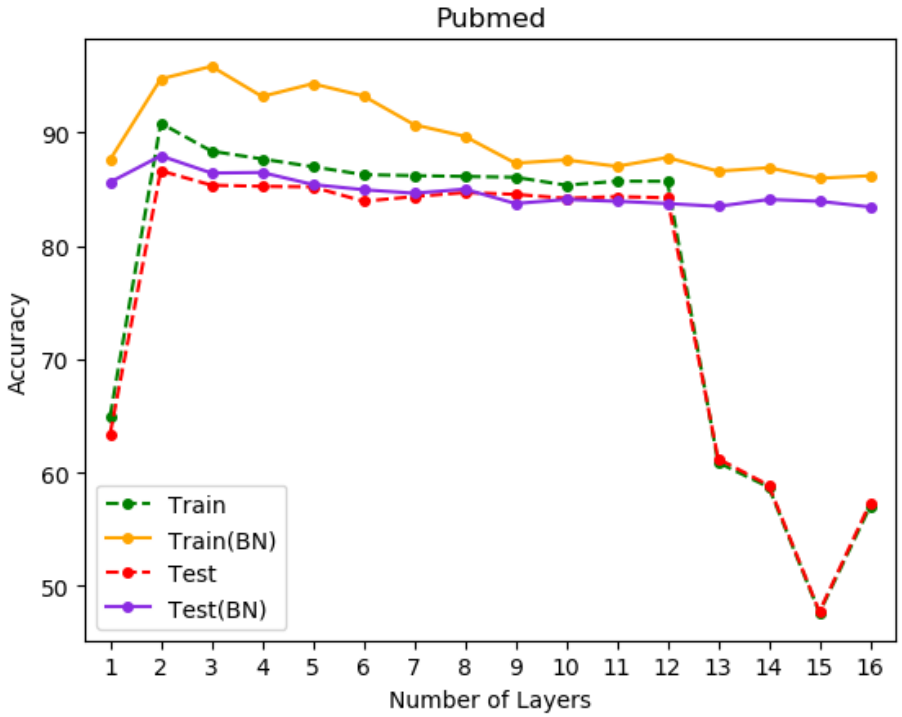

图 5.3 Pubmed 上批量归一化算法的实验结果

同样地，我们在规模最大的 Pubmed 数据集上实验了[1,16]层的 GCN，见图 5.3。采

用了批量归一化后，准确率的走势表现得非常好。即使当层数达到 16 层时，GCN 的性能也没有骤降。在第 6 章中，我们会在多个数据集上，进一步探究批量归一化。实验表明，当层数进一步加深时，使用了批量归一化的 GCN 在多个数据集上性能仍会骤降。此处在 Pubmed 上表现较好的原因是，引用网络结点间连接比较稀疏，过平滑不是特别严重。而批量归一化隐含数据增强的效果，在一定程度上起到了图数据预处理的作用，对过平滑有一些缓解。

表 5.3 梯度修剪算法的实验结果

数据集	模型	2 层	4 层	8 层	16 层
Cora	GCN	87.15	89.16	85.14	30.92
	GCN(BN)	77.51	77.51	81.93	85.54
Citeseer	GCN	79.25	76.89	75.94	64.32
	GCN(BN)	67.92	61.79	70.28	73.58
Pubmed	GCN	86.41	85.8	83.37	54.87
	GCN(BN)	87.73	85.45	83.92	83.57

5.6 本章小结

本章首先介绍了梯度消失/爆炸问题的具体含义，接着介绍了三种常用的缓解该问题的算法：Xavier 初始化，梯度修剪和批量归一化，并在 GCN 上做了一些实验。实验表明，梯度消失/爆炸也是限制 GCN 加深的一个因素，传统的几种算法可以缓解该问题。但是当层数增加到一定程度时，GCN 性能仍然会骤降，限制 GCN 加深的主要因素不是梯度消失/爆炸问题。

第 6 章　面向过平滑的算法

过平滑是限制图卷积神经网络加深的特有问题，本章首先对该问题进行了理论分析，接着设计了实验对理论进行验证，然后从四个角度提出了缓解算法：基于图数据预处理的算法、基于控制邻居权重的算法、基于平衡局部全局的算法和基于增强自身特征的算法，最后在 9 个数据集上进行了实验。

6.1　问题定义

GCN 可以分为两个步骤。首先对结点特征进行图卷积操作，接着再进行一次线性变换操作，其中图卷积是性能提升的关键。我们定义结点特征的每个通道的拉普拉斯平滑见公式（6.1）。

$$\hat{y}_i = (1-\gamma)x_i + \gamma\sum_j \frac{\tilde{a}_{ij}}{d_i}x_j \tag{6.1}$$

其中$\tilde{a}_{ij}$是添加了自循环的邻接矩阵$\tilde{A}=A+I$的分量，$0<\gamma<1$是平衡结点自身特征和邻居特征的权重参数。我们可以将公式（6.1）写成矩阵形式，见公式（6.2）。

$$\hat{Y} = X - \gamma\tilde{D}^{-1}\tilde{L}X = \left(I-\gamma\tilde{D}^{-1}\tilde{L}\right)X \tag{6.2}$$

这里$\tilde{L}=\tilde{D}-\tilde{A}$，$\tilde{D}^{-1}\tilde{L}$是归一化拉普拉斯矩阵。假设不使用自身特征，令$\gamma=1$，则$\hat{Y}=\tilde{D}^{-1}\tilde{A}X$，我们得到拉普拉斯平滑的标准形式。如果用对称归一化拉普拉斯矩阵代替归一化拉普拉斯矩阵，我们进一步得到 GCN，所以 GCN 是一种特殊的拉普拉斯平滑，即对称拉普拉斯平滑。由于邻接矩阵添加了自循环，拉普拉斯平滑仍然包含结点自身特征。通过计算自身特征和邻居特征的基于结点度数的加权平均，我们得到结点特征的新的表示。

连通分量的指示向量的定义见公式（6.3）表述，该指示向量描述结点j是否在分量C_i中。

$$1_j^{(i)} = \begin{cases} 1, v_j \in C_i \\ 0, v_j \notin C_i \end{cases} \tag{6.3}$$

