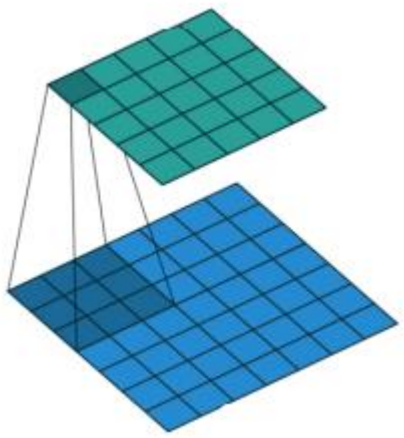
Recitation: Graph Neural Networks

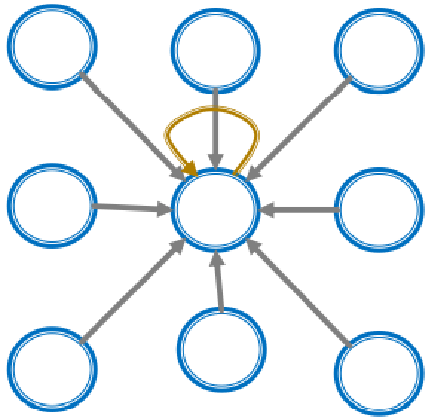
- **Quickly review GCN message passing process**
- **Graph Convolution layer forward**
- **Graph Convolution layer backward**
- **GCN code example**

A single layer of GNN: Graph Convolution

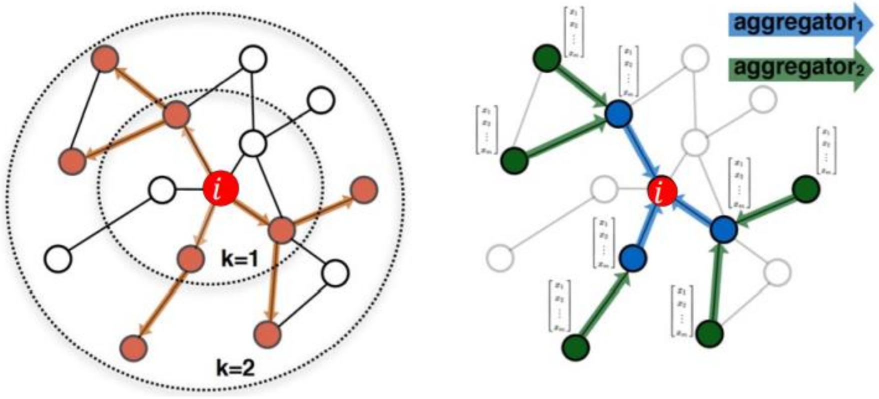
Key idea: Node's neighborhood defines a computation graph



CNN: pixel convolution



CNN: pixel convolution



GNN: graph convolution

➢ **Learning a node feature by propagating and aggregating neighbor information!**

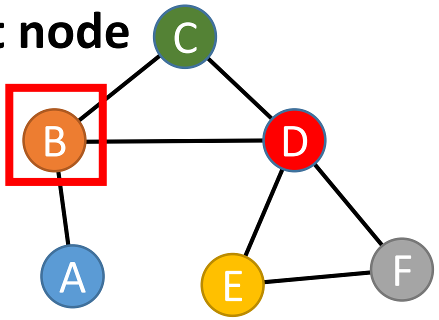
➢ **Node embedding can be defined by local network neighborhoods!**

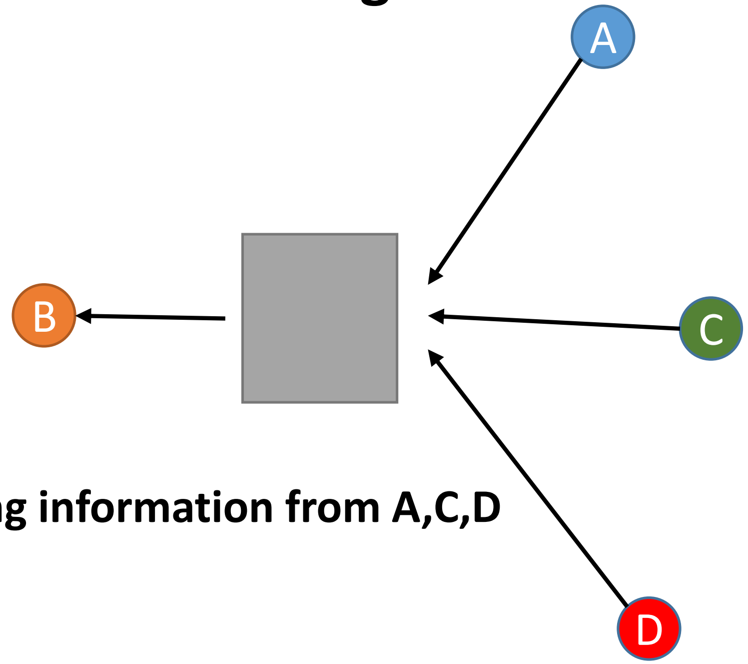
A single layer of GNN: Graph Convolution

