作业6：GNN(1)

- **GCN模型：**

$$\mathbf{h}_v^k = \sigma\left(\mathbf{W}_k \sum_{u \in N(v) \cup v} \frac{\mathbf{h}_u^{k-1}}{\sqrt{|N(u)||N(v)|}}\right)$$

作业6：GNN(1)

- （10分）用pytorch和pyg实现一个可训练的GCN网络层，按下述代码框架进行补全，不能直接调库。之后按作业5的方式测试单层或多层GCN在Cora数据集上进行点分类的性能。

```
class GCNLayer(nn.Module):
    def __init__(self, in_size, out_size):
        super(GCNLayer, self).__init__()
        self.W = nn.Linear(in_size, out_size)

    def forward(self, A, X):
        ### A: adjacency matrix, shape: (N, N)
        ### X: input feature matrix, shape: (N, in_size)
        ### return: output feature matrix, shape: (N, out_size)
        ###
        ### YOUR CODE HERE
        ###
        return
```

作业6：GNN(1)

- **可选题**：使用scatter函数(https://pytorch-scatter.readthedocs.io/en/latest/functions/scatter.html)，实现一个稀疏图(edge_index作为输入)的GCN网络层，并探索一些问题（见下页）：

```
class GCNLayer(nn.Module):
    def __init__(self, in_size, out_size):
        super(GCNLayer, self).__init__()
        self.W = nn.Linear(in_size, out_size)

    def forward(self, edge_index, X):
        ### edge_index: edge index tensor, shape: (2, E)
        ### X: input feature matrix, shape: (N, in_size)
        ### return: output feature matrix, shape: (N, out_size)
        ###
        ### YOUR CODE HERE
        ###
        return
```

作业6：GNN(1)

- 在Cora数据上进行实验，在权重W相同时，你实现的使用edge_index和A作为输入的两个模型的计算结果是否相同？（去除浮点数的硬件随机误差）
- 生成一些较大的虚拟图（可以使用pyg的ERGraph生成器生成Erdos-Renyi图，尝试100k-1M节点，连边概率0.1-1）进行实验：在图的大小变化时，两种方法的计算效率和内存占用随输入图的N和E的变化趋势为何？