对于任意的$\alpha\in(0,1], w\in R^n$，我们有关于图卷积的结论[4]，见公式（6.4）-（6.5）表述。其中$\theta_1\in R^k, \theta_2\in R^k$。

$$\lim_{m\to+\infty}(I-\alpha L_{rw})^m w = \left[1^{(1)}, 1^{(2)}, \dots, 1^{(k)}\right]\theta_1 \tag{6.4}$$

$$\lim_{m\to+\infty}\left(I-\alpha L_{sym}\right)^m w = D^{-\frac{1}{2}}\left[1^{(1)}, 1^{(2)}, \dots, 1^{(k)}\right]\theta_2 \tag{6.5}$$

可以看到，随着归一化拉普拉斯平滑的不断使用，图的每个连通分量内结点的特征会收敛到同一个值。对于对称归一化拉普拉斯平滑，该值与结点度数的二分之一次幂成

正比。如果每个类簇恰好是一个连通分量，那么这将有利于分类任务。但是事实上实验用到的图数据集，不同类簇之间是连通的，甚至整张图都是连通的，而重复使用拉普拉斯平滑可能会混合不同类簇中的结点特征使得它们难以被区分，随着层数增加最终所有结点都收敛到相似的值完全无法区分。

6.2 实验验证

6.2.1 批量归一化验证

在第 5 章中，我们在 Pubmed 数据集上用批量归一化算法实验了[1,16]层的 GCN，实验结果表明 GCN 的准确率没有发生骤降。考虑到批量归一化隐含的数据增强效果，以及引用网络结点间的连接比较稀疏，过平滑问题不是很严重，我们又在 Chameleon 数据集上用批量归一化实验了[1,16]层的 GCN，见图 6.1。该数据集结点间的连接很密集，过平滑问题比较严重。

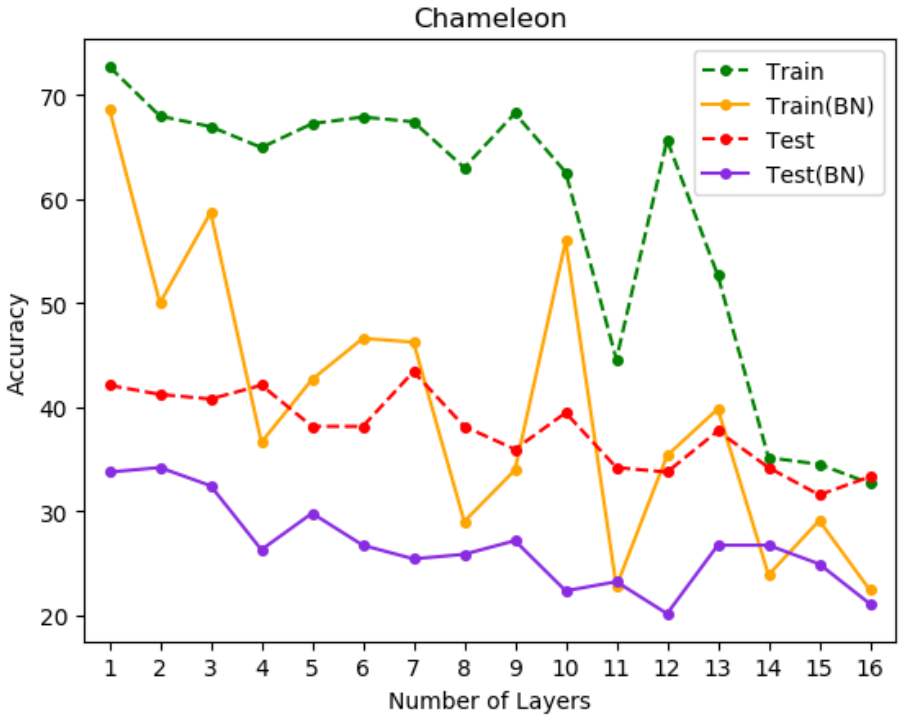

图 6.1　Chameleon 上批量归一化算法的实验结果

可以看到，即使使用了批量归一化算法，训练集和测试集的准确率也都会在波动中大幅下降。此外，该算法还起到了负面作用， 损害了 GCN 的性能。

6.2.2 过平滑理论验证

过平滑问题是由于重复使用拉普拉斯平滑，不同类簇中的结点特征发生混合而难以区分。同一类簇的内部结点倾向于连接比较密集，越往中心连接越密集，而边缘结点连

接比较稀疏，不同类簇间连接也比较稀疏。当层数较少时混杂的结点也较少，此时性能缓慢下降，当层数到达某一阈值时，平滑范围触及了连接密集区域，混杂的结点骤增，因而性能急剧下降。当层数在[1-3]区间时，浅层学习到的结构信息匮乏为主要问题，增加层数可以学习到更多结构信息，因此增加层数可以提高性能。我们可以人为地将图数据集按照标签类别分割为几个连通分量，根据对过平滑的分析，此时随着层数增加，连通分量内的结点收敛到各自的值，准确率会持续上升直到趋于稳定。