Key idea: Generate node embedding based on local network neighborhoods

Considering 1 step of feature aggregation of the nearest neighbor

Target node



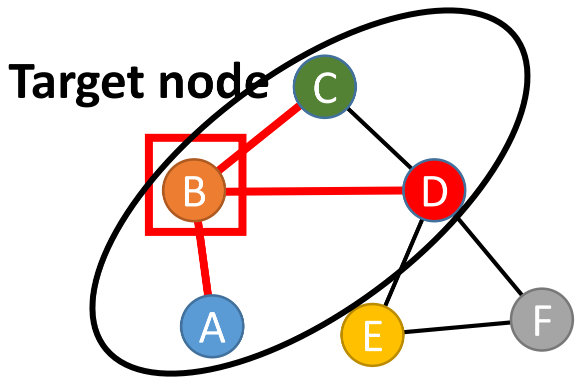


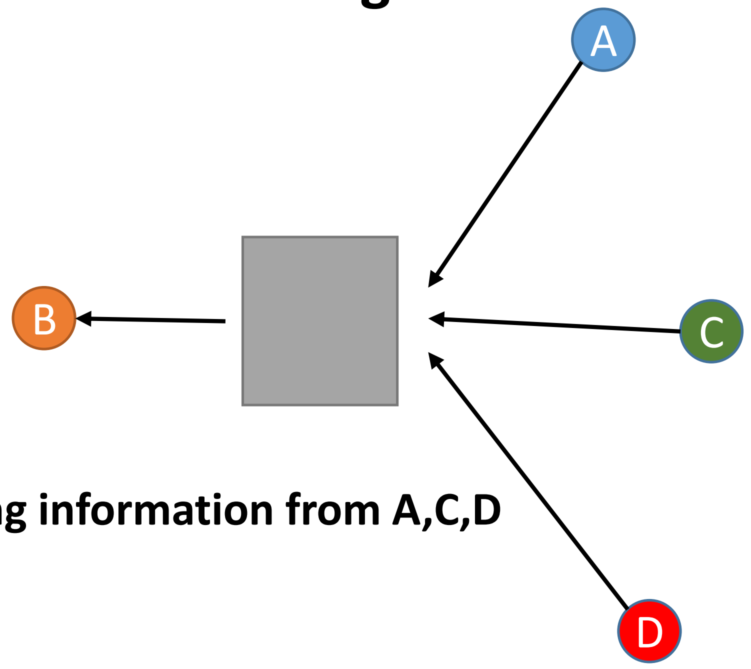
Processing information from A,C,D

A single layer of GNN: Graph Convolution

Key idea: Generate node embedding based on local network neighborhoods

Considering 1 step of feature aggregation of the nearest neighbor





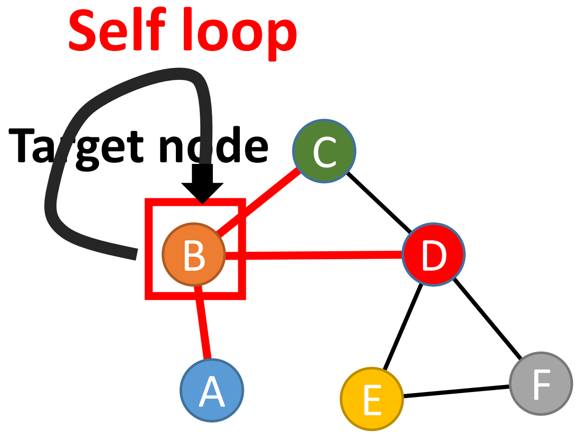
Processing information from A,C,D

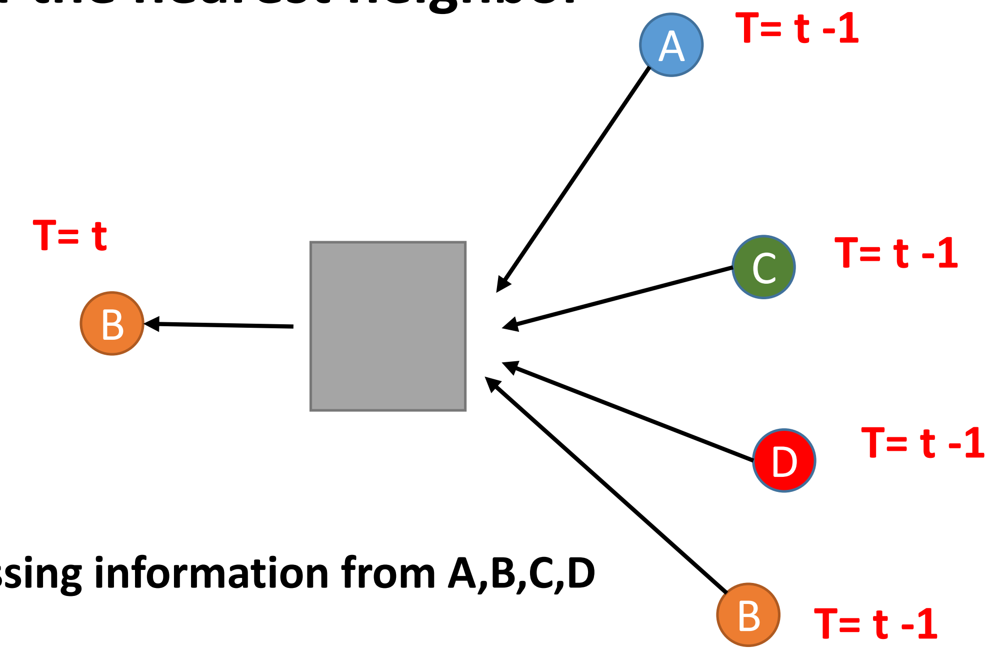
Now B have the information from it's first nearest neighbors

A single layer of GNN: Graph Convolution

Key idea: Generate node embedding based on local network neighborhoods

Considering 1 step of feature aggregation of the nearest neighbor



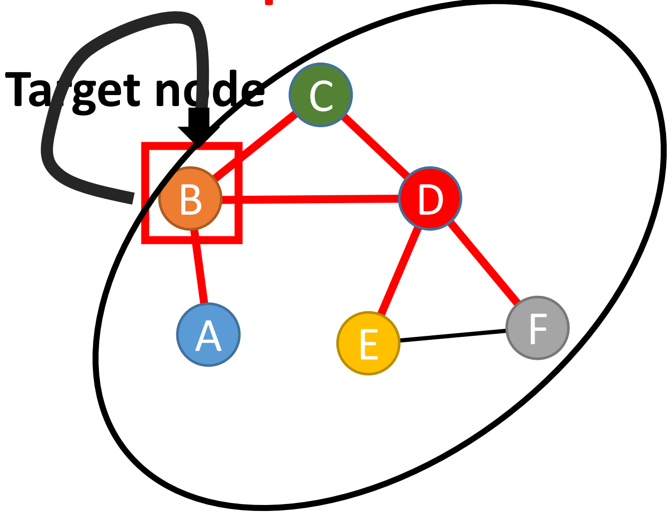


Also we don't want to lose information from B itself

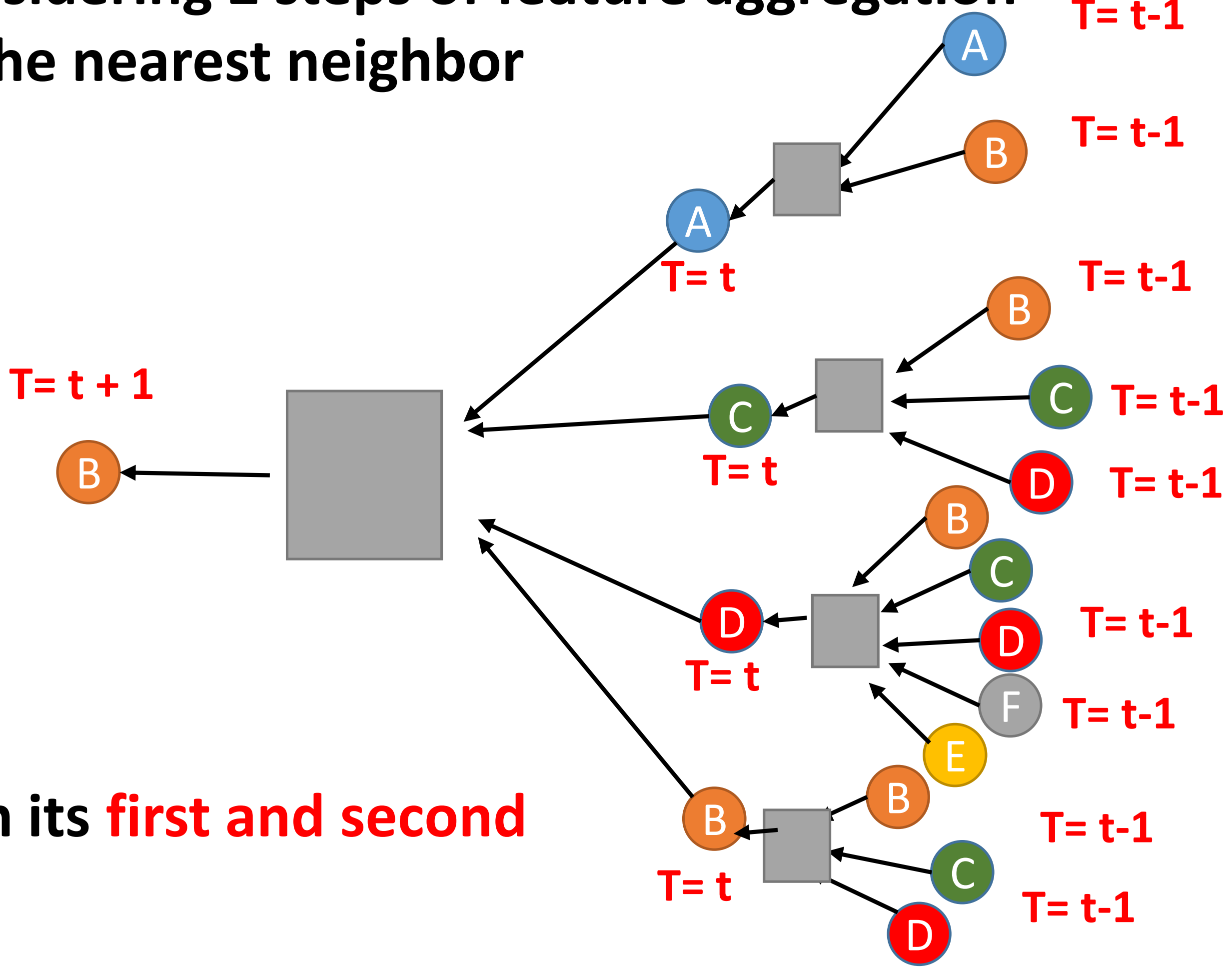
A single layer of GNN: Graph Convolution

Key idea: Generate node embedding based on local network neighborhoods

Self loop



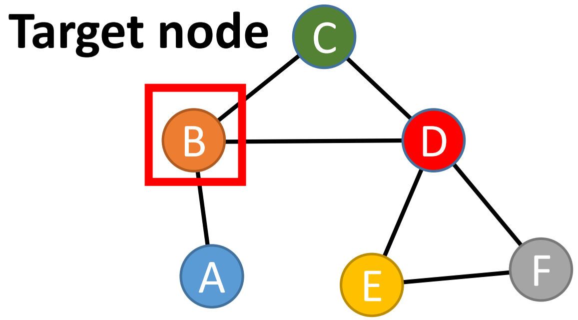
Considering 2 steps of feature aggregation of the nearest neighbor



