Report

1. 基础知识学习

为理解 paper 涉及的理论，简单入门了前馈神经网络 MLP、卷积神经网络 CNN、循环神经网络 RNN（吴恩达老师的课程），在此基础上初步学习了图神经网络 GNN、图卷积神经网络 GCN（文献、博客和知乎）。

表 1 该部分涉及到的文献

序号	名称	类型	年份	主要内容
1	Graph Neural Networks: A Review of Methods and Applications	arxiv	2018	详细回顾了已有的图神经网络模型，对其应用领域进行了系统的分类，并提出了四个有待进一步研究的问题。
2	A Comprehensive Survey on Graph Neural Networks	arxiv	2019	对 GNNs 进行了分类，回顾了一些最新提出的 GCN 架构，讨论了 GNN 在各个领域的应用，总结了现有算法的开源实现，并提出了几个潜在研究方向。
3	The graph neural network model	IEEE TNN	2009	提出了一种新型的基于不动点理论的神经网络模型--图神经网络(GNN)模型，对已有的神经网络模型进行了拓展，适用于处理可以表示为图的数据。
4	Gated Graph Sequence Neural Networks	ICLR	2016	在之前 GNN 的工作上进行了一些修改以使用门控循环单元和现代优化技术，得到了一类灵活且适用广泛的神经网络模型（GGNN）。
5	Neural Message Passing for Quantum Chemistry	ICML	2017	将已有的模型重新构建为一个成为消息传递神经网络（MPNN）的通用框架，并探索了该框架内的其他有效变体。
6	Inductive Representation Learning on Large Graphs	NIPS	2017	提出一种适用于大规模网络的归纳式学习方法 GraphSAGE，能够为新增结点快速生成 embedding 而无需额外训练过程。
7	Learning Convolutional Neural Networks for Graphs	ICML	2016	提出了一种图序列化的方法，从而将一个图结构的数据转化为 CNN 能够高效处理的结构。

8	Spectral Networks and Locally Connected Networks on Graphs	ICLR	2014	将 CNN 泛化到非欧几里得空间，提出两种分别基于空域和频域的模型。
9	Convolutional neural networks on graphs with fast localized spectral filtering	NIPS	2016	形式化定义了图谱域的卷积公式并优化了计算复杂度，提出了一种高效的图池化方法。

2.1 图神经网络 GNN

(1) 初代 GNN[3]

① 模型

通过迭代方式更新所有结点的隐藏状态，t+1 时刻，结点 v 的隐藏状态更新如下：

$$\mathbf{h}_v^{t+1} = f(\mathbf{x}_v, \mathbf{x}_c o[v], \mathbf{h}_n^t e[v], \mathbf{x}_n e[v])$$

f 为局部变换函数，也是一个压缩映射：不断利用当前时刻邻居结点的隐藏状态作为部分输入生成中心结点下一时刻的隐藏状态，直至每个结点的隐藏状态趋于不变。通过比较两个时刻的 p-范数的差值是否小于指定的阈值判断是否收敛，如下所示：

$$||\mathbf{H}^{t+1}||_2 - ||\mathbf{H}^t||_2 < \epsilon$$

可以用神经网络来拟合复杂函数 f，例如通过一个简单的前馈神经网络实现：

$$\mathbf{h}_v^{t+1} = f(\mathbf{x}_v, \mathbf{x}_c o[v], \mathbf{h}_n^t e[v], \mathbf{x}_n e[v])$$

$$= \sum_{u \in ne[v]} FNN([\mathbf{x}_v; \mathbf{x}_{(u,v)}; \mathbf{h}_u^t; \mathbf{x}_u])$$

压缩映射的定义如下：

$$||F(x) - F(y)|| \leq c||x - y||, 0 \leq c < 1$$

可以推导出 f 为压缩映射的充要条件是 f 的导数（梯度）的范数小于 1，进一步推广到雅克比矩阵的范数小于 1。根据拉格朗日乘子法将有约束优化问题转化为带惩罚项的无约束优化问题：

$$J = Loss + \lambda \cdot \max(\frac{||\partial FNN||}{||\partial \mathbf{h}||} - c, 0), c \in (0, 1)$$