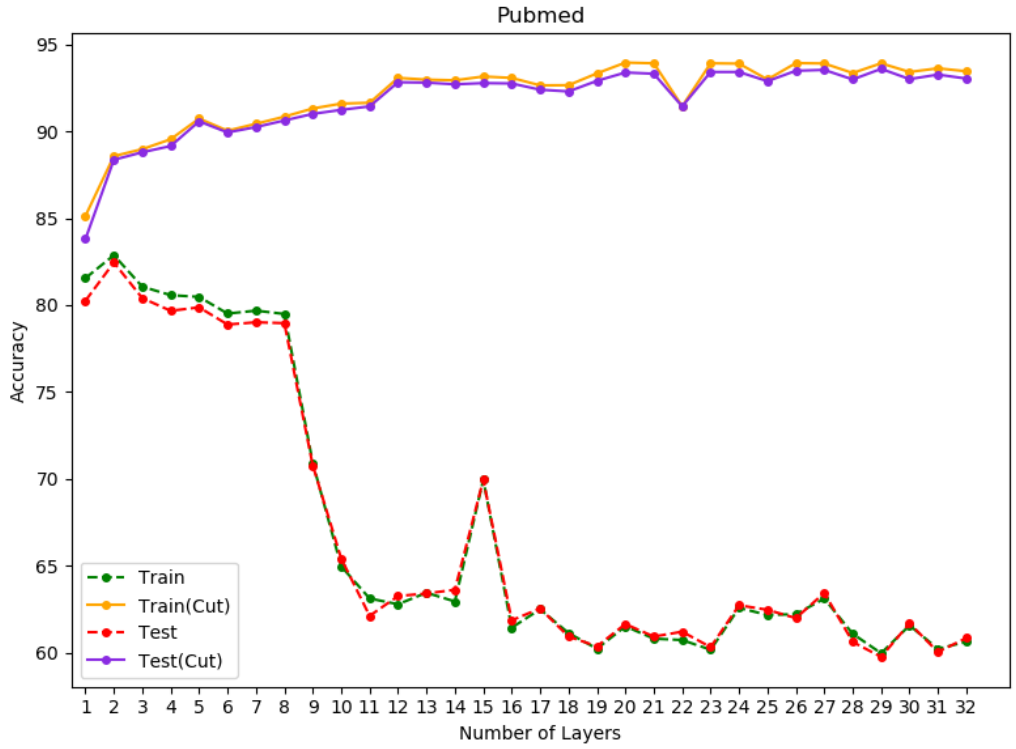

图 6.2 过平滑理论验证

为排除过拟合和梯度消失/爆炸问题的混合影响，我们采用 SGC 模型开展对过平滑问题的研究。SGC 模型是对 GCN 模型的简化，它假设 GCN 层间的非线性不是关键，局部邻居的聚合操作才是关键[21]。通过删除层间的非线性激活函数，只保留分类任务中最终的 softmax 函数，得到的 SGC 模型见公式（6.6）表述。

$$\begin{aligned}\hat{Y} &= softmax\left(\hat{A}\ldots\hat{A}\hat{A}X\Theta^{(1)}\Theta^{(2)}\ldots\Theta^{(K)}\right)\\ &= softmax\left(\hat{A}^{K}X\Theta\right)\end{aligned} \tag{6.6}$$

这里$\hat{A}$是添加了自循环的对称归一化邻接矩阵。可以看到，SGC 模型由两部分组成，一个不含参数的特征提取器$\bar{X}=\hat{A}^{K}X$，一个线性逻辑回归分类器$\hat{Y}=softmax(\bar{X}\Theta)$。由于 SGC 只包含一层可学习参数，$K$表示的是拉普拉斯平滑的次数，$K$的增加并不会导致

过拟合和梯度消失/爆炸问题，因此 SGC 适合用来研究过平滑问题。

对过平滑理论的验证实验结果见图 6.2，其中 Cut 表示进行了图割处理，即去除了不同类簇间的噪声边。可以看到，未做图割处理时，在[1,2]层训练集和测试集的准确率上升，2 层后准确率缓慢下降，8 层后准确率开始骤降，随着层数增加最终趋于稳定。进行图割处理后，训练集和测试集的准确率随着层数的增加持续上升直至趋于稳定。该实验结果与理论分析完全吻合，充分证实了过平滑的成因分析。我们可以根据该分析来提出一些缓解算法。

6.3 基于图数据预处理的算法

根据对过平滑问题的理论分析，只要我们能够去除不同类簇间的噪声边，就能够从根本上解决该问题。理想情况下，不同类簇间完全不连通，也就不存在过平滑问题。

DropEdge 通过随机丢弃一定比例的边，在引用数据集上取得了一定的效果。被丢弃的边集合中包含了一部分噪声边，因此 DropEdge 起到了缓解过平滑的作用。但是实验表明，在密集连接的 Chameleon 等数据集上，DropEdge 反而会降低 GCN 的性能。这是因为 Chameleon 等数据集包含大量的噪声边，基于随机性丢弃边会导致同一类簇间的有效边的损失。如果我们能够根据某种指标，针对性地丢弃一些边，使得被丢弃的边中包含更多的噪声边，那么就能提升 DropEdge 的性能。我们基于两种假设，分别改进了 DropEdge。

一类是基于结点相似度的改进。假设不同类簇间的结点相似度较小，我们可以据此对图数据进行割边，使得更多的噪声边被丢弃。我们采用余弦相似度进行计算，并将计算结果进行$softmax$归一化处理，见公式（6.7）表述。

$$a_{0,j} = \frac{\exp\left(\cos\left(x_0, x_j\right)\right)}{\sum_{k \in \mathcal{N}(v_0)} \exp(\cos(x_0, x_k))} \tag{6.7}$$

这里$\mathcal{N}(v_0)$表示结点v_0的邻居集合，$a_{0,j}$表示注意力权重，描述了结点和邻居的相对重要性。我们将边按照注意力权重排序，按照比例α删除权重值较小的边。在代码实现中，通过将边的权重置为零进行删边操作。

另一类是基于结点度数的改进。假设不同类簇间的结点连接比较稀疏，我们可以据此对图数据进行割边，使得更多的噪声边被丢弃。我们将结点按照度数排序，按照比例α筛选出度数较小的结点，接着在这些结点上按照比例β随机删除边。

6.4 基于控制邻居权重的算法

我们也可以通过控制结点对邻居结点的聚合权重来缓解过平滑问题。理想情况下，

A 类簇的结点对位于 B 类簇的邻居结点的聚合权重为 0，不同类簇的结点不会产生混合，过平滑问题得到解决。我们在基于注意力机制的 GAT 模型上做了一点改进，提高了模型的运行速度，同时保证性能不受影响。GAT 利用参数向量学习结点和邻居间的相对重要性[22]，见公式（6.8）表述。

$$a_{0,j}=\frac{\exp\left(LeakyReLU\left(a\left[Wx_0\|Wx_j\right]\right)\right)}{\sum_{k\in\mathcal{N}(v_0)}\exp\left(LeakyReLU\left(a\left[Wx_0\|Wx_k\right]\right)\right)} \tag{6.8}$$