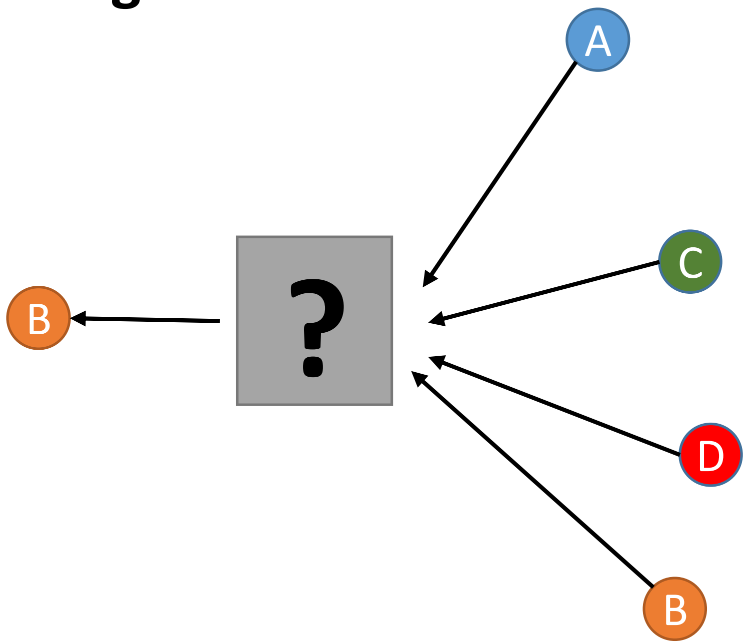
Now B have the information from its first and second nearest neighbors

A single layer of GNN: Graph Convolution

Key idea: Generate node embedding based on local network neighborhoods



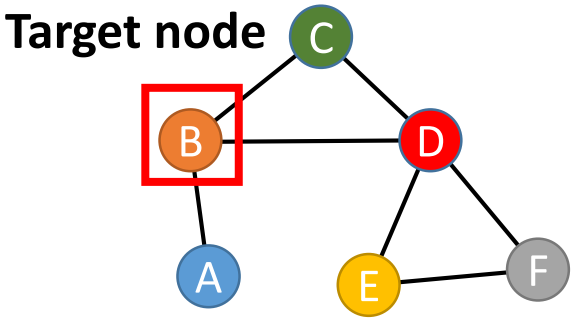
How to process and mix the information from neighbor?

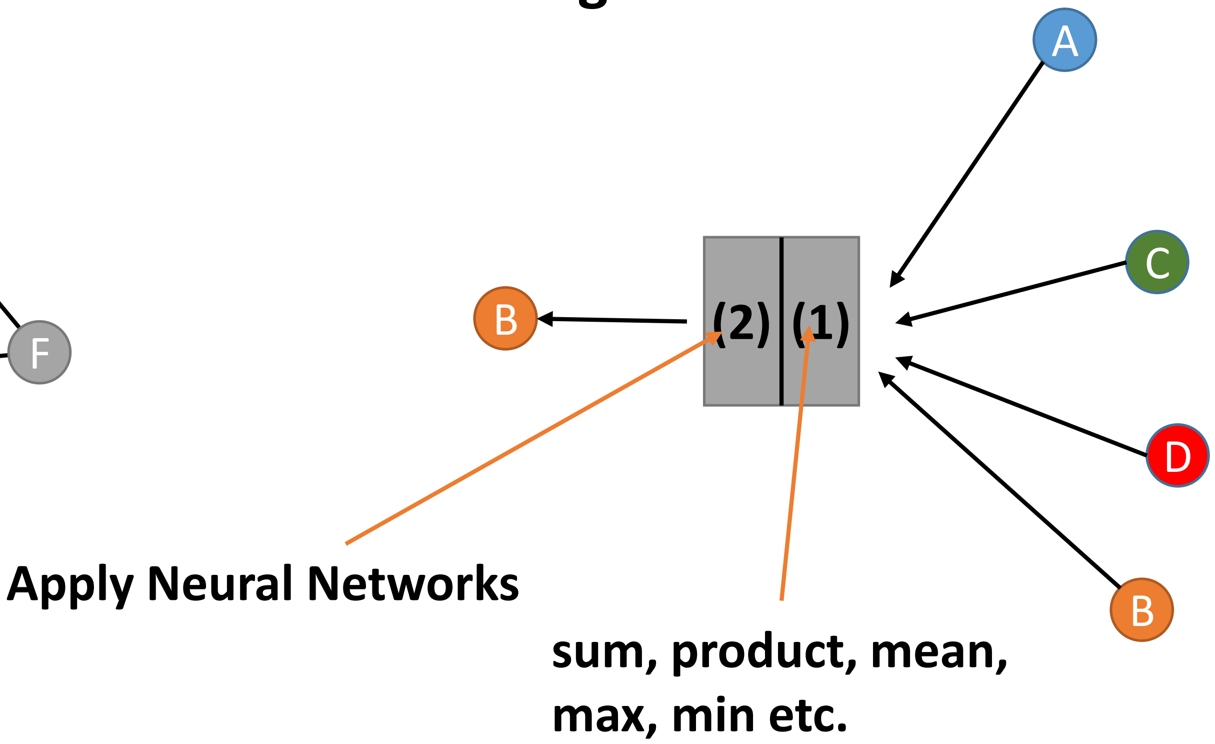


A single layer of GNN: Graph Convolution

Key idea: Generate node embedding based on local network neighborhoods

How to process and mix the information from neighbor?

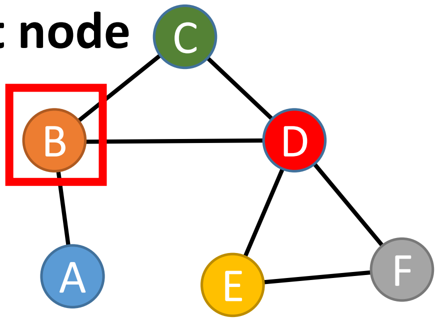


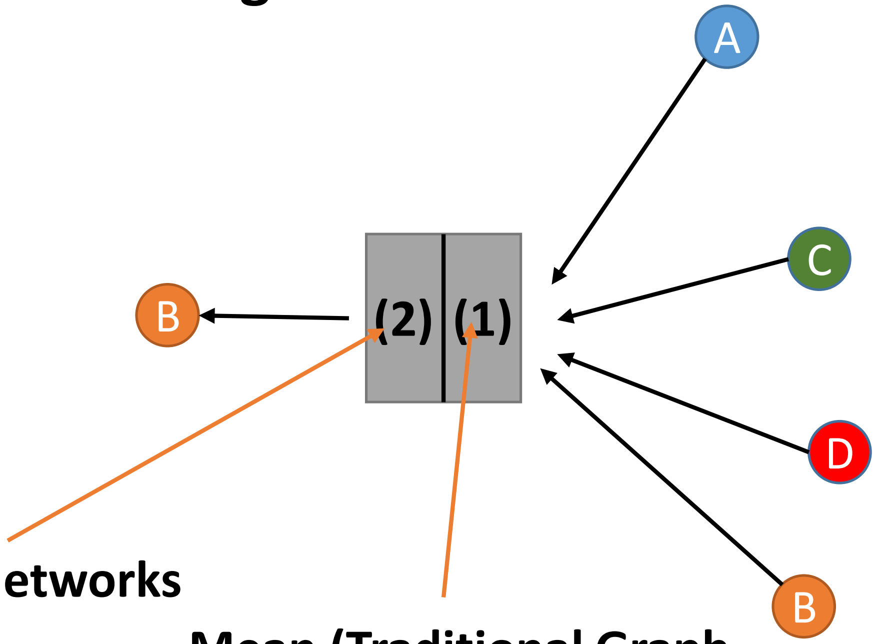


A single layer of GNN: Graph Convolution

Key idea: Generate node embedding based on local network neighborhoods

How to process and mix the information from neighbor?



