H1 图神经网络：GNN

GNN最早来自[2]，基于不动点理论。对于图G，每个节点v都有特征x_v，连接两个节点u和v的边也有自己的特征$x(u,v)$，GNN学习目标是获取每个节点的图感知的隐藏状态h_v。GNN通过迭代更新所有节点的隐藏状态来让每个节点感知到图上的其它节点：

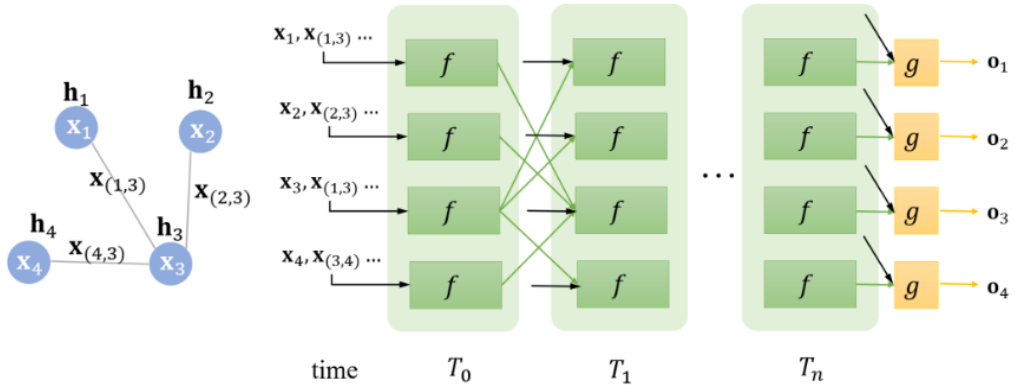
$$h_v^{t+1} = f(x_v, x_c o[v], h_n^t e[v], x_n e[v])$$

f是状态迭代更新函数，对每个节点都成立，是全局共享的。$x_c o[v]$是与接待你v相邻边的特征，$x_n e[v]$是节点v邻居节点的特征，$h_n^t e[v]$是t时刻的隐藏状态。

因此GNN思想就是**不断地利用当前时刻邻居结点的隐藏状态作为部分输入来生成下一时刻中心结点的隐藏状态，直到每个结点的隐藏状态变化幅度很小，整个图的信息流动趋于平稳。**除此之外还有个函数g来描述如何适应下游任务，即控制输出。

$$O_v = g(h_v, x_v)$$

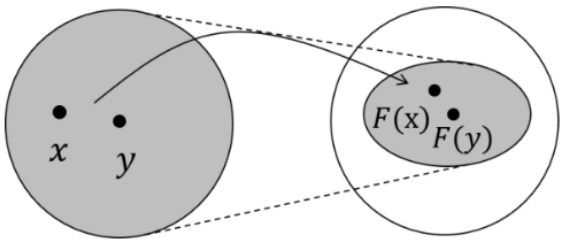
g被称为局部输出函数，也是全局共享的函数。



对于不同的图来说，收敛时刻是不同的，是通过判断两个时刻的$p-$范数的差值是否小于某个阈值ϵ来判定的。

关于不动点理论：

假设F是若干个f堆叠得到的函数，也称全局更新函数。那么状态更新可以大致写为$H_{t+1} = F(H_t, X)$。不动点理论就是无论H_0是多少，只要F是一个压缩映射，H_0就会收敛成一个固定的点。



$$d\big(F(x),F(y)\big) \le cd(x,y)$$
$$0 \le c < 1$$

（箭头就是压缩映射）

要想保证f是个压缩映射，是通过限制f对偏导数矩阵的大小，通过一个雅可比矩阵的惩罚项来实现的。将惩罚项的范数小于等于c的条件根据拉格朗日乘子法变为无约束优化问题，训练目标可表示为：

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

Loss的话就是输出和label的差异，最简单的就是差值和。那么参数通过反向传播迭代更新，得到每一层对隐层状态h_v^n的梯度。这就是Almeida-Pineda算法（AP）。

GNN的局限：

- GNN只将边作为一种传播手段，但并未区分不同边的功能。
- 如果把GNN应用在图表示的场景中，使用不动点理论并不合适。这主要是因为基于不动点的收敛会导致结点之间的隐藏状态间存在较多信息共享，从而导致结点的状态太过光滑(Over Smooth)，并且属于结点自身的特征信息匮乏(Less Informative)。

门控图神经网络将节点的邻居节点隐藏信息按权加和放入更新函数，特别是加入了可学习参数W：

$$h^{t+1} = GRU(h_v^t, \sum u \in ne[v] W_{edge} h_u^t)$$

GGNN将节点特征作为初始隐藏状态，然后按照公式迭代若干步，利用BP反向传播求得W_{edge}和GRU参数的梯度。

> 摘自 https://www.cnblogs.com/SivilTaram/p/graph_neural_network_1.html