这里a是可学习的参数向量，用于学习相对重要性。W是可学习的参数矩阵，用于对输入特征做线性变换，||是向量拼接操作。由于 GAT 包含许多可学习参数，训练速度相对来说比较慢。我们用余弦相似度直接计算相对重要性，代替参数向量a和激活函数$LeakyReLU(\cdot)$，见公式（6.9）表述。

$$a_{0,j}=\frac{\exp\left(\cos\left(Wx_0,Wx_j\right)\right)}{\sum_{k\in\mathcal{N}(v_0)}\exp\left(\cos\left(Wx_0,Wx_k\right)\right)} \tag{6.9}$$

6.5 基于平衡局部全局的算法

近距离的邻居比远距离的邻居更重要，并且远距离的邻居容易导致过平滑。如果 GCN 能够平衡好局部与全局的信息，就能在一定程度上缓解过平滑问题。我们可以借鉴 CNN 中的残差网络 ResNet 和密集网络 DenseNet[23]的结构来改进 GCN，见图 6.3。

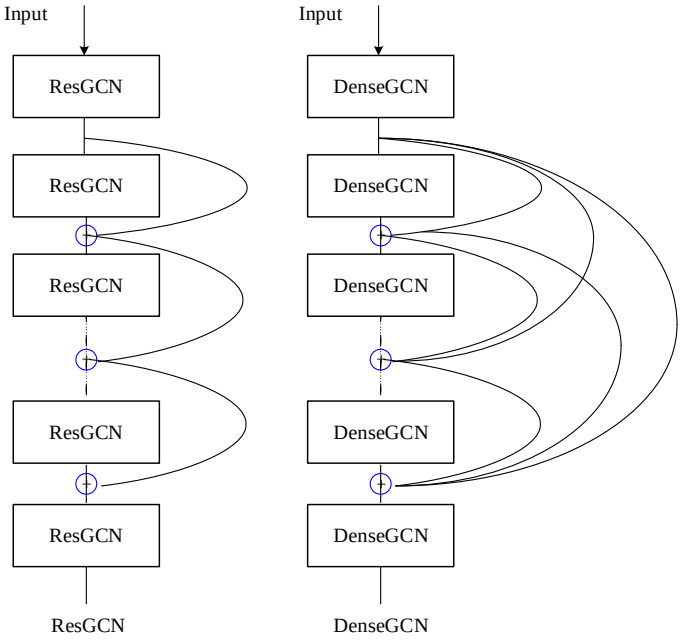

图 6.3 ResGCN、DenseGCN 网络结构

提出残差连接的初衷，是让模型的内部结构至少有恒等映射的能力，以保证在堆叠网络的过程中，网络不会因为继续堆叠而产生退化[2]。此外，残差连接还有其他一些作用。即使批量归一化处理后梯度的模稳定在正常范围，但是梯度的相关性会随着层数增

加持续衰减，而残差连接可以有效较少这种相关性的衰减[24]。另外，浅层特征具有高分辨率低级语义，深层特征具有高级语义低分辨率，而残差连接可以实现不同分辨率特征的组合[25]。

我们可以将残差连接应用到 GCN，从而组合高低层不同范围的邻居信息。进一步地，我们可以使用密集连接加强该作用。此外，相比较 JK-Net，残差连接和密集连接都缓解了较深层产生的输出仍然存在不同类簇间的结点混合的问题。

我们在残差连接上引入了权重参数，该参数平衡了局部和全局的相对重要性，α的值越大，说明局部性越重要，见公式（6.10）表述。

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) + \alpha H^{(l)} \tag{6.10}$$

6.6 基于增强自身特征的算法

一个结点的信息主要由两部分组成，自身信息和结构信息。其中自身信息由结点特征体现，结构信息由邻居结点体现。然而，随着层数加深，越来越多的邻居结点被聚合，结点自身的信息越来越匮乏。为了缓解该问题，我们对 GCN 做了一些改进，见公式（6.11）表述，该算法也可用于 SGC 等模型。

$$H^{(l+1)} = \sigma\left(\left((1-\alpha)\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)} + +\alpha H^{(0)}\right)W^{(l)}\right) \tag{6.12}$$

该公式相当于在每一层引入了输入层的带权重的跳接，见图 6.4。从随机游走的角度来看，α表示游走过程中回退到出发结点的概率，也起到了平衡局部和全局的作用。

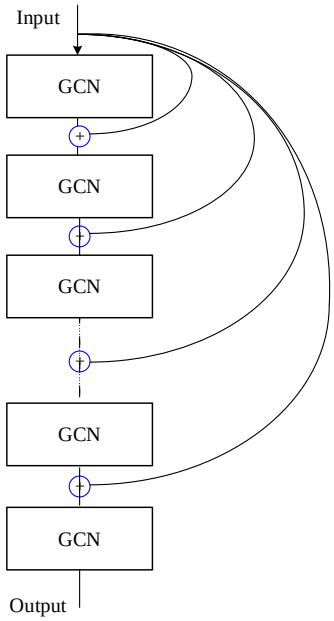

图 6.4 SelfNet 网络结构

6.7 实验分析

(1) 基于图数据预处理的算法

实验中采用基于结点相似度的改进算法，将原算法 DropEdge 作为对照组之一，分别在 SGC 和 GCN 模型上进行了实验。我们用网格搜索对丢弃比例超参数α进行了优化，搜索空间为[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.05, 0.01]，最终确定了 9 个数据集上各自的最优值，其中原算法 DropEdge 为[0.2, 0.1, 0.05, 0.05, 0.01, 0.01, 0.6, 0.1, 0.01]，基于结点相似度的改进算法为[0.05, 0.05, 0.01, 0.6, 0.6, 0.7, 0.6, 0.5, 0.6]。详细的实验结果见表 6.1。