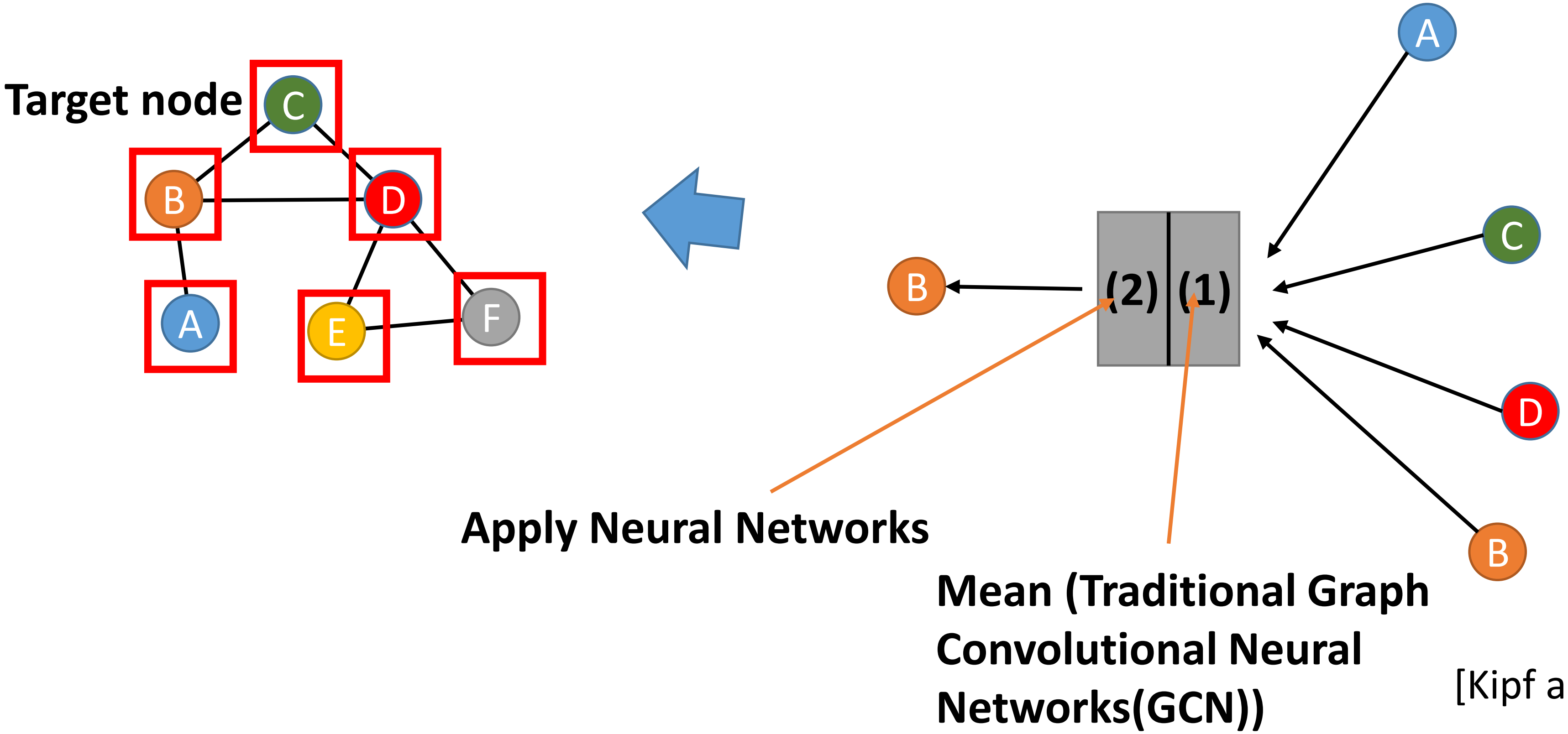


[Kipf and Welling, ICLR 2017]

A single layer of GNN: Graph Convolution

Key idea: Generate node embedding based on local network neighborhoods

During a single Graph Convolution layer, we apply the feature aggregation to every node in the graph at the same time (T)



[Kipf and Welling, ICLR 2017]

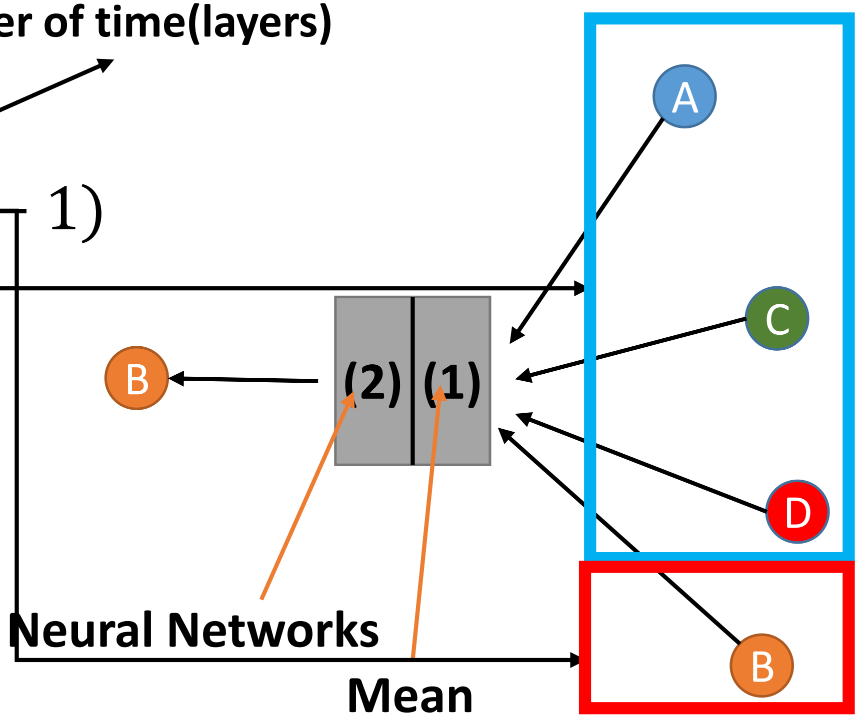
A single layer of GNN: Graph Convolution-Forward

Math for a single layer of graph convolution

$$h_v^0 = x_v^{\,(1\times F)}$$

0th layer embedding = node v initial feature

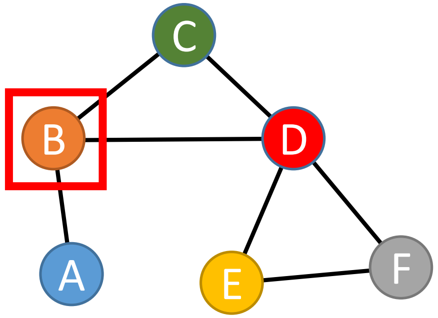
$$\underbrace{\overset{(1\times F)}{h_v^{t+1}}}_{\text{Node v feature at time(layer) t+1}} = \underbrace{\sigma}_{\text{Non-linear activation (i.e. relu())}}\left(\underbrace{W_k}_{\text{Learnable weight}} \underbrace{\sum_{u\in N(v)} \frac{\overset{(1\times F)}{h_u^t}}{|N(v)|}}_{\text{Average the neighbor node feature at time(layer) t}} + \underbrace{B_k}_{\text{Learnable weight}} \underbrace{\overset{(1\times F)}{h_v^t}}_{\text{Node v feature at time(layer) t}}\right), \forall t \in (0, \ldots\ldots \underbrace{T}_{\text{Number of time(layers)}} - 1)$$

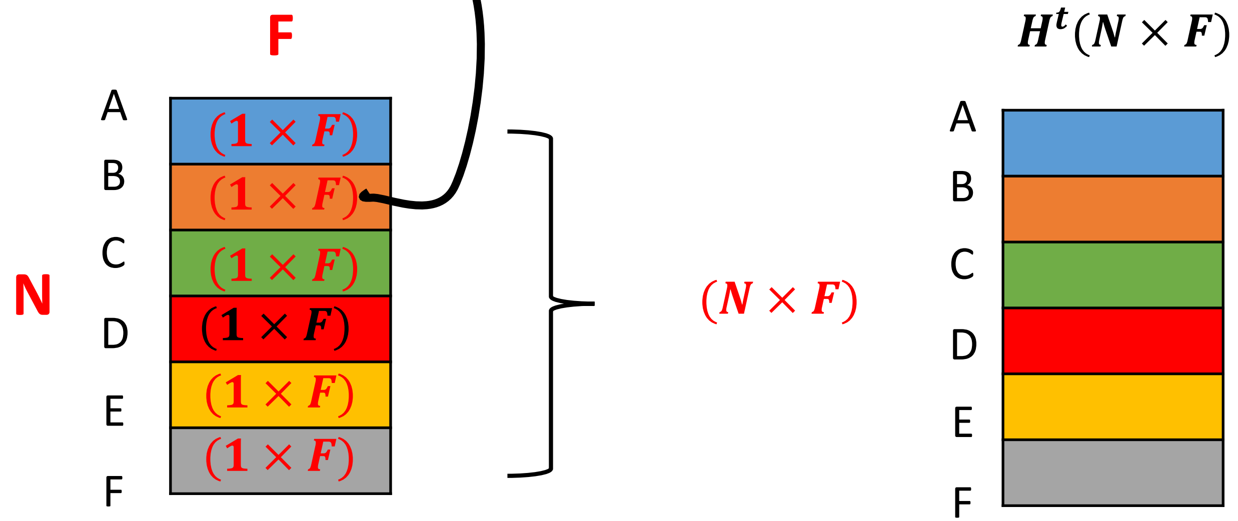


A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution

$$\overset{(1\times F)}{\boxed{h_v^{t+1}}} = \sigma\left(W_k \sum_{u\in N(v)} \frac{\overset{(1\times F)}{\boxed{h_u^t}}}{|N(v)|} + B_k \overset{(1\times F)}{\boxed{h_v^t}} \right), \forall t \in (0, \dots \dots, T-1)$$