通过如下局部输出函数 g 适应下游任务，与 f 类似，g 也可以由神经网络表达。

$$\mathbf{o}_v = g(\mathbf{h}_v, \mathbf{x}_v)$$

② **学习算法**

A. 前向传播：调用 f 函数 T_N 次，直到 h_v 收敛，将隐藏状态通过 g 函数得到输出，进而计算出损失。

B. 反向传播：迭代地计算 T_N 次梯度，最终求出 f 和 g 对初始隐藏状态 h_v^0 梯度。

C. 梯度下降：根据梯度下降法更新模型参数。

③ **GNN 与 RNN**

A. GNN 以不动点理论为支撑，沿时间展开的长度是动态的，根据收敛条件决定；RNN 沿时间展开的长度即序列本身的长度。

B. GNN 采用 AP 反向传播算法优化，对收敛性有要求；RNN 采用 BPTT 反向传播算法优化，对收敛性没要求。

C. GNN 在隐藏状态收敛后才能输出；RNN 在每个时间步都能输出。

④ **局限性**

A. GNN 没有为边设置可学习的参数，无法通过模型学习到边的特性。

B. 基于不动点理论的 GNN 导致收敛后的结点之间的隐藏状态存在较多信息共享，结点的状态过于光滑，并且结点自身特征信息匮乏。

(2) 门控图神经网络 GGNN[4]

GGNN 不再以不动点理论为基础，无需迭代至收敛条件，迭代固定步长即可，优化算法从 AP 转向 BPTT，同时借鉴了 GRU 的设计，改善了 GNN 的局限性。

① **模型**

GGNN 同样包括状态更新与输出两个过程，与 GNN 的主要区别在状态更新阶段。GGNN 的状态更新函数如下：

$$\mathbf{h}_v^{t+1} = \mathrm{GRU}(\mathbf{h}_v^t, \sum_{u \in ne[v]} \mathbf{W}_{edge}\mathbf{h}_u^t)$$

除了参考了 GRU 的设计，该函数还针对不同类型的边引入了可学习的参数 W_{edge}，使得 GGNN 能够处理异构图。此外，GGNN 将结点特征作为隐藏状态的初始化。

② **GNN、GGNN、RNN 与多层神经网络**

A. GNN、GGNN 与 RNN 的设计类似，RNN 的优势在于能够处理任意长的序列，但是计算是串行的若干个时间步，所以基于循环的图神经网络在更新隐藏状态时都不够高效。

B. 基于循环的图神经网络每次迭代共享相同的参数；多层神经网络每层的参数不同，可视为一种层次化特征抽取方法。

2.2 图卷积网络 GCN

在图像为代表的欧式空间中，结点的邻居数量都是固定的，并且采用固定大小可学习的卷积核抽取像素的特征。而图上结点的邻居结点数量不固定，图卷积的目的是找到适用于图的可学习卷积核，主要有两种思路：

- 将非欧空间的图转化为欧式空间
- 找到一种可处理不固定邻居结点的卷积核。

(1) 空域卷积

空域卷积与深度学习中卷积的设计理念相似，核心在于聚合邻居结点的信息。

① 消息传递网络 MPNN[5]

MPNN 是空域卷积的一种形式化框架，由两个阶段组成，消息传递阶段和读出阶段。消息传递阶段运行 T 步基于空间的图卷积，卷积算子由消息函数 M_t 和更新函数 U_t 组成：

$$\mathbf{h}_v^{l+1} = U_{l+1}(\mathbf{h}_v, \sum_{u \in ne[v]} M_{l+1}(\mathbf{h}_v^l, \mathbf{h}_u^l, \mathbf{x}_{vu}))$$

该公式与 GGNN 的公式很像，但是 GCN 通过级联的层捕捉邻居结点信息，GNN 通过级联的时间捕捉邻居结点信息。

读出阶段是一个池操作，根据每个结点隐含表示生成整个图的表示，产生的输出可用于 graph-level 任务。

$$\hat{y} = R(h_v^T | v \in G)$$

② 图采样与聚合 GraphSage[6]