这里 SGC（DR0）和 SGC（DR1）分别表示采用了原算法 DropEdge 和基于结点相似度的改进算法的 SGC，括号里的数字表示取得最佳准确率的层数。可以看到，DropEdge 对性能的提升有限，而基于结点相似度改进的 DropEdge 却在多个数据集上表现突出，特别是在过平滑比较严重的密集连接数据集上有较大提升，并且取得最佳准确率的层数也有所提高。我们也在 GCN 上进行了实验，实验结果表明，基于结点相似度改进的 DropEdge 对 GCN 也有所增益。此外，GCN 模型比 SGC 模型的整体结果更好，说明非线性变换可以增强学习能力。

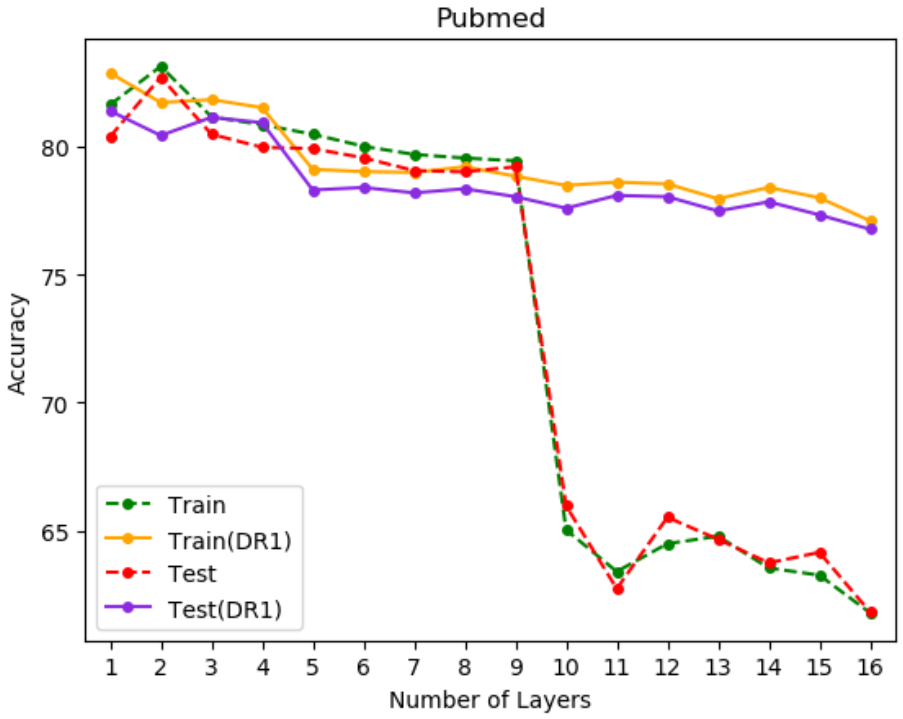

图 6.5 Pubmed 上基于图数据预处理的算法的实验结果

为了排除过拟合和梯度消失/爆炸的混合影响，我们用 SGC 模型在层数区间[1,16]

上进行了实验，见图 6.5。可以看到，采用了基于结点相似度改进的 DropEdge 算法后，过平滑问题得到了很大程度的缓解。

表 6.1 基于图数据预处理的算法的实验结果

模型	SGC	SGC(DR0)	SGC(DR1)	GCN	GCN(DR0)	GCN(DR1)
Cora	84.34(3)	85.54(2)	84.34(3)	87.95(3)	86.75(6)	87.35(4)
Cite.	76.89(2)	75.24(1)	77.36(2)	76.18(2)	76.65(2)	76.65(2)
Pubm.	82.25(1)	82.18(2)	82.3(2)	86.92(2)	87.04(2)	87.2(2)
Cham.	42.98(2)	41.01(2)	45.83(2)	43.2(2)	44.3(2)	46.27(1)
Squi.	28.28(5)	28.31(2)	29.17(2)	27.83(6)	27.64(2)	29.08(2)
Actor	28.51(1)	28.09(1)	34.93(1)	27.63(2)	27.76(2)	33.55(3)
Corn.	26.32(1)	34.21(2)	34.21(4)	26.32(2)	26.32(1)	63.16(3)
Texa.	64.91(2)	65.79(1)	73.68(2)	68.42(2)	63.16(2)	71.05(3)
Wisc.	60.26(2)	59.62(4)	80.77(4)	57.69(2)	57.69(2)	82.69(5)

(2) 基于控制邻居权重的算法

我们在 SGC 和 GCN 上分别对基于控制邻居权重的算法进行了实验，详细的实验结果见表 6.2，其中 SGC（WE）表示使用了该算法的 SGC 模型。可以看到，基于控制邻居权重的算法有一定效果，但是不如基于图数据预处理的算法。

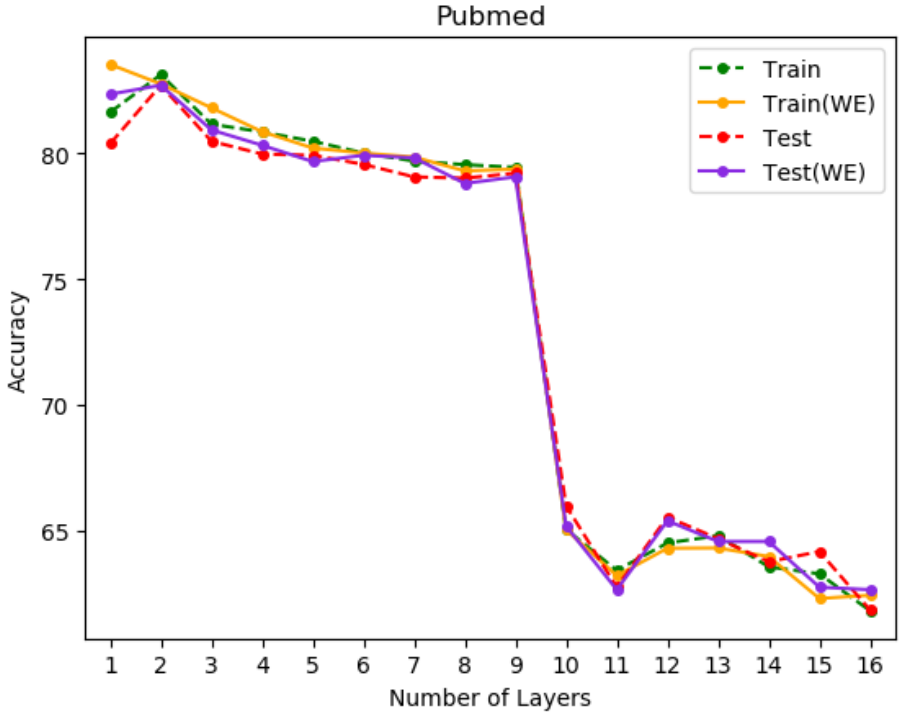

图 6.6 Pubmed 上基于控制邻居权重的算法的实验结果

同样地，我们也用 SGC 模型在 Pubmed 上进行了实验，见图 6.6。可以看到，基于控制邻居权重的算法效果有限，并且也无法缓解过平滑问题，层数达到 9 层后训练集和测试集的准确率都大幅下降。这是因为采用了该算法后，噪声边的权重确实降低了，但