We stack multiple $h_v^t(1 \times F)$ together into $H^t(N \times F)$

A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution

$$\underbrace{h_v^{t+1}}_{(1\times F)} = \sigma\left(W_k \boxed{\sum_{u\in N(v)} \frac{\overbrace{h_u^t}^{(1\times F)}}{|N(v)|}} + B_k \overbrace{h_v^t}^{(1\times F)} \right), \forall t \in (0, \dots\dots, T-1)$$

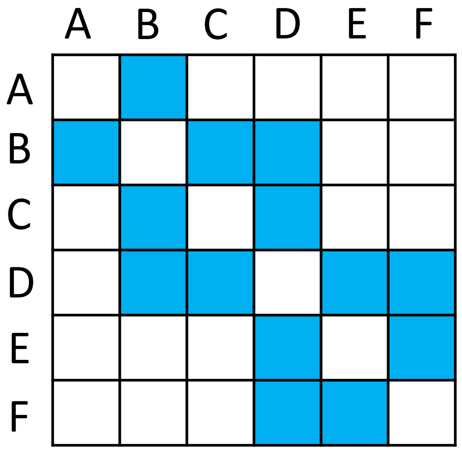
$D^{-1}(N \times N)$

$$\mathbf{N}\begin{bmatrix} \frac{1}{|N(v)|} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{|N(v)|} \end{bmatrix}$$

N

$\times$ **N**

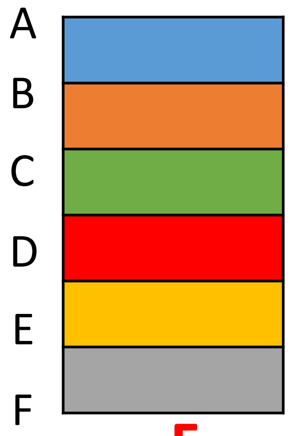
$A(N \times N)$



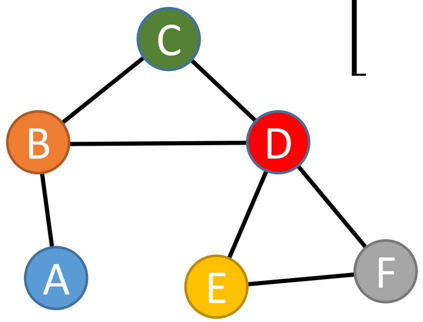
N

$\times$ **N**

$H^t(N \times F)$



F



A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution

$$\overset{(1\times F)}{h_v^{t+1}} = \sigma\left(\boxed{W_k}\,\overset{(1\times F)}{\boxed{\sum_{u\in N(v)} \frac{h_u^t}{|N(v)|}}} + \overset{(1\times F)}{B_k h_v^t}\right), \forall t \in (0, \dots\dots, T-1)$$

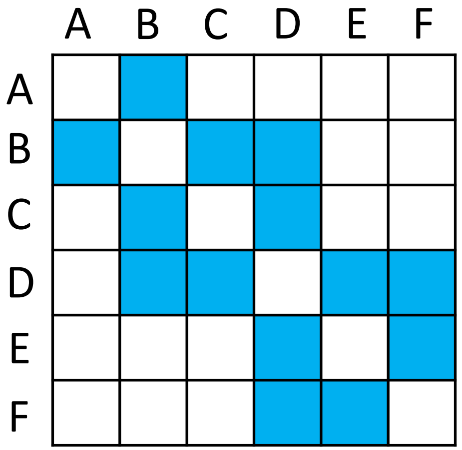
Noted that W^T is a learnable weight matrix

$\boldsymbol{D^{-1}(N \times N)}$

$$\begin{array}{c c} \mathbf{N} & \begin{bmatrix} \frac{1}{|N(v)|} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{|N(v)|} \end{bmatrix} \\ & \mathbf{N} \end{array}$$

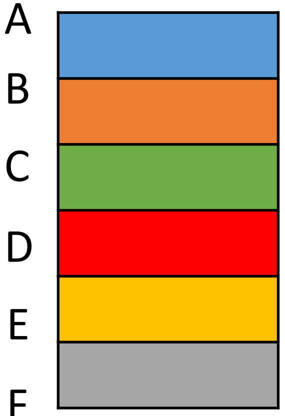
$\times$

$\boldsymbol{A(N \times N)}$



$\times$ **N**

$\boldsymbol{H^t(N \times F)}$

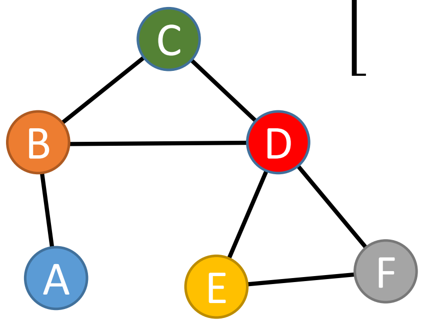


$\times$ **F**

$\boldsymbol{W^T(F \times F)}$

$$\begin{bmatrix} w_{11} & \cdots & w_{1f} \\ \vdots & \ddots & \vdots \\ w_{f1} & \cdots & w_{ff} \end{bmatrix}$$

F



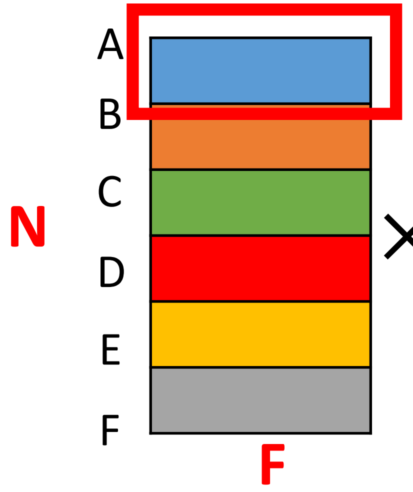
A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution

$(1 \times F)$ $(1 \times F)$ $(1 \times F)$

$$h_v^{t+1} = \sigma\left(\boxed{W_k}\boxed{\sum_{u \in N(v)} \frac{h_u^t}{|N(v)|}} + B_k h_v^t\right), \forall t \in (\,0, \ldots \ldots, T-1)$$

$H^t(N \times F)$



$\times$

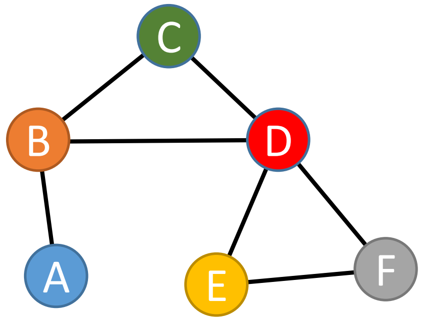
$W^T(F \times F)$

$$\begin{bmatrix} w_{11} & \cdots & w_{1f} \\ \vdots & \ddots & \vdots \\ w_{f1} & \cdots & w_{ff} \end{bmatrix}$$

F F

Why put W^T on the right hand site of H^t?

Why not left? With a shape of $(N \times N)$?