MPNN 框架下卷积操作的是整张图，难以处理实际场景中的大规模图。这也是 GraphSage 提出的动机之一。GraphSage 通过采样部分结点进行学习，不需要对整张图同时卷积，但仍需要聚合邻居结点的信息。其过程可以分为三步：

A. 在图中随机采样指定个数的结点，对每个结点随机选择固定数目的邻居结点（也可以是二阶邻居）构成卷积操作的图。
B. 通过 aggregate 函数聚合邻居结点的信息更新采样结点。
C. 计算采样结点处的损失。

GraphSage 状态更新公式如下：

$$\mathbf{h}_v^{l+1} = \sigma(\mathbf{W}^{l+1} \cdot aggregate(\mathbf{h}_v^l, \{\mathbf{h}_u^l\}), \forall u \in ne[v])$$

aggregate 是可以处理变长数据的函数，既可以是不带参数的 max 等，也可以是带参数的 LSTM 等。

③ 图序列化 PATCHY-SAN[7]

PATCHY-SAN 将图结构转换为序列结构，然后直接应用卷积神经网络。难点在于转换过程中需要保持图结构的两个特点：1.同构性质。2.邻居结点的连接关系。PATCHY-SAN 通过如下步骤来解决这两个问题：

A. Node Squenece Selection：基于人为定义的规则（度数大的结点分数高等）对图中结点排序，选取前 w 个结点（不够填充）作为整个图的代表。
B. Neighborhood graph construction：以第一步中 w 个结点作为中心结点，与其邻居结点（也可以是二阶邻居等）构成 w 个团。对每个团中的邻居结点进行排序，选取前 k 个邻居结点（不够填充），即得到 w 个有序团。
C. Graph Noermalization：按照结点顺序将团转换为固定长度（k+1）的序列，再将 w 个团序列按照中心结点顺序拼接成长度为 w*（k+1）的代表整个图的序列。

(2) 频域卷积

频域卷积主要利用图傅里叶变换实现卷积。傅里叶变换的一个重要性质如下：

$$(f*g)(t)=F^{-1}[F[f(t)]\odot F[g(t)]]$$

该等式表明计算 f 与 g 的卷积，可以先将 f 与 g 通过傅里叶变换到频域中相乘，再通过傅里叶逆变换得到最终结果。傅里叶变换如下：

$$\hat{f}(t)=\int f(x)\exp^{-2\pi ixt}dx$$

其中，$e^{-2\pi ixt}$是拉普拉斯算子的特征函数。通过定义图的拉普拉斯矩阵导出其特征向量可以得到图傅里叶变换。图的拉普拉斯矩阵为$L=D-A$，其中 D 为度矩阵，A 为邻接矩阵。如果图是无向的，则 L 是对称矩阵，可以作如下特征分解：

$$L=U\Lambda U^{T}$$

结合傅里叶变换公式及其性质可以的得到如下图的卷积公式：

$$(f*_G g)=U(U^{T}f\odot U^{T}g)=U(U^{T}g\odot U^{T}f)$$

将$U^{T}g$整体作为可学习的卷积核，记为g_θ，则最终图的卷积公式为：

$$o=(f*_G g)_\theta=Ug_\theta U^{T}f$$

① **Spectral CNN**[8]

上文中的g_θ即 Spectral CNN 的卷积核，状态更新公式如下：

$$h_{:,j}^{l+1} = \sigma(U\sum_{i=1}^{d_l}\Theta_{i,j}^{l}U^T h_{:,i}^{l})$$

$$\Theta_{i,j}^{l} = g_\theta = \begin{bmatrix} \theta_1 & \dots & 0 \\ \dots & \dots & \dots \\ 0 & \dots & \theta_N \end{bmatrix}$$

② **ChebNet**[9]

针对 Spectral CNN 需要计算拉普拉斯矩阵所有的特征值和特征向量，研究者提出了 ChebNet，利用切比雪夫多项式加速求解，卷积核的计算方法如下，其中T_k为切比雪夫多项式的第 K 项。

$$g_\theta = \sum_{i=1}^{K}\theta_i T_k(\widetilde{\Lambda}),\ \widetilde{\Lambda} = 2\Lambda/\lambda_{max} - I_N$$