是仍然是一个正值，经过多层聚合叠加后，不同类簇的结点特征还是发生了混合。

表 6.2 基于控制邻居权重的算法的实验结果

模型	SGC	SGC(WE)	GCN	GCN(WE)
Cora	84.34(3)	85.34(3)	87.95(3)	87.55(2)
Cite.	76.89(2)	74.76(1)	76.18(2)	76.42(2)
Pubm.	82.25(1)	82.96(2)	86.92(2)	87.22(3)
Cham.	42.98(2)	45.83(4)	43.2(2)	46.49(3)
Squi.	28.28(5)	28.98(4)	27.83(6)	29.17(1)
Actor	28.51(1)	33.22(1)	27.63(2)	33.03(1)
Corn.	26.32(1)	26.32(1)	26.32(2)	26.32(1)
Texa.	64.91(2)	68.42(1)	68.42(2)	68.42(2)
Wisc.	60.26(2)	63.46(1)	57.69(2)	59.85(3)

(3) 基于平衡局部全局的算法

我们用 GCN 模型实验了残差连接，带权重的改进残差连接和密集连接，分别用 GCN（RES0）、GCN（RES1）和 GCN（DEN）表示。通过网格搜索算法对权重超参数进行了优化，搜索空间为[0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5]，最后在 9 个数据集上的优化值为[0.5, 1.5, 1.5, 2.5, 1, 2, 3, 2, 4]。详细的实验结果见表 6.3。可以看到，带权重的改进残差连接效果最好，其次是原始残差连接。密集连接几乎没有效果，这可能是因为对多层输出拼接做线性变换引入过多参数，不足以学习到局部和全局的平衡信息。

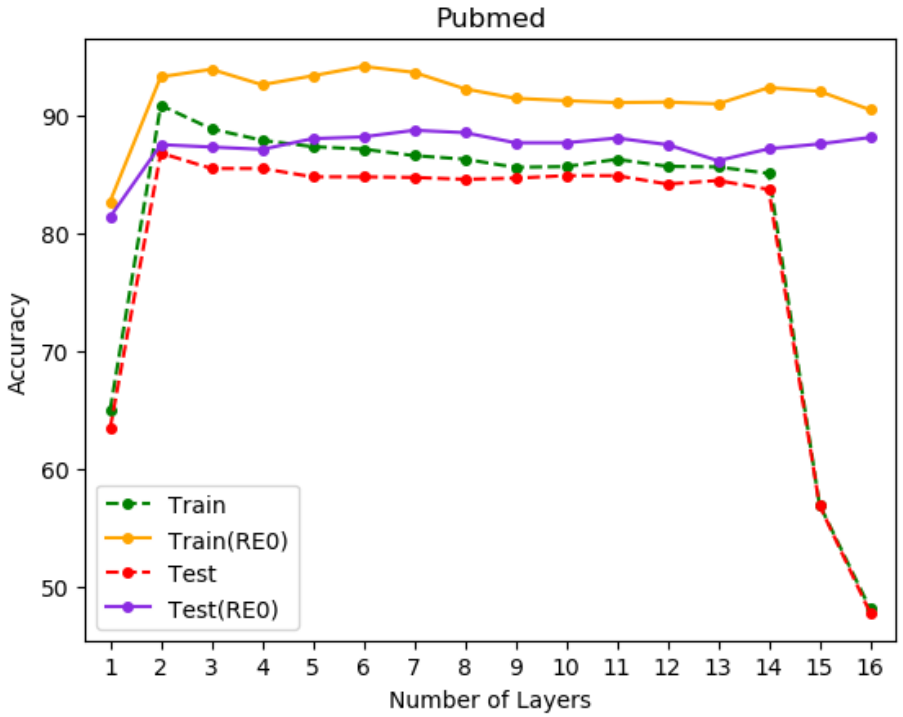

图 6.7 Pubmed 上基于平衡局部全局的算法的实验结果

我们用 GCN 在 Pubmed 上实验了残差连接，见图 6.7。可以看到，残差连接的表现

非常好，在每层上都有提升，同时随着层数增加，性能还会缓慢上升。残差连接能够很好地学习局部与全局信息。

表 6.3 基于平衡局部全局的算法的实验结果

模型	GCN	GCN(RES0)	GCN(RES1)	GCN(DEN)
Cora	87.95(3)	87.15(6)	87.55(3)	85.74(2)
Cite.	76.18(2)	76.89(2)	78.07(2)	76.65(2)
Pubm.	86.92(2)	88.18(5)	88.59(5)	86.82(6)
Cham.	43.2(2)	46.49(1)	49.12(1)	43.64(4)
Squi.	27.83(6)	30.81(8)	30.71(8)	27.83(7)
Actor	27.63(2)	33.88(1)	34.54(1)	26.71(2)
Corn.	26.32(2)	34.21(2)	60.53(2)	26.32(2)
Texa.	68.42(2)	71.05(1)	78.95(4)	68.42(8)
Wisc.	57.69(2)	65.38(1)	75.0(2)	59.62(2)