A single layer of GNN: Graph Convolution-Forward

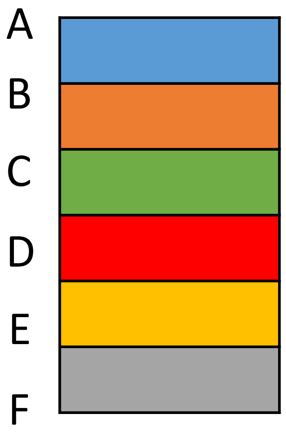
Matrix form for a single layer of graph convolution

$(\mathbf{1}\times \boldsymbol{F})$ $(\mathbf{1}\times \boldsymbol{F})$ $(\mathbf{1}\times \boldsymbol{F})$

$$h_v^{t+1} = \sigma\left(\boxed{W_k}\boxed{\sum_{u\in N(v)} \frac{h_u^t}{|N(v)|}} + B_k h_v^t\right), \forall t \in (\,0, \dots \dots, T-1)$$

$\boldsymbol{W}^{\boldsymbol{T}}(\boldsymbol{N}\times \boldsymbol{N})$ $\boldsymbol{H}^{\boldsymbol{t}}(\boldsymbol{N}\times \boldsymbol{F})$

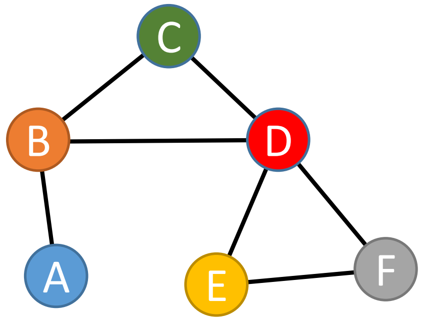
$$\begin{array}{cc} & \mathbf{N} \\ \mathbf{N} & \begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{n1} & \cdots & w_{nn} \end{bmatrix} \end{array} \times \mathbf{N}$$



F

Like this?

Seems like nothing goes wrong, the result matrix shape is still $(N \times F)$?



What happen if we still put W on the left hand site?

A single layer of GNN: Graph Convolution-Forward

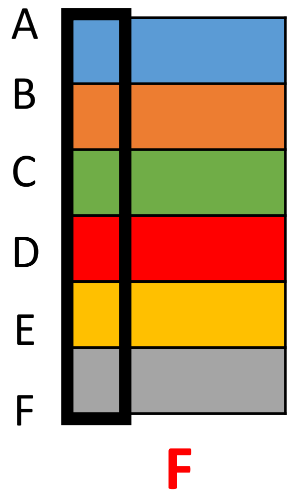
Matrix form for a single layer of graph convolution

$(1 \times F)$ $\quad(1 \times F)\quad(1 \times F)$

$$h_v^{t+1} = \sigma\left(\boxed{W_k}\boxed{\sum_{u \in N(v)} \frac{h_u^t}{|N(v)|}} + B_k h_v^t\right), \forall t \in (\,0, \dots \dots, T-1)$$

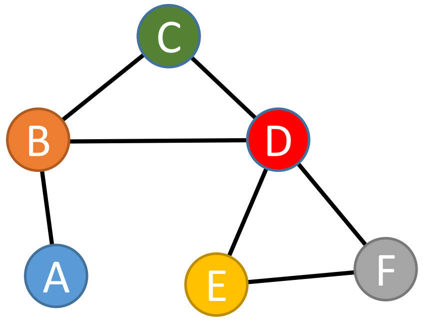
$\boldsymbol{W^T}(\boldsymbol{N} \times \boldsymbol{N})$ $\quad \boldsymbol{H^t}(\boldsymbol{N} \times \boldsymbol{F})$

$$\begin{array}{cc} \mathbf{N} & \begin{bmatrix} \boxed{w_{11} \quad \cdots \quad w_{1n}} \\ \vdots \quad \ddots \quad \vdots \\ w_{n1} \quad \cdots \quad w_{nn} \end{bmatrix} \\ & \mathbf{N} \end{array} \times \quad \mathbf{N}$$



Seems like nothing goes wrong, the result matrix shape is still $(N \times F)$?

No, it's wrong, because we are still mixing information among different nodes, which has the same function with adjacent matrix, feature within node does not receive any mixing

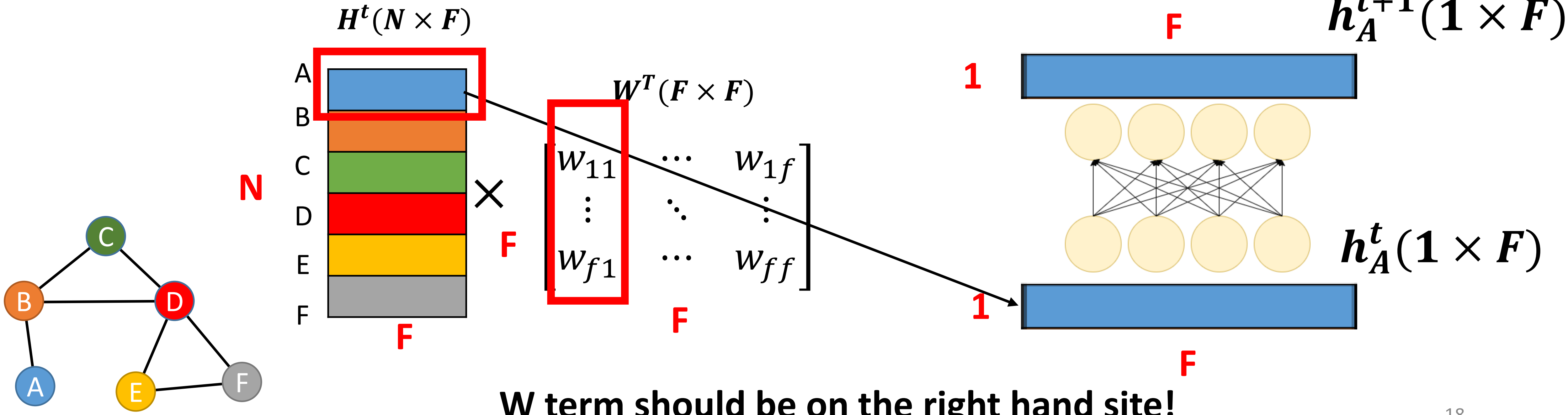


A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution

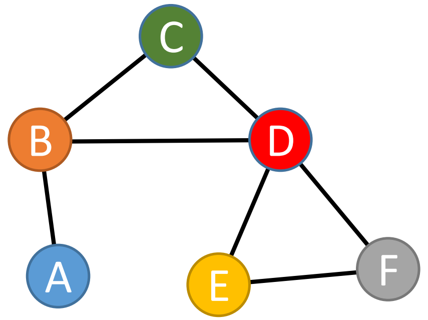
$$\underset{(1\times F)}{h_v^{t+1}} = \sigma\left(\boxed{W_k}\boxed{\sum_{u\in N(v)} \frac{h_u^t}{|N(v)|}}^{(1\times F)} + B_k \overset{(1\times F)}{h_v^t}\right), \forall t \in (0, \dots\dots, T-1)$$

Learnable weight is used to mix information along the feature **within a single node**



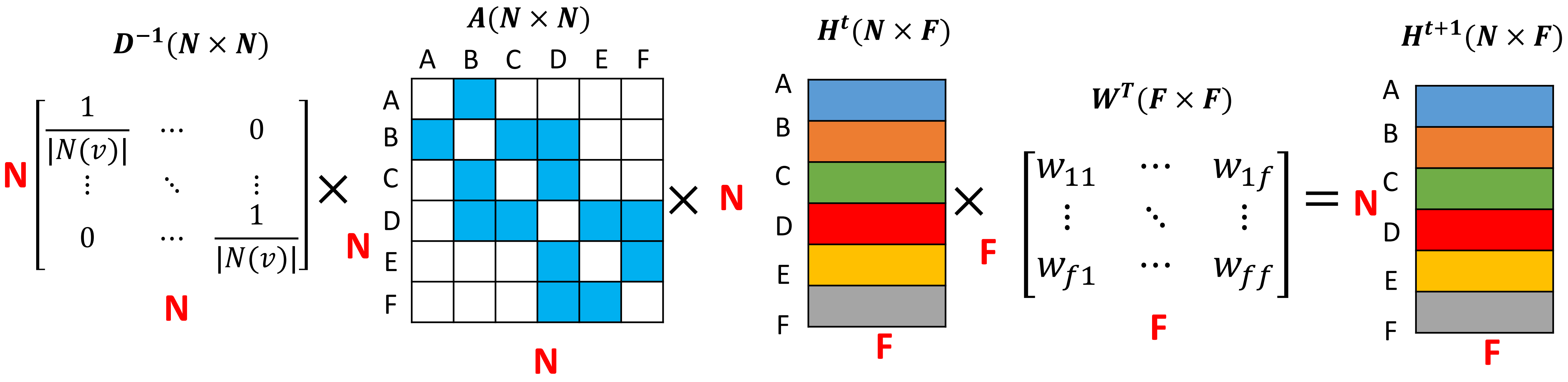
W term should be on the right hand site!