信号 x 的卷积如下，其中$\widetilde{L} = 2L/\lambda_{max} - I_N$。

$$x * Gg_\theta = U(\sum_{i=1}^{K})\theta_i T_k(\widetilde{\Lambda})U^T x$$
$$= \sum_{i=1}^{K}\theta_i T_i(\widetilde{L})x$$

由于避免了特征向量的计算，计算复杂度从$O(N^3)$降到了O(KM)。

2. Predict then Propagate Graph Neural Networks meet Personalized PageRank

2.1 问题背景

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。简单地堆叠层以获取更多邻居结点信息会带来两个问题：

- 聚合次数过多会导致过平滑，丧失了结点的局部特性。
- 堆叠层数过多会导致参数量过大，有可能造成过拟合。

2.2 解决方法

作者首先分析了：k 层 GCN 近似于 k 步随机游走，随着层数增加，GCN 收敛到随机游走的极限分布，该分布是全图的一个性质 ，与随机游走的起点没有关系，不再适合描述邻居结点的信息。

受 Personalized PageRank 启发，作者提出一种新的传播算法，该方法可以平衡局部性和对更大范围邻居信息的需求，从而规避过光滑的问题，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。

(1) Personalized PageRank

普通的 PageRank 计算公式为：

$$\boldsymbol{\pi}_{\text{pr}} = \boldsymbol{A}_{\text{rw}}\boldsymbol{\pi}_{\text{pr}}, \text{ with } \boldsymbol{A}_{\text{rw}} = \boldsymbol{A}\boldsymbol{D}^{-1}$$

Personalized PageRank 将根节点i_x也考虑进来，其迭代公式为：

$$\boldsymbol{\pi}_{\text{ppr}}(\boldsymbol{i}_x) = (1-\alpha)\hat{\tilde{\boldsymbol{A}}}\boldsymbol{\pi}_{\text{ppr}}(\boldsymbol{i}_x)+\alpha\boldsymbol{i}_x,$$

其中：

- $\hat{\tilde{A}} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$为添加了自循环的堆成归一化邻接矩阵。
- $\tilde{A} = A + I$，A 为图的邻接矩阵。
- $\tilde{D}$是$\tilde{A}$的度矩阵。$\mathrm{D} = \mathrm{diag}(d_1, \ldots, d_n)$代表度矩阵，其中$d_i = \sum_j a_{ij}$。

求解上式可得：

$$\boldsymbol{\pi}_{\text{ppr}}(\boldsymbol{i}_x) = \alpha\left(\boldsymbol{I}_n - (1-\alpha)\hat{\tilde{\boldsymbol{A}}}\right)^{-1}\boldsymbol{i}_x$$

进一步可以得到矩阵的表示：

$$\boldsymbol{\Pi}_{\text{ppr}} = \alpha(\boldsymbol{I}_n - (1-\alpha)\hat{\tilde{\boldsymbol{A}}})^{-1}$$

它的每个元素（yx）表示结点 x 对 y 的影响分数大小，并且每个根节点的值都不同。

(2) Personalized propagation of neural predictions (PPNP)

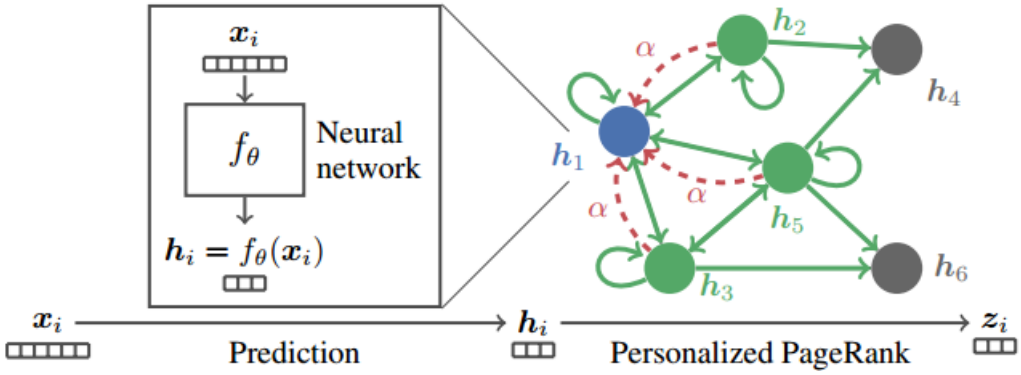
对于半监督结点分类任务，为了利用上述得到的 Personalized PageRank 影响分数，作者提出将高层特征与该分数一起用于生成每个结点的类别概率分布，计算公式如下：

