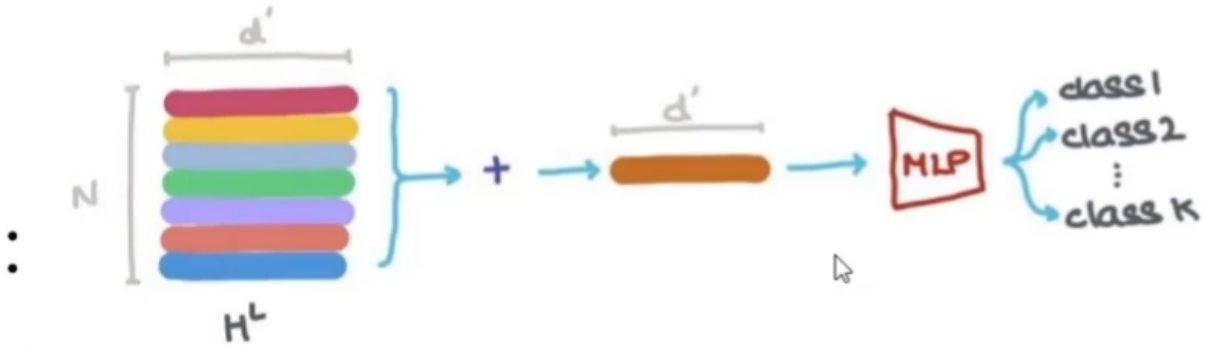
GNN

GNN可以有很多层

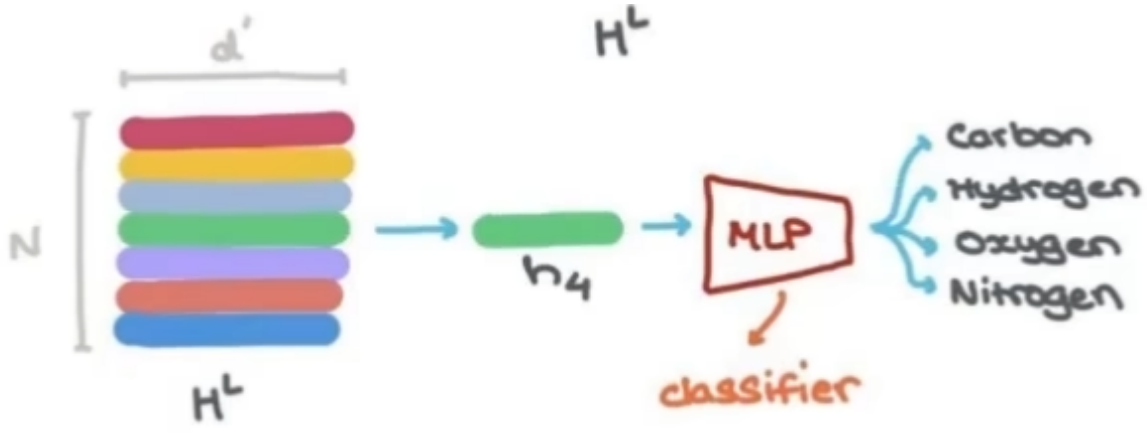
- GNN的本质就是更新各部分特征
- 其中输入是特征，输出也是特征，邻居矩阵也不会变
- 多层的GNN允许消息从局部到全局的一个传播，增大每个节点的感受野

