ECE4880J: Computer Vision July 29, 2022

Homework 6: Graph Neural Networks

Instructor: Siheng Chen **Zhou Haoquan**

Q1. Permutation-equivalent Property of PointNet

(a)

Consider the expression

$$y = \text{PointNet}(X) = \text{maxpool}_1(\text{ReLU}(\text{ReLU}(XW_1)W_2))$$

we have $\text{ReLU}(XW_1) \in \mathbb{R}^{N\times d_1}$, and then $\text{ReLU}(\text{ReLU}(XW_1)W_2) \in \mathbb{R}^{N\times d_2}$. After maxpooling, the first dimension is reduced to 1, then $y \in \mathbb{R}^{1\times d_2}$.

(b)

Now the expression becomes

$$y^{'} = \text{PointNet}(PX) = \text{maxpool}_1(\text{ReLU}(\text{ReLU}(PXW_1)W_2))$$

We take a careful look at the part

$$\text{ReLU}(\text{ReLU}(PXW_1)W_2)$$

We find that since $P \in \{0,1\}^{N\times N}$ does not generate negative numbers, we can extract it out and apply the associative law of matrix multiplication, write

$$P\text{ReLU}(\text{ReLU}(XW_1)W_2)$$

Also, note that $PI_{ij} = I_{arg(P(i,\cdot)=1),j}$, so the result is equivalent to doing permutation on each column of input matrix I. Meanwhile, maxpooling will always extract the maximum value for each column of input matrix I. So $\text{maxpool}_1(PI) = \text{maxpool}_1(I)$ for $\forall P \in \{0,1\}^{N,N}$. Now we can conclude that

$$\text{maxpool}_1(P\text{ReLU}(\text{ReLU}(XW_1)W_2)) = \text{maxpool}_1(\text{ReLU}(\text{ReLU}(XW_1)W_2))$$

which means

$$y^{'} = y$$

(c)

No difference. Training of W_1 and W_2 may be different for y and $y^{'}$, and in fact W_1 and W_2 will almost surely be different for y and $y^{'}$, since they have different permutations. However, the maxpooling will makes the results identical.

Q2. Oversmoothing Issue of GCNs

(a)

We have $AXW \in \mathbb{R}^{N \times D}$, thus $\text{ReLU}(AXW) \in \mathbb{R}^{N \times D}$

(b)

We have $Y_{i+1} = A(Y_i)W_{i+1}$, by concatenating this repetitive structure, $Y_K = A^K X(\Pi_{i=1}^K W_i)$

(c)

The adjacency matrix is

$$A = \begin{bmatrix} 0 & 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0 & 0 & 0 & 0 \\ 0.25 & 0 & 0 & 0 & 0.75 \\ 0.25 & 0 & 0 & 0 & 0 \\ 0.25 & 0 & 0.75 & 0 & 0 \end{bmatrix}$$

(d)

$$A^{10} = \begin{bmatrix} 0.067 & 0.018 & 0.103 & 0.018 & 0.103 \\ 0.018 & 0.005 & 0.028 & 0.005 & 0.028 \\ 0.103 & 0.028 & 0.188 & 0.028 & 0.131 \\ 0.018 & 0.005 & 0.028 & 0.005 & 0.028 \\ 0.103 & 0.028 & 0.131 & 0.028 & 0.188 \end{bmatrix}$$

$$A^{100} = \begin{bmatrix} 1.579e-05 & 4.330e-06 & 2.445e-05 & 4.330e-06 & 2.445e-05 \\ 4.330e-06 & 1.190e-06 & 6.710e-06 & 1.190e-06 & 6.710e-06 \\ 2.445e-05 & 6.710e-06 & 3.787e-05 & 6.710e-06 & 3.787e-05 \\ 4.330e-06 & 1.190e-06 & 6.710e-06 & 1.190e-06 & 6.710e-06 \\ 2.445e-05 & 6.710e-06 & 3.787e-05 & 6.710e-06 & 3.787e-05 \end{bmatrix}$$

$$A^{1000} = \begin{bmatrix} 8.970e-42 & 2.460e-42 & 1.389e-41 & 2.460e-42 & 1.389e-41 \\ 2.460e-42 & 6.700e-43 & 3.810e-42 & 6.700e-43 & 3.810e-42 \\ 1.389e-41 & 3.810e-42 & 2.151e-41 & 3.810e-42 & 2.151e-41 \\ 2.460e-42 & 6.700e-43 & 3.810e-42 & 6.700e-43 & 3.810e-42 \\ 1.389e-41 & 3.810e-42 & 2.151e-41 & 3.810e-42 & 2.151e-41 \end{bmatrix}$$

We find that the matrixs tend to vanish as the number of power increases. Also, we find that the difference between each entry also becomes smaller due to the decrease of scale. This is because for each matrix multiplication makes the entry smaller, and makes the difference between entries smaller.

(e)

$$A^{1000}x_1 = \begin{bmatrix} 8.970e-39 & 2.460e-39 & 1.389e-38 & 2.460e-39 & 1.389e-38 \end{bmatrix}$$

$$A^{1000}x_2 = \begin{bmatrix} 2.460e-39 & 6.748e-40 & 3.810e-39 & 6.748e-40 & 3.810e-39 \end{bmatrix}$$

We find that even x_1 and x_2 has huge difference in two dimensions, the results obtained after multiplication do not differ much. This is because A^{1000} is too small and smooth to capture the differences. And this leads to the oversmoothing issue.