$$\boldsymbol{Z}_{\text{PPNP}} = \text{softmax}\left(\alpha\left(\boldsymbol{I}_n - (1-\alpha)\hat{\tilde{\boldsymbol{A}}}\right)^{-1}\boldsymbol{H}\right), \qquad \boldsymbol{H}_{i,:} = f_\theta(\boldsymbol{X}_{i,:}),$$

其中：

- X 为每个结点的输入特征
- H $\in R^{n\times c}$为每个结点的隐层特征
- 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（允许并行），并且该过程中没有聚合操作。

该计算过程如下图所示：



该设计将神经网络和传播算法分离，神经网络的深度完全独立于传播算法，并且我们可以用任意神经网络生成结点的表示。此外 Personalize PageRank 使得聚合时可以使用更大范围的邻居信息，这在传统的 GCN 中是很难做到的。

从反向传播的角度来看，由于 Personalized PageRank 得分矩阵的存在，梯度会流向许多结点，而不只是传统 GCN 里 k-hop 的邻居范围。

(3) Approximate personalized propagation of neural predictions (APPNP)

直接计算矩阵Π_{ppr}的时间和空间复杂度都很高，作者提出一种近似方法来提升计算效率。H $\in R^{n\times c}$的每一列可以视为所有结点没有归一化的转移分布，从而可以用迭代地方法进行近似，计算过程如下：

$$\begin{aligned}
\boldsymbol{Z}^{(0)} &= \boldsymbol{H} = f_\theta(\boldsymbol{X}),\\
\boldsymbol{Z}^{(k+1)} &= (1-\alpha)\hat{\tilde{\boldsymbol{A}}}\boldsymbol{Z}^{(k)} + \alpha\boldsymbol{H},\\
\boldsymbol{Z}^{(K)} &= \text{softmax}\left((1-\alpha)\hat{\tilde{\boldsymbol{A}}}\boldsymbol{Z}^{(K-1)} + \alpha\boldsymbol{H}\right),
\end{aligned}$$

其中：

- k 是超参数，表示迭代轮数。

- 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可以选择不同的转移概率α。

2.3 实验分析

作者设计了一套相对严谨的评估方法，在不同数据集上从不同角度进行了实验。

- Overall accuracy：PPNP 和 APPNP 在各个数据集上的性能都比其他模型要好。其中 PPNP 由于空间复杂度较高，在 PUBMED 和 MS ACADEMIC 数据集上超出了内存限制。同时作者也比较了各个模型的准确率分布，PPNP，APPNP 和 GAT 的鲁棒性最好。消息传递算法大多对数据划分和权重初始化比较敏感。
- Training time per epoch：PPNP 同样只能用在一般规模数据集上，而无法扩展到大数据集上。PPNP 和 APPNP 的效率均比 GCN 略低（需要更多矩阵操作），但是优于复杂模型 GAT 等。
- Training set size：标签样本较少时，PPNP 和 APPNP 的性能明显高出其他模型，这是由于基于 Personalized PageRank 的传播算法可以使用更大范围的邻居结点信息。
- Number of power iteration steps：可以看到，随着幂迭代次数 K 的上升，GCNs 的性能最终急剧下降，而 APPNP 的性能却逐步上升并最终趋于稳定，这表明基于 Personalized propagation 的传播算法是有效的。
- Teleport probability α：超参数 α 对准确率也会有影响，尽管在每个数据集上最优值不同，但是通过实验发现其分布在[0.05,0.2]上，对于不同的图或任务我们可以设置不同的值。