输出的特征能干嘛：

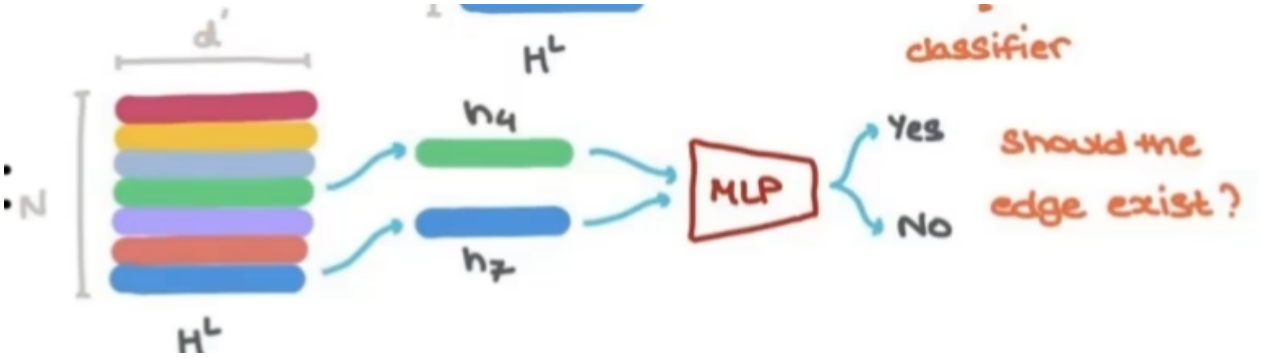
- 各个点特征组合，可以图分类



- 各个节点也可以分类



- 边也是如此



- 其实只是利用图结构得到特征，最终要做什么还是我们自己定

GCN的基本思想：

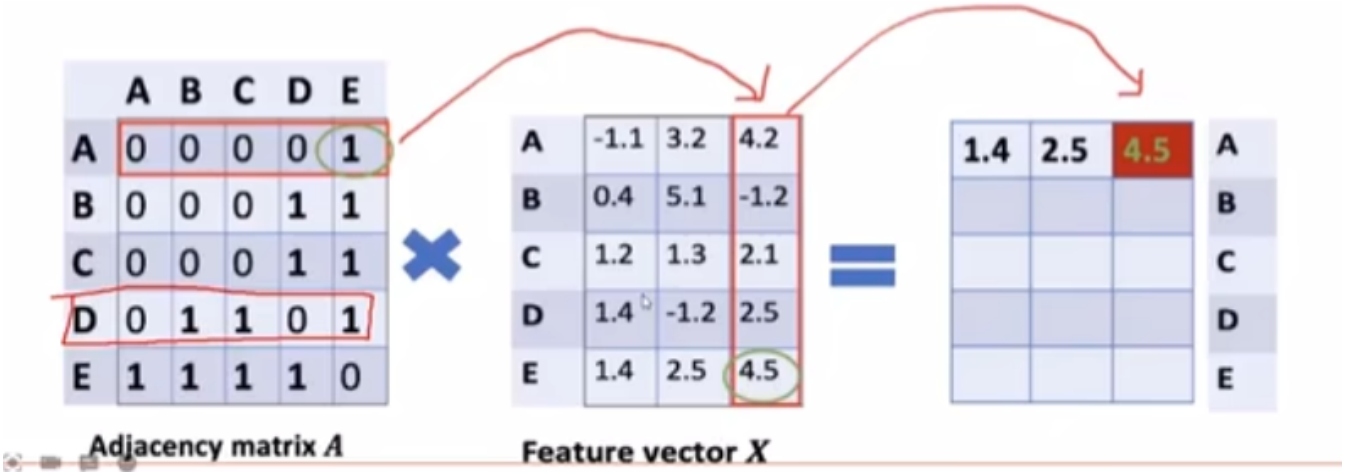
> 针对橙色节点，计算它的特征，（也称为聚合）平均其邻居特征（包括自身）后传入神经网络。

图的基本组成

- G：图
- A：邻接矩阵
- D：各个节点的度
- F：每个节点的特征

特征计算方法:

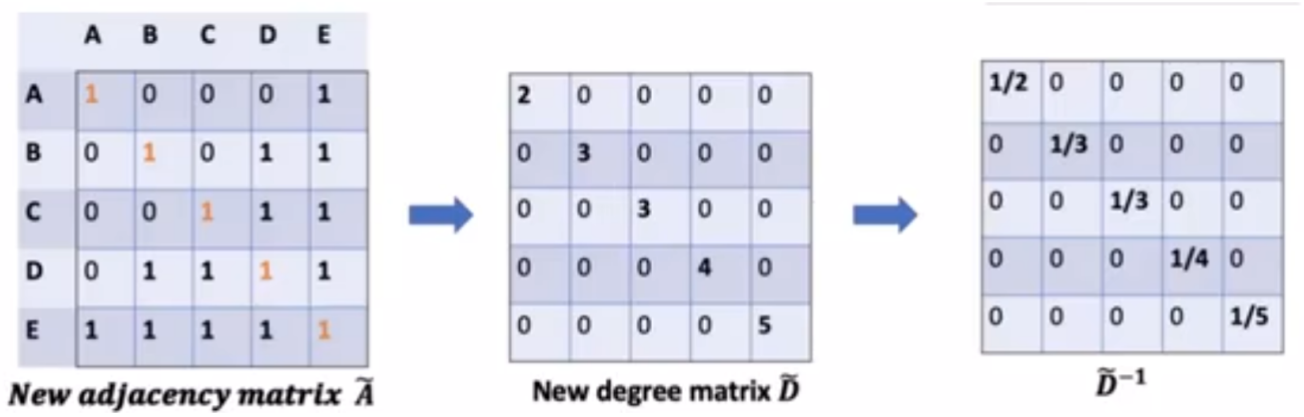
- 邻接矩阵与特征矩阵进程（点）乘法操作，表示聚合邻居信息



- 邻接矩阵需要变：还需要加上单位矩阵形成自环。

$$\tilde{A} = A + \lambda I_N$$

- 度矩阵也要变一变,最后取个倒数的目的是将A*F的结果进行求平均。



矩阵scale：

目前公式变为$(\widetilde{D}^{-1}\widetilde{A})X$，由于是点乘，所以可以变换为$(\widetilde{D}^{-1}\widetilde{A})X$,其中$\widetilde{D}^{-1}\widetilde{A}$，可以看成A的归一化操作，所以$\widetilde{D}^{-1}$就相当于scale方法了。

- 左乘相当于对行做归一化
- 那列怎么办? 右乘。

现在公式变为$\widetilde{D}^{-1}\widehat{A}\widetilde{D}^{-1}X$，相当于对$\widetilde{A}$进行了行列归一化。

但是每个值都被做了两次归一化，所以再对公式进行变换：$\widetilde{D}^{-\frac{1}{2}}\widehat{A}\widetilde{D}^{-\frac{1}{2}}X$