（4）基于增强自身特征的算法

实验中采用网格搜索进行超参数寻优，搜索空间为[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.05, 0.01]，在 9 个数据集上的搜索结果为[0.2, 0.4, 0.2, 0.4, 0.9, 0.9, 0.9, 0.8, 0.7]。详细的实验结果见表 6.4。其中 SGC（SE）表示使用了基于增强自身特征的算法的 SGC。可以看到，该算法对 SGC 和 GCN 模型都有较好的增益效果。

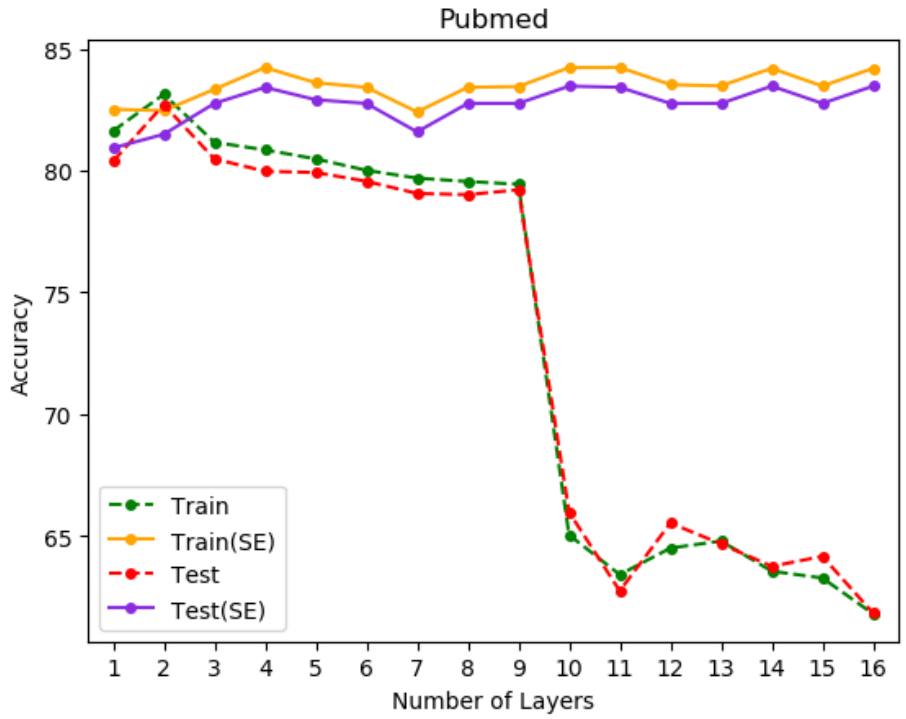

图 6.8 Pubmed 上基于增强自身特征的算法的实验结果

我们也在 Pubmed 上用 SGC 实验了该算法，见图 6.8。可以看到，该算法和残差连接一样表现出色，训练集和测试集的准确率都随着层数持续波动上升。不仅缓解了过平

滑问题，同时整体性能也有提高。

表 6.4 基于增强自身特征的算法的实验结果

模型	SGC	SGC(SE)	GCN	GCN(SE)
Cora	84.34(3)	84.34(6)	87.95(3)	87.75(3)
Cite.	76.89(2)	78.3(1)	76.18(2)	77.12(2)
Pubm.	82.25(1)	83.22(2)	86.92(2)	86.71(2)
Cham.	42.98(2)	48.25(5)	43.2(2)	43.86(3)
Squi.	28.28(5)	30.9(4)	27.83(6)	30.04(2)
Actor	28.51(1)	37.11(1)	27.63(2)	32.43(2)
Corn.	26.32(1)	26.32(1)	26.32(2)	50.0(1)
Texa.	64.91(2)	68.42(2)	68.42(2)	71.05(3)
Wisc.	60.26(2)	65.38(1)	57.69(2)	65.38(1)

6.8 本章小结

本章首先对过平滑问题进行了理论分析，接着精心设计实验验证了该理论，然后基于理论分析从不同角度提出了缓解算法：基于图数据预处理的算法、基于控制邻居权重的算法、基于平衡局部全局的算法和基于增强自身特征的算法，最后对这些算法进行了充分的实验。实验结果表明，以上算法都有一定效果，其中基于结点相似度的改进 DropEdge，带权重的残差连接表现最突出。

第 7 章 总结与展望

7.1 本文总结

本文通过理论分析和实验验证相结合的方式，系统地对图卷积神经网络无法加深这一问题开展了研究。

文章首先介绍了几种主要的深度图卷积神经网络模型，阐述了它们的优缺点以及本文所做的改进。

接着针对过拟合问题，从理论角度进行了分析，在图卷积神经网络上引入了三种正则化算法：权重衰减、提前终止和丢弃法，并在引用数据集上进行了实验。实验结果表明，过平滑是限制图卷积神经网络加深的一个因素，但不是主要因素，传统的正则化算法在该问题上对 GCN 也有效。本文将提前终止作为实验的一种辅助手段，以节省不必要的计算开销。

然后针对梯度消失问题，从理论角度进行了分析，在图卷积神经网络上引入了三种传统算法：Xavier 初始化、梯度修剪和批量归一化，同样在引用数据集上进行了实验。实验结果表明，梯度消失也不是限制图卷积神经网络加深的主要因素，传统的几种算法在该问题上对 GCN 也有效。由于 Xavier 初始化的有效性，本文将其作为 GCN 的固定配置。值得注意的是，批量归一化在引用数据集上表现突出，在[1,16]层数区间内保持了稳定的性能。考虑到引用数据集的连接稀疏性，本文引入了几个密集连接的数据集，在共计 9 个数据集上开展后续实验研究。

最后针对过平滑问题，也从理论角度进行了分析，并精心设计了验证实验。本文采用 SGC 模型进行过平滑的实验研究，避免了过拟合和梯度消失问题的干扰。从图数据预处理的角度，基于结点相似度对 DropEdge 进行了改进。从控制邻居权重的角度，基于余弦相似度对 GAT 进行了改进。从平衡局部全局的角度，在 GCN 上引入了 CNN 中的残差连接和密集连接，并提出了带权重的残差连接以进一步强调局部全局。从增强自身特征的角度，在网络的每一层引入了输入层的跳接，形成了新的网络结构。实验结果表明，过平滑是限制图卷积神经网络加深的主要因素，除了基于余弦相似度的改进 GCN，本文提出的几种算法都能较好地缓解该问题。

7.2 下一步工作

尽管本文引入或提出的算法在多个数据集上取得了较好的表现，但是由于图神经网络的基准数据集规模较小，因此实验结果对算法表现的区分度还不够。最近有研究者新

提出了几个中等规模的图基准数据集，后续可以在这些数据集上进一步开展实验。此外，个别算法也存在着不足之处。基于结点相似度改进的 DropEdge 在密集连接数据集上表现突出，然而在稀疏连接的引用数据集上提升有限，可以寻找其他标准来进行割边，以进一步提高噪声边被丢弃的概率。

参考文献

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

[2] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[3] Xu K, Li C, Tian Y, et al. Representation learning on graphs with jumping knowledge networks[J]. arXiv preprint arXiv:1806.03536, 2018.