3. Simplifying Graph Convolutional Networks

3.1 问题背景

传统的机器学习方法的复杂度变化趋势都是从简单到复杂：从线性 Perceptron 到非线性 MLP，从线性图片 filters 到 CNN 都是这个趋势。GCNs 的灵感主要来自最近的一些深度学习方法，因此可能会继承不必要的复杂度和冗余计算。

3.2 解决方法

本文假设 GCN 层之间的非线性不是最关键的，通过反复消除 GCN 层之间的非线性并将得到的函数折叠成一个线性变换来减少 GCNs 的额外复杂度。

(1) 图卷积网络 GCNs

GCNs 和 MLPs 相似，都是通过多层网络学习一个节点的特征向量，最后再经过一个线性分类器进行分类任务。

① 特征传播

不同于 MLP，GCN 每一层的输入都是结点局部邻居的平均值：

$$\bar{\mathbf{h}}_i^{(k)} \leftarrow \frac{1}{d_i+1}\mathbf{h}_i^{(k-1)} + \sum_{j=1}^{n} \frac{a_{ij}}{\sqrt{(d_i+1)(d_j+1)}}\mathbf{h}_j^{(k-1)} \quad (2)$$

公式（2）的更新可以用矩阵表示为：

$$\mathbf{S} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}} \quad (3)$$

其中：

- $\tilde{A} = A + I$，A 为图的邻接矩阵。
- $\tilde{D}$是$\tilde{A}$的度矩阵。$\mathrm{D} = \mathrm{diag}(d_1, \ldots, d_n)$代表度矩阵，其中$d_i = \sum_j a_{ij}$。
- S 表示添加自循环的“归一化”的邻接矩阵。

将公式（2）同时应用到所有结点上，得到如下的稀疏矩阵乘法：

$$\bar{\mathbf{H}}^{(k)} \leftarrow \mathbf{S}\mathbf{H}^{(k-1)} \quad (4)$$

② 特征转换与非线性过渡

局部平滑后，一个 GCN 层相当于一个标准的 MLP。每一层对应一个可学习的权重矩阵$\theta^{(k)}$，作用在平滑处理后的隐藏特征$\bar{H}^{(k)}$上进行线性转换。最后再逐结点应用一个非线性激活函数，如 ReLU 输出特征表示$H^{(k)}$：

$$\mathbf{H}^{(k)} \leftarrow \mathrm{ReLU}\left(\bar{\mathbf{H}}^{(k)}\boldsymbol{\Theta}^{(k)}\right) \quad (5)$$

③ 分类器

与 MLP 相似，对于结点分类任务，使用一个 softmax 分类器预测结点标

签，一个 K 层的 GCN 的所有结点的类别预测如下所示：

$$\hat{\mathbf{Y}}_{\mathrm{GCN}}=\operatorname{softmax}\left(\mathbf{S}\mathbf{H}^{(K-1)}\boldsymbol{\Theta}^{(K)}\right) \quad (6)$$

其中：

- $\hat{Y}\in R^{n\times C}$表示所有结点的预测的类别。
- $\hat{y}_{ic}$表示结点 i 预测为类别 c。
- $\mathrm{softmax}(x)=\exp(x)/\sum_{c=1}^{C}\exp(x_c)$是一个归一化操作。

(2) 简化图卷积 SGC

① 线性化

假设 GCN 层之间的非线性是非关键的，最关键的是局部邻居的平均聚合操作。删除层之间的非线性转换函数（ReLU），只保留最终的 softmax 函数，得到如下线性模型：

$$\hat{\mathbf{Y}}=\operatorname{softmax}\left(\mathbf{S}\ldots\mathbf{S}\mathbf{S}\mathbf{X}\boldsymbol{\Theta}^{(1)}\boldsymbol{\Theta}^{(2)}\ldots\boldsymbol{\Theta}^{(K)}\right) \quad (7)$$

