1 递归神经网络

主要参考【1】

1.1 Recurrent graph networks (RecGNNs) 递归图神经网络

递归神经网络与 RecGNN

从数据形式上来说，递归神经网络通常被分为 结构递归神经网络与时间递归神经网络两类；其中，结构递归神经网络 即是通常习惯中所称的递归神经网络，而时间递归神经网络更多时候被称为循环神经网络。从逻辑上来说，循环神经网络 ∈ 递归神经网络。

因此，作为图神经网络最先开始的研究方向之一，RecGNN 虽也属于递归神经网络，但其与通常的在时间上展开的 RNN 不一样，属于一种在空间（图结构）上展开递归神经网络。

RecRNN 的基本原理即 在图中的节点上反复应用**相同的参数集**以**提取高级节点表示**。

1.2 Example

GNN - 1st Generation

基于信息扩散机制，GNN（Scarselli 等人所提出【2】） 通过反复交换邻域信息来更新节点的状态，直到达到稳定的平衡，节点的隐藏状态通过以下方式递归更新：

$$h_v^{(t)} = \sum_{u \in N(v)} f\left(x_v, x^e(u,v), x_u, h_u^{(t-1)}\right), \tag{1.1}$$

这里的$f(\cdot)$是一个参数函数，$h_v^{(0)}$ 随机初始化。为了保证收敛，递归函数$f(\cdot)$必须是一个压缩映射。

Def 1. 压缩映射（泛函分析）

设(X,d)为非空的完备度量空间，若$T: X \to X$为 X 上的一个压缩映射，当且仅当存在一个非负的实数$L < 1$，使得对于所有X中的x, y有:

$$d\big(T(x),T(y)\big) \le L \cdot d(x,y). \tag{1.2}$$

压缩映射本质上可以看作一个满足 Lipschitz 条件$L < 1$的映射，同时可以证明满足 L 条件的函数必然连续。通常来说，经过压缩映射后的空间会变小；同时由定义可推得存在一个经映射后保持不动的点（即不动点），即若不断对空间X进行压缩映射，该空间最终会收敛到同一个点。

引理 1. 压缩映射原理 1922

设$< X,d >$是完备的距离空间。T是一个X上的压缩映射。则 T 在 X 中恰有唯一的不动点。设不动点为$\bar{x}$，则对任意初始点$x_0 \in X$，逐次迭代$x_{n+1} = Tx_n$，于是有不等式

$$d(x_n,\bar{x}) \le L^n(1-L)^{-1}d(Tx_0,x_0). \tag{1.3}$$

引理 2. 压缩映射的充要条件

一元实函数f为压缩映射当且仅当f在任意点的梯度（导函数）的绝对值小于 1。由压缩压缩的定义

$$\begin{gathered} \big||f(x)-f(y)|\big| \le L\big||x-y|\big|, \\ \frac{\big||f(x)-f(y)|\big|}{\big||x-y|\big|} \le L, \\ \big||f'(x)|\big| = \left|\left|\frac{df(x)}{dx}\right|\right| \le L, \end{gathered} \tag{1.4}$$

将以上形式化推导推广到多元函数上，即有：**函数 f 是压缩映射当且仅当其雅克比矩阵的范数小于 1。(待补 1.1：缺乏验证)**

该引理提供了一种规范函数$f(\cdot)$形式的充分条件，即满足该引理条件的函数必然能保证 GCN 的特征表示在重复递归中收敛。

GGNN 门控图神经网络

GGNN 采用门控循环单元（GRU）作为递归函数，节点隐藏状态更新方式为：

$$h_v^{(t)} = GRU\left(h_v^{(t-1)}, \sum_{u \in N(v)} W h_u^{(t-1)}\right). \tag{1.5}$$

GRU 模块中的t并非意味式（1.5）是在一维时序上递归，而是对节点表示的反复迭代递归，此过程中所有节点不断整合邻居节点与本节点的信息，最终在一定次数递归后获得所需的高级节点表示 $h^{(t)}$。另外值得注意的是，GGNN 中 GRU 模块的使用避免了式（1.1）中参数函数的选择，但这并不一定保证重复调用 GRU 最终能够使得节点表示收敛**（待补 1.2：缺乏验证）**。最后，GGNN 采用 BPTT 算法学习模型参数，因此 GGNN 在处理大型图可能存在问题，因为 GGNN 需要在所有节点上多次运行递归函数，需要将所有节点的中间状态储存在内存中。

Stochastic steady-state Embedding（SSE）随机稳态嵌入

随机稳态嵌入提出了一种学习算法，该算法对于大型图更具有可扩展性。SSE 以随机和异步的方式反复更新节点隐藏状态。它交替地对一批节点进行状态更新，对另一批节点进行梯度计算。为了保持稳定性，SSE 的循环函数被定义为历史状态和新状态的加权平均

$$h_v^{(t)} = (1-\alpha) \cdot h_v^{(t-1)} + \alpha \cdot W_1 \sigma\left(W_2\left[x_v, \sum_{u \in N(v)} \left[h_u^{(t-1)}, x_u\right]\right]\right), \tag{1.6}$$

其中α是一个超参数，$h_u^{(0)}$随机初始化。虽具有 **SSE 并没有在理论上证明节点状态会通过重复迭代而逐渐收敛。**