A single layer of GNN: Graph Convolution-Forward



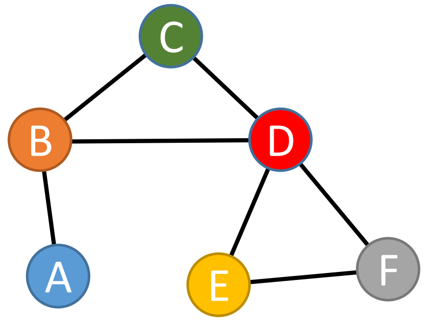
Matrix form for a single layer of graph convolution

$$\overset{(1\times F)}{h_v^{t+1}} = \sigma\left(\boxed{W_k \sum_{u\in N(v)} \overset{(1\times F)}{\frac{h_u^t}{|N(v)|}}} + \overset{(1\times F)}{B_k h_v^t}\right), \forall t \in (\,0, \dots\dots, T-1)$$



A single layer of GNN: Graph Convolution-Forward

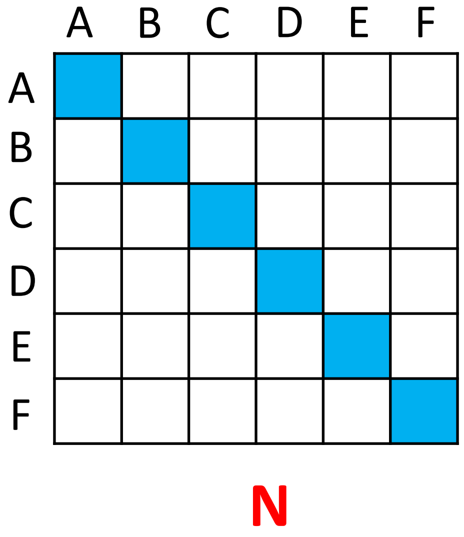
Matrix form for a single layer of graph convolution



$$\underset{(1\times F)}{h_v^{t+1}} = \sigma\left(W_k \sum_{u\in N(v)} \frac{\overset{(1\times F)}{h_u^t}}{|N(v)|} + \boxed{\overset{(1\times F)}{B_k h_v^t}} \right), \forall t \in (0, \dots\dots, T-1)$$

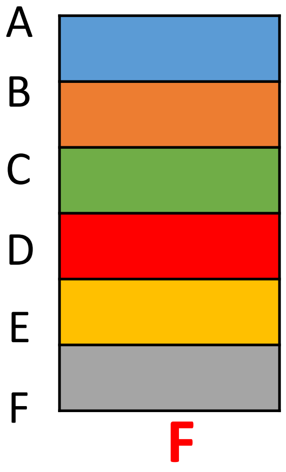
Noted that B^T is a learnable weight matrix

$\boldsymbol{A'(N\times N)}$ Indentity matrix



$\times$ N

$\boldsymbol{H_v^t\ (N\times F)}$



$\times$ F

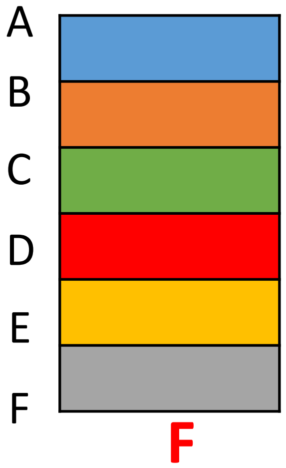
$\boldsymbol{B^T(F\times F)}$

$$\begin{bmatrix} b_{11} & \cdots & b_{1f} \\ \vdots & \ddots & \vdots \\ b_{f1} & \cdots & b_{ff} \end{bmatrix}$$

F

$=$ N

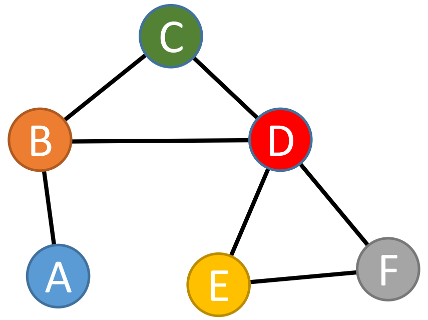
$\boldsymbol{H_v^{t+1}(N\times F)}$



Self loop adjacent matrix is a diagonal matrix!

A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution



$$\overset{(1\times F)}{h_v^{t+1}} = \sigma\left(W_k \sum_{u \in N(v)} \frac{\overset{(1\times F)}{h_u^t}}{|N(v)|} + B_k \overset{(1\times F)}{h_v^t} \right), \forall t \in (\,0, \ldots \ldots, T-1)$$

Now let's rewrite the scalar form above into matrix form

Non-Linear Activation

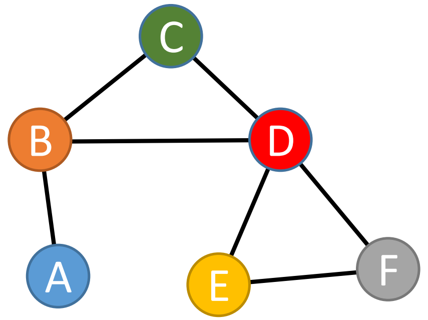
$$H^{t+1} = \sigma(D^{-1}AH^tW^T + A'\overset{(N\times F)}{H_v^t}\overset{(F\times F)}{B^T})$$

Aggregating neighbor node feature

Aggregating self node feature

A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution

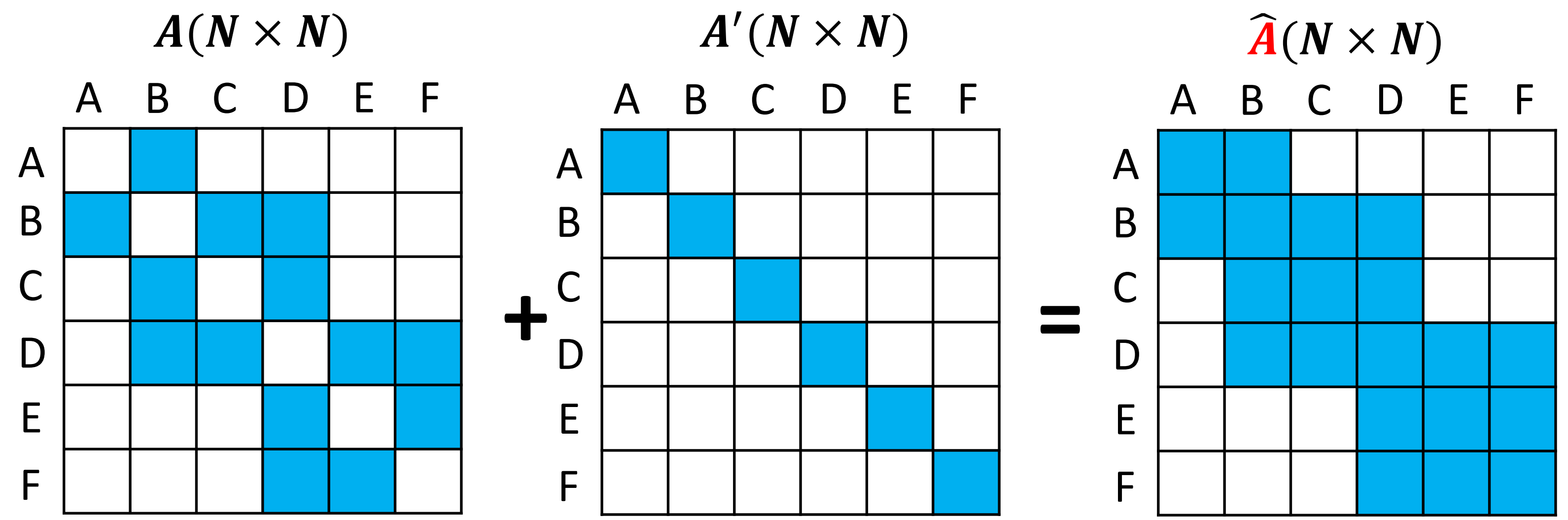


$(\mathbf{1 \times F})$ $(\mathbf{1 \times F})$ $(\mathbf{1 \times F})$

$$h_v^{t+1} = \sigma\left(W_k \sum_{u \in N(v)} \frac{h_u^t}{|N(v)|} + B_k h_v^t\right), \forall t \in (\,0, \dots\dots, T-1)$$