[4] Li Q, Han Z, Wu X M. Deeper insights into graph convolutional networks for semi-supervised learning[C]//Thirty-Second AAAI Conference on Artificial Intelligence. 2018.

[5] Xu K, Hu W, Leskovec J, et al. How powerful are graph neural networks?[J]. arXiv preprint arXiv:1810.00826, 2018.

[6] Klicpera J, Bojchevski A, Günnemann S. Predict then propagate: Graph neural networks meet personalized pagerank[J]. arXiv preprint arXiv:1810.05997, 2018.

[7] 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.

[8] Abu-El-Haija S, Kapoor A, Perozzi B, et al. N-gcn: Multi-scale graph convolution for semi-supervised node classification[J]. arXiv preprint arXiv:1802.08888, 2018.

[9] Huang B, Carley K M. Residual or gate? towards deeper graph neural networks for inductive graph representation learning[J]. arXiv preprint arXiv:1904.08035, 2019.

[10] Li G, Müller M, Thabet A, et al. Can GCNs Go as Deep as CNNs?[J]. arXiv preprint arXiv:1904.03751, 2019.

[11] Rong Y , Huang W , Xu T , et al. DropEdge: Towards Deep Graph Convolutional Networks on Node Classification[J]. 2019.

[12] Luan S, Zhao M, Chang X W, et al. Break the Ceiling: Stronger Multi-scale Deep Graph Convolutional Networks[C]//Advances in Neural Information Processing Systems. 2019: 10943-10953.

[13] Zhao L, Akoglu L. PairNorm: Tackling Oversmoothing in GNNs[J]. arXiv preprint arXiv:1909.12223, 2019.

[14] Hoang N T, Maehara T. Revisiting graph neural networks: All we have is low-pass filters[J]. arXiv preprint arXiv:1905.09550, 2019.

[15] Pei H, Wei B, Chang K C C, et al. Geom-gcn: Geometric graph convolutional networks[J]. arXiv preprint arXiv:2002.05287, 2020.

[16] Géron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems[M]. O'Reilly Media, 2019.

[17] 邱锡鹏. 神经网络与深度学习[M]. 第 1 版. 北京：机械工业出版社，2020.

[18] Srivastava N, Hinton G, Krizhevsky A, et al., 2014. Dropout: A simple way to prevent neural networks from overfitting[J]. The Journal of Machine Learning Research, 15(1):1929-1958.

[19] Glorot X, Bengio Y, 2010. Understanding the difficulty of training deep feedforward neural networks[C]//Proceedings of International conference on artificial intelligence and statistics. 249-256.

[20] Ioffe S, Szegedy C, 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift[C]//Proceedings of the 32nd International Conference on Machine Learning. 448-456.

[21] Wu F, Zhang T, Souza Jr A H, et al. Simplifying graph convolutional networks[J]. arXiv preprint arXiv:1902.07153, 2019.

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

[23] Huang G, Liu S, Van der Maaten L, et al. Condensenet: An efficient densenet using learned group convolutions[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 2752-2761.

[24] Balduzzi D, Frean M, Leary L, et al. The shattered gradients problem: If resnets are the answer, then what is the question?[C]//Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, 2017: 342-350.

[25] Lin T Y, Dollár P, Girshick R, et al. Feature pyramid networks for object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 2117-2125.

致　谢

光阴似箭，日月如梭，四年的本科生活也将在尚未结束的疫情中画上句号。回首这四年来，经历过曲折和坎坷，也收获过喜悦与快乐。许许多多的人曾帮助或指导过我，在我学习和生活的道路上不断地支持我，鼓励我，让我拥有了一个美好而难忘的大学生活，我将永远心存感激。

首先我要感谢东北大学，感谢软件学院，感谢学校给我们提供的良好的教育资源和学习环境，感谢软件学院的老师们辛勤的付出，是你们给我四年的精心教育和指导，谢谢你们的教育为我以后的学习和生活打下了坚实的基础。感谢毕业设计的校内指导老师张伟老师，感谢您在我毕设期间辛勤的付出，对我的论文和相关材料认真的审阅和校对，并给予我细心的指导。

特别地，感谢复旦大学大数据学院的黄增峰老师，很幸运能在黄老师的指导和帮助下进行此次毕业设计。在毕设期间，每当我遇到研究工作中的难题时，黄老师总是耐心指导，给予细致、具体的说明，更给与我极大的鼓励和支持。黄老师扎实的学术功底、认真严谨且一丝不苟的学术作风，不辞劳累的工作态度让我深深地折服与敬佩。

还要感谢我亲爱的朋友们，是你们在生活中给我鼓励，陪我前行。生活中我们有争吵也有欢喜，是你们陪我度过了本科四年中最长的岁月，与你们的一起的时光是我永远珍贵的回忆，与你们的友谊也将是我一生珍惜的情谊。

还要感谢我的前女友，虽然我们最近分手了，但是我从这段关系中学到了很多，在自我反省中快速成长。真诚地祝你今后一切顺利。

最后深深地感谢呵护我成长的父母，感谢你们对我无条件的支持和鼓励，你们是我求学和成长之路上的动力源泉。

衷心地感谢所有帮助和支持过我的人。在人生的新阶段，我也将朝着下一个目标继续努力。