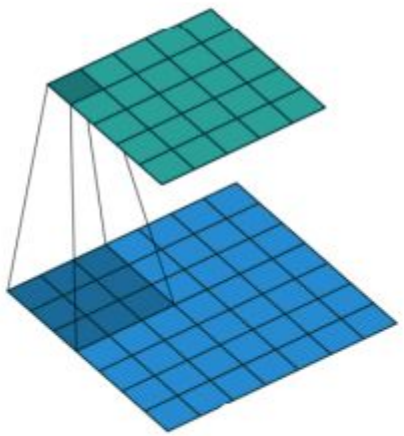
Recitation: Graph Neural Networks

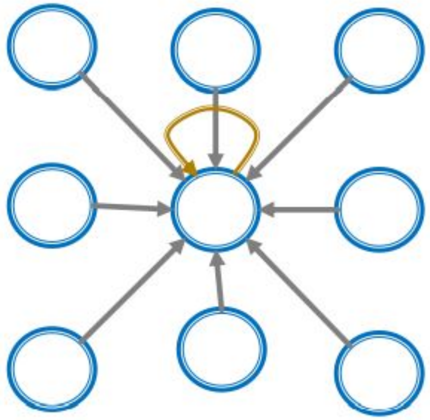
- **Quickly review GCN message passing process**
- **Graph Convolution layer forward**
- **Graph Convolution layer backward**
- **GCN code example**

A single layer of GNN: Graph Convolution

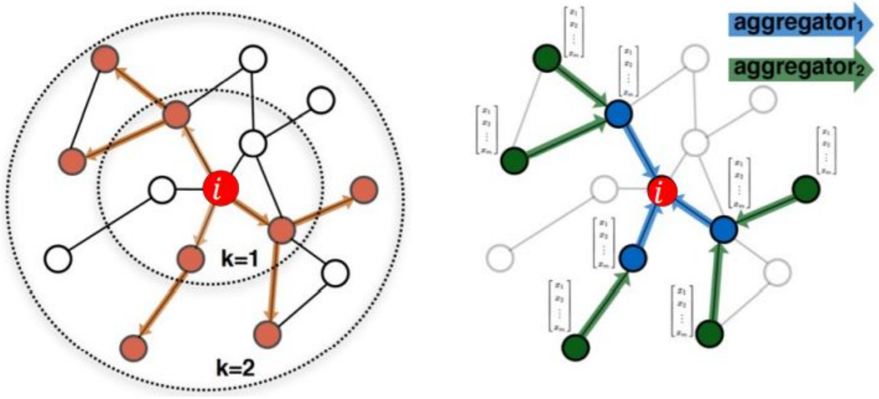
Key idea: Node’s neighborhood defines a computation graph



CNN: pixel convolution



CNN: pixel convolution



GNN: graph convolution

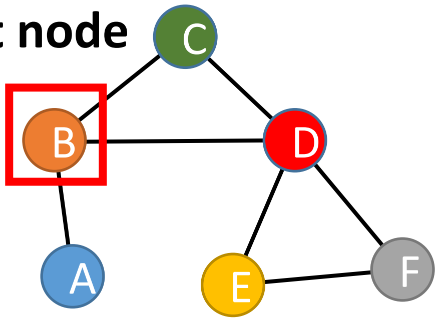
- **Learning a node feature by propagating and aggregating neighbor information!**
- **Node embedding can be defined by local network neighborhoods!**

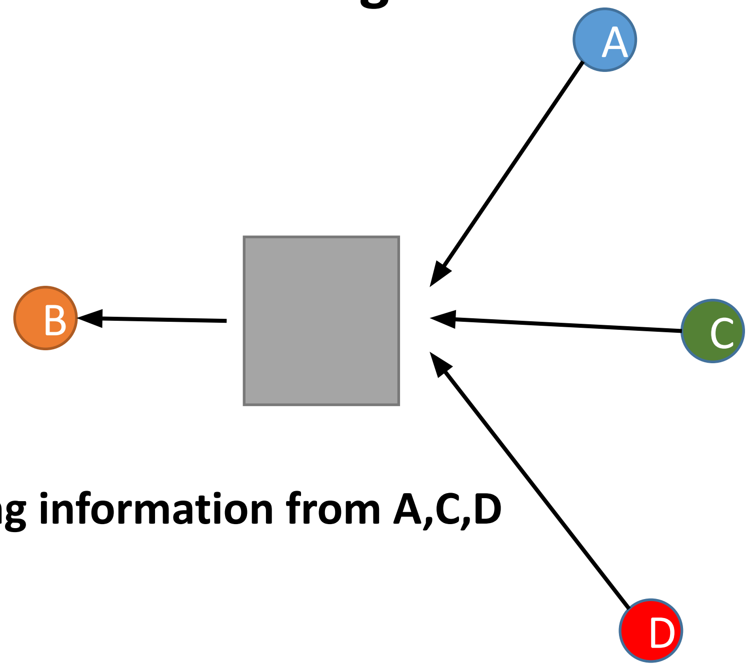
A single layer of GNN: Graph Convolution