简化如下：

$$\hat{\mathbf{Y}}_{\mathrm{SGC}}=\operatorname{softmax}\left(\mathbf{S}^{K}\mathbf{X}\boldsymbol{\Theta}\right) \quad (8)$$

② 逻辑回归

从公式（8）可以看出，SGC 由两部分组成：

- 一个不含参数的固定的特征提取器：$\bar{X}=S^{K}X$。
- 一个线性逻辑回归分类器：$\hat{Y}=softmax(\bar{X}\theta)$

计算$\bar{X}$不需要权值，可以将该部分作为特征的预处理步骤，整个模型的训练简化为对预处理特征$\bar{X}$的多类逻辑回归。

③ 优化细节

逻辑回归的训练是凸优化问题，可以用任何有效的二阶优化方法或随机梯度下降法进行。在图连通模式足够稀疏的情况下，SGD 可以很自然地应用到大图上，并且 SGC 的训练比 GCN 快得多。

(3) 理论分析（部分细节还没看懂）

作者证明了 SGC 在图谱域上对应于一个固定的滤波器。在原始图上添加自循环，可以有效地缩小底层图的谱，此时 SGC 相当于一个低通滤波器。

① 图上的初步做法

对信号 X 和滤波器 g 进行傅里叶变换后，GCN 的卷积操作为：

$$\mathbf{g}*\mathbf{x}=\theta(\mathbf{I}+\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2})\mathbf{x} \quad (11)$$

将卷积推广到 d 维通道输入中的多个滤波器上，并在层之间添加非线性激活函数，就得到如公式（5）定义的 GCN 传播规则。

② 简化的图卷积和低通滤波器

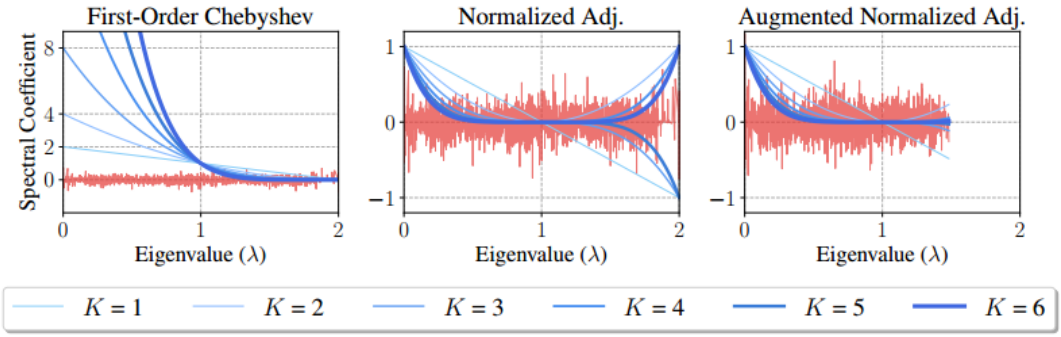
对于一个简单无孤立结点的无向图，令$\tilde{A}=A+\gamma I,\gamma>0$，$\lambda_1$和$\lambda_2$分别是对

称的归一化的拉普拉斯矩阵$\Delta_{sym} = \mathrm{I} - D^{-1/2}AD^{-1/2}$的最小特征值和最大特征值。令$\tilde{\lambda}_1$和$\tilde{\lambda}_1$分别是$\tilde{\Delta}_{sym} = \mathrm{I} - \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$的最小特征值和最大特征值，则：

$$0 = \lambda_1 = \tilde{\lambda}_1 < \tilde{\lambda}_n < \lambda_n \quad (12)$$

可以看出，当$\gamma > 0$时，相当于图中添加了自循环，归一化的拉普拉斯矩阵的最大特征值会变小。

下图描述了在 Cora 数据集上不同传播矩阵下特征值和滤波器系数的变化关系。



其中：

- Normalized Adj：$S_{adj} = D^{-1/2}AD^{-1/2}$；Augmented Normalized Adj：$\tilde{S}_{adj} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$；First-Order Chebyshev：$S_{1-order} = I + D^{-1/2}AD^{-1/2}$。
- $S_{adj} = D^{-1/2}AD^{-1/2}$对应的滤波器$\mathrm{g}(\lambda_i) = (1-\lambda_i)^K$对应的取值范围为[0,2]。
- S_{adj}的奇数次幂在$\lambda_i > 1$时滤波系数为负。
- 添加了自循环的$\tilde{S}_{adj}$的最大特征值从 2 减小至 1.5 左右，并且消除了负滤波系数的影响。
- 可以用$\tilde{S}_{adj}$的 K>1 幂定义滤波器，此时为一个低通滤波器。