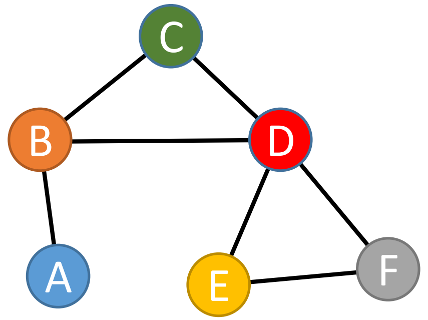
$(\boldsymbol{N \times N})(\boldsymbol{N \times N})(\boldsymbol{N \times F})(\boldsymbol{F \times F})$ $(\boldsymbol{N \times N})$

$$\boldsymbol{H^{t+1} = \sigma(D^{-1}\hat{A}H^{t'}W'^{T})}$$



A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution

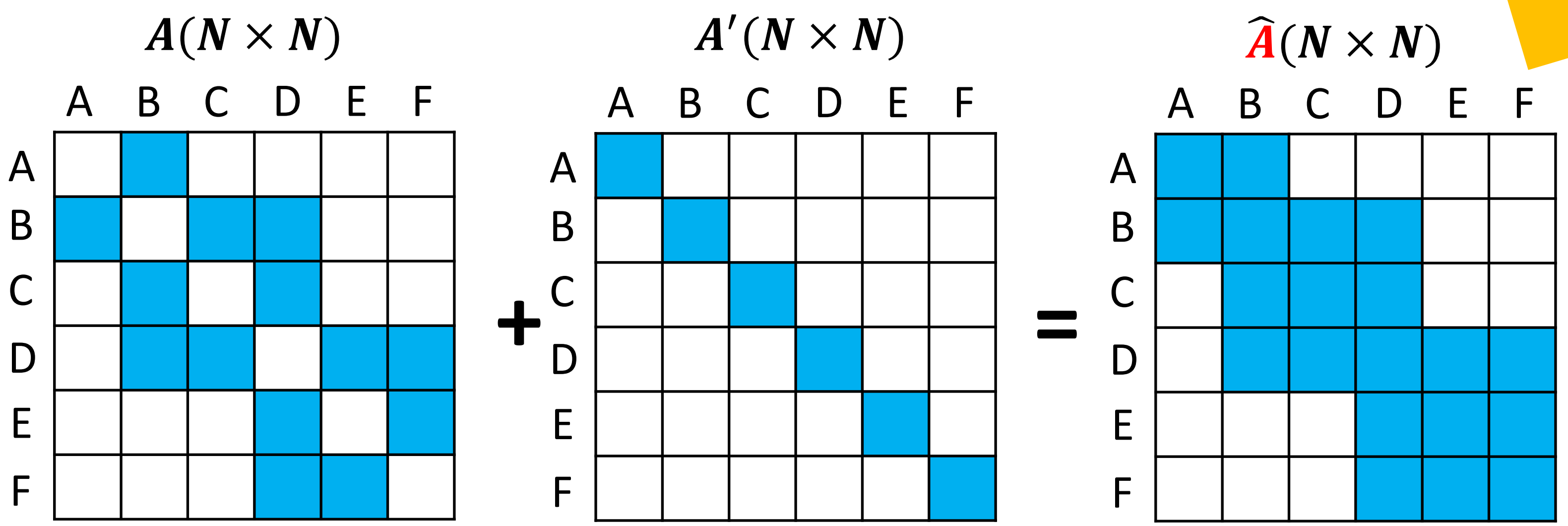


$$\underbrace{h_v^{t+1}}_{(1\times F)} = \sigma\left(W_k \sum_{u \in N(v)} \frac{\overbrace{h_u^t}^{(1\times F)}}{|N(v)|} + B_k \overbrace{h_v^t}^{(1\times F)} \right), \forall t \in (\,0, \ldots\ldots, T-1)$$

$$\boldsymbol{H}^{t+1} = \sigma(\underbrace{\boldsymbol{D}^{-1}}_{(N\times N)}\underbrace{\widehat{\boldsymbol{A}}}_{(N\times N)}\underbrace{\boldsymbol{H}^{t'}}_{(N\times F)}\underbrace{\boldsymbol{W}'^{T}}_{(F\times F)})$$

$(\boldsymbol{N}\times\boldsymbol{N})$

Forward equation for GCN



A single layer of GNN: Graph Convolution-Backward

GCN Backwards:

$$H^{t+1} = \sigma(D^{-1}\hat{A}H^{t}W^{T})$$

$$\overset{4\times2}{Y} = \overset{4\times3}{X}\overset{3\times2}{W} \qquad \frac{\partial L}{\partial Y} \text{ Known} \qquad \frac{\partial L}{\partial X}, \frac{\partial L}{\partial W}?$$

$$\overset{4\times3}{\frac{\partial L}{\partial X}} = \overset{4\times2}{\frac{\partial L}{\partial Y}} \times \overset{2\times3}{\frac{\partial Y}{\partial X}} = \frac{\partial L}{\partial Y} \times W^{T}$$

$$\frac{\partial Y}{\partial X} = W^T$$

$\odot$

$\frac{\partial L}{\partial W}$? $\qquad Y^T = (XW)^T \Rightarrow Y^T = W^T X^T$

$$\frac{\partial L}{\partial W^T} = \frac{\partial L}{\partial Y^T} \times \frac{\partial Y^T}{\partial W^T} = \frac{\partial L}{\partial Y^T} \times X$$

$$\frac{\partial Y^T}{\partial W^T} = (X^T)^T$$

$$\frac{\partial L}{\partial W} = \left(\left(\frac{\partial L}{\partial W}\right)^T\right)^T = \left(\frac{\partial L}{\partial W^T}\right)^T = \left(\overset{2\times4}{\frac{\partial L}{\partial Y^T}} \times \overset{4\times3}{X}\right)^T$$

3×2

2×3

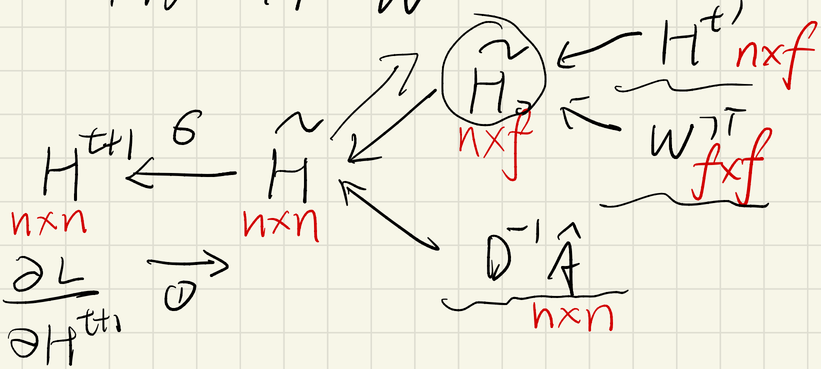
$$\underbrace{H^{t+1}}_{n\times n} = \sigma(\underbrace{D^{-1}}_{n\times n}\underbrace{\hat{A}}_{n\times n}\underbrace{H^{t'}}_{n\times f}\underbrace{W'^{T}}_{f\times f})$$

n : # nodes

f : # node features

Let $\tilde{H} = D^{-1}\hat{A}H^{t'}W'^{T}$ $\frac{\partial L}{\partial H^{t'}}, \frac{\partial L}{\partial W'^{T}}$?

Let $\tilde{H}_0 = H^{t'}W'^{T}$



① $\frac{\partial L}{\partial \tilde{H}} = \frac{\partial L}{\partial H^{t+1}} \odot \frac{\partial H^{t+1}}{\partial \tilde{H}}$

② $\frac{\partial L}{\partial \tilde{H}_0}$ $\tilde{H} = (D^{-1}\hat{A})\,\tilde{H}_0$

$$\frac{\partial L}{\partial \tilde{H}_0} = \left(\frac{\partial L}{\partial \tilde{H}} \times D^{-1}\hat{A}\right)^{T}$$

③ $\tilde{H}_0 = H^{t'}\,W'^{T}$

$$\frac{\partial L}{\partial H^{t'}} = \left(\frac{\partial L}{\partial \tilde{H}^{T}} \times (D^{-1}\hat{A})\right)^{T} \times W'$$

$$\frac{\partial L}{\partial W'^{T}} = \left(\frac{\partial L}{\partial \tilde{H}^{T}} \times (D^{-1}\hat{A}) \times H^{t'}\right]^{T}$$