Key idea: Generate node embedding based on local network neighborhoods

Considering 1 step of feature aggregation of the nearest neighbor

Target node



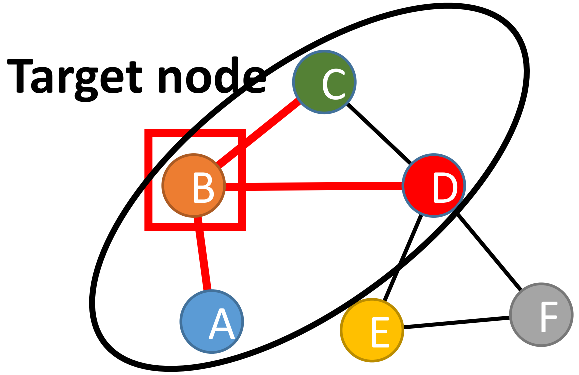


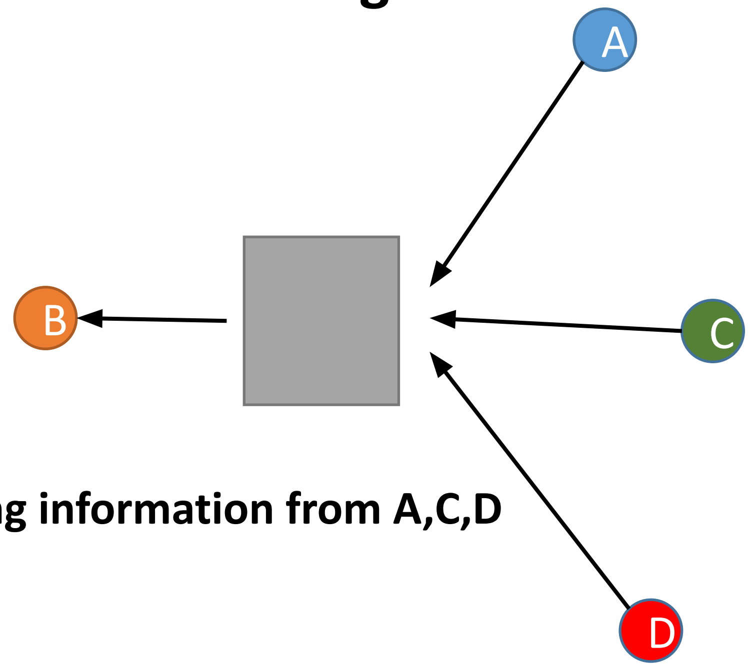
Processing information from A,C,D

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



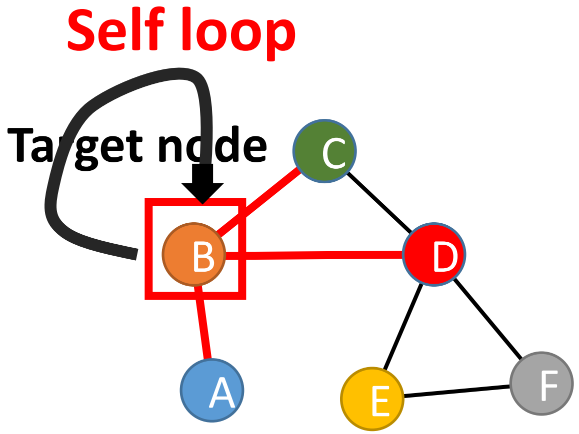


Processing information from A,C,D

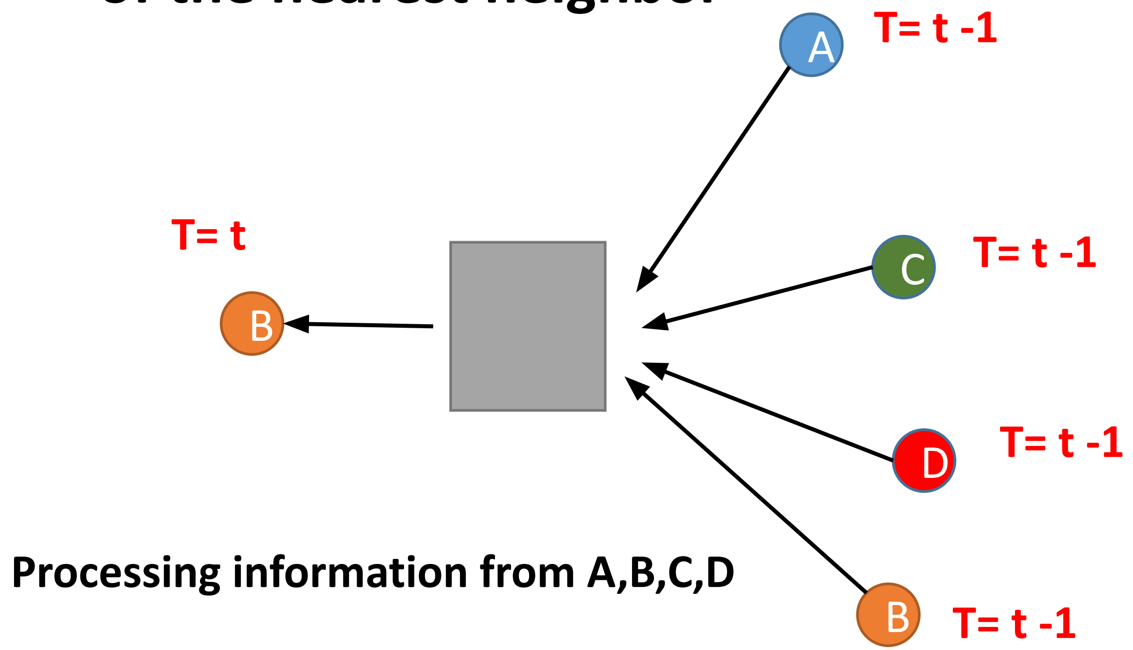
Now B have the information from it's first nearest neighbors

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