(4) 实验分析

SGC 在很多任务上性能不亚于 GCN、GNN，并且效率更高。

① Citation Networks & Social Networks

A. 性能

在 Citation Networks 上：

- SGC 和 GCN 性能相当，且在 Citeseer 数据集上高出 GCN1%，性能提升的可能原因是 SGC 的参数较少减轻了过拟合的情况。
- GIN 过拟合严重。

- LNet 和 AdaLNet 在 Citation Networks 不稳定。

  在 Reddit Social Networks 上：
- SGC 比 GCN 的变种 GraphSAGE 和 FastGCN 的性能高出 1%。
- DGI 和随机初始化的 DGI 在 Reddit 上的表现都不如 SGC，该结果表明，额外的权重和非线性激活函数在 DGI 中可能是多余的。

B. 效率

- SGC 的效率是最高的。
- 由于$S^K X$是预先计算的，SGC 在训练时只需要学习一个权重矩阵θ，减少了内存使用。
- 由于 S 一般是稀疏的，且 K 通常比较小，因此可以用快速稀疏稠密矩阵乘法计算$S^K X$。
- 由于内存限制在大图上无法进行 GCN 训练。FastGCN 和 GraphSAGE 等采用采样方法，Deep Graph InfoMax 通过限制模型规模来解决该问题。
- SGC 比使用快速采样的 FastGCN 快两个数量级，并且性能几乎没有损失。

② 下游任务

在文本分类，半监督用户定位、关系提取和 zero-shot 图像分类任务上获得了不亚于基于 GCN 的模型的性能，并且效率更高。在图分类任务上，SGC 与 GCN 相当，但远远落后于 GIN。

4. 学习总结

起初简单运行完代码后便试图直接阅读文献，却发现没读几句就遇到一堆不懂的概念。于是找了两篇近两年的文献综述，读完对 GNN、GCN 的背景知识、已有模型、应用领域、发展现状等有了一定了解。然后根据综述总结的几种模型找到对应的文献，没读几句又碰到一些不太懂的深度学习概念。于是又跟着吴恩达老师的视频课程学了一遍深度学习，简单入门了 MLP、CNN 和 RNN。之后再回到综述涉及到的几篇文献，跳过了一些难懂的细节，读完后对 GNN、GCN 的基本模型和方法有了一定的了解并做了简单总结。最后再回到那两篇文献，终于可以相对顺畅地读完了。通过这样的递归学习学到了很多知识。

在阅读综述相关的文献的过程中感受到了自身数学基础的薄弱，例如图的频域卷积涉及到的傅里叶变换，我们的高数课程中讲到了傅里叶级数却是简单带过，最后只记住了几个展开公式，对其背景和意义却一无所知，造成了该部分理解上的困难。

此外，我认识到了 ideal 的重要性，Simplifying Graph Convolutional Networks 中只是消除了层之间的非线性却能得到一个相当不错的结果。我也认

识到了实验的重要性，这有点像物理学科，最终都得通过实验数据来佐证想法的正确性。

接下来，我会学完《Hands-On Machine Learning》剩余部分，巩固之前学习的深度学习知识，同时强化对 Scikit-Learn 和 TensorFlow 库的使用。然后通过《Pattern Recognition and Machine Learning》和《Deep Learning》系统深入地学习机器学习和深度学习的理论知识，并在该过程中对涉及到的数学知识进行查漏补缺。同时广泛阅读 GNN 相关文献（在 github 上发现了清华大学一个组创建的 GNN 必读文献项目 https://github.com/thunlp/GNNPapers），进一步了解和学习该研究方向。