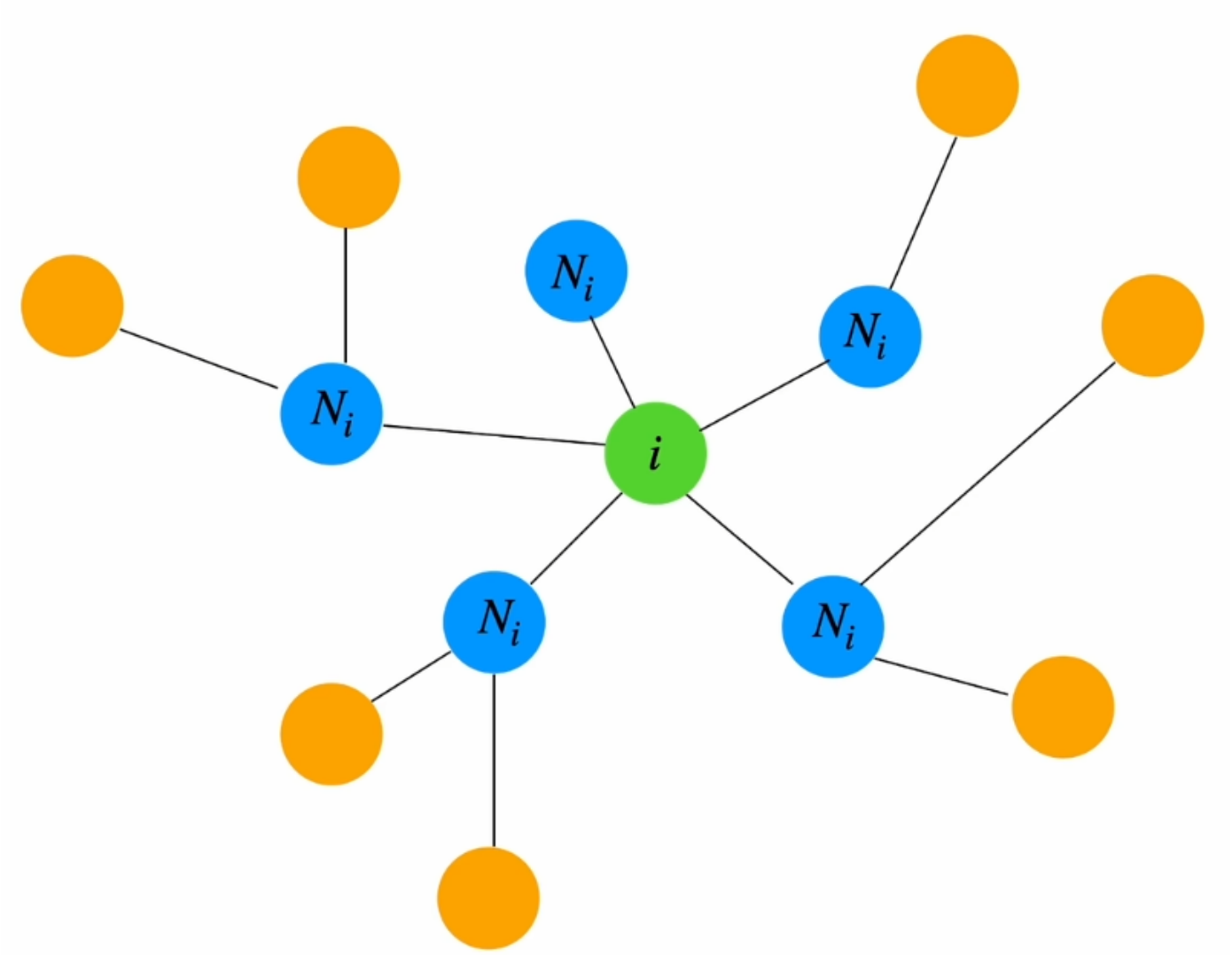
这样变换的目的是防止度小的节点的受到度大的邻居节点的影响而导致特征计算出现严重的偏差。

基本公式：

- 例如完成一个十分类任务的，F就为10表示输出层

> $\widehat{A} = \widetilde{D}^{1/2}\widetilde{A}\widetilde{D}^{1/2}$scaled 邻接矩阵
>
> $$Z = f(X, A) = softmax(\widehat{A}ReLU(\widehat{A}XW^{(0)})W^{(1)})$$
>
> $ReLU(\widehat{A}XW^{(0)})$:第一层
>
> X:特征向量矩阵（NxC）
>
> $W^{(0)}$: trainable weights(CxH)
>
> $W^{(1)}$:trainable weights(HxF)

GCN的层数，在多个图数据的中，都可以发现两三层的比较合适，多了反而差了。



图注意力网络（GAT）

GAT本质上有两种计算方式：

> - Global Graph Attention

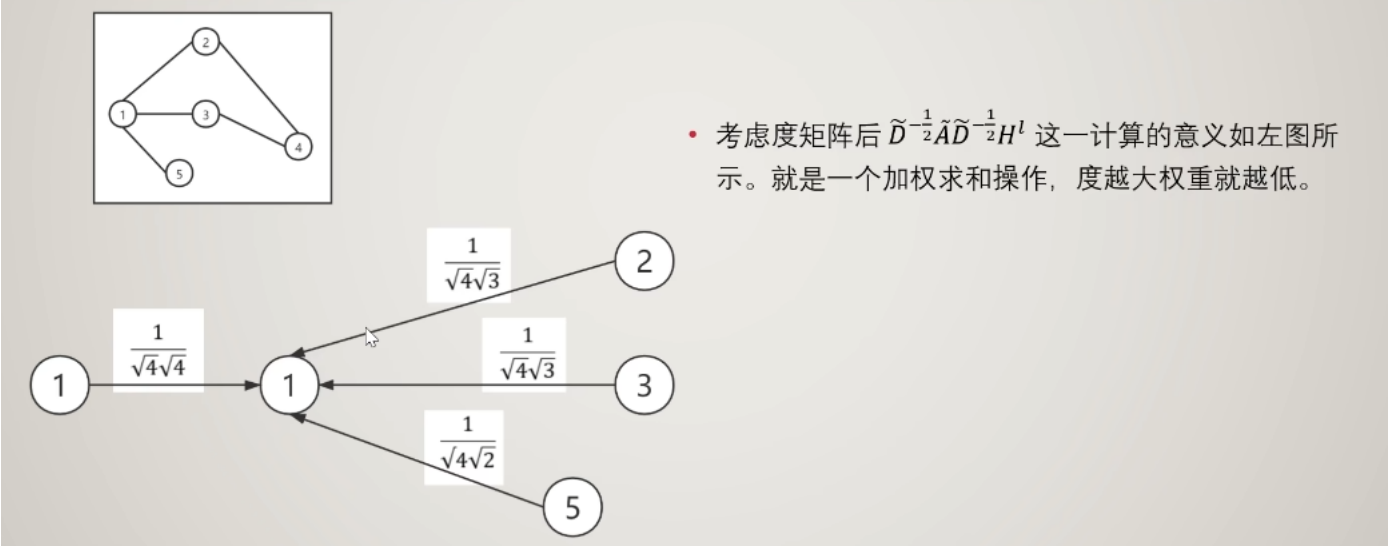
> 节点i（绿色）与图上所有节点（橘色节点和蓝色节点）做attention计算：
>
> - 丢失了图的结构特征
> - 运算成本巨大

- Mask Graph Attention

> 节点i（绿色）至于（一阶）邻居节点（蓝色节点）做attention计算；
>
> - 利用图的结构特征
> - 减少了运算量。

GAT和GCN的区别在于：

- GCN使用度矩阵去衡量一个节点对另一个节点的影响程度
- GAT使用注意力机制去衡量一个节点对另一个节点的影响程度



GAT：加入了注意力机制的图神经网络，其消息传递的权重是通过注意力机制得到的。

GAT计算过程：

- $a_{ij} = softmax_j(e_{ij}) = \frac{exp(e_{ij})}{\sum_{k \in N_i} exp(e_{ik})}$
- $e_{ij} = LeakyRelu(\alpha^T[Wh_i || Wh_j])$
  - h_i和h_j是当前输出层的节点i与节点j的特征表示，W是线性变换矩阵，$W \in R^{F \times F'}$,其中F就是输入特征的维度。F'是输出特征的维度。

    - ||是向量拼接操作，原本维度为F的h_i和 h_j 经过W线性变换后维度均变为F'，经过拼接后得到维度为$2F'$的向量。此时再点乘一个维度为$2F'$的单层矩阵α的转置，然后经LeakyReLU激活后得到1维的e_{ij}
    - 得到所有e_{ij} 后，进行softmax操作，得到注意力权重a_{ij}
- 计算节点i在当前GAT网络层的输出向量h_i' 可以描述为：$h_i' = \sigma(\sum_{j \in N_i} \alpha_{ij} W h_j)$
    - 其中$\sigma(\cdot)$代表任意激活函数，N_i 代表节点i的一阶邻居集，W与注意力计算中的W一样。
    - 这是一个消息传递，并用加权求和的方式进行消息聚合的计算过程。
- 在GAT中，可以进行多次消息传递操作，然后将每次得到的向量拼接或者求平均。这成为多头注意力（Multi-Head Attention）。
- GAT的论文中建议在GAT网络中间的隐藏层采取拼接操作，而最后一层采取平均操作。
    - 拼接每一层弹头消息传递得到的向量：

      $$h_i' = \Big\|_{k=1}^{K} \sigma(\sum_{j \in N_i} \alpha_{ij}^k W^k h_j)$$

    - 平均每一层弹头消息传递得到的向量：

      $$h_i' = \sigma(\frac{1}{K} \sum_{k=1}^{K} \sum_{j \in N_i} \alpha_{ij}^k W^k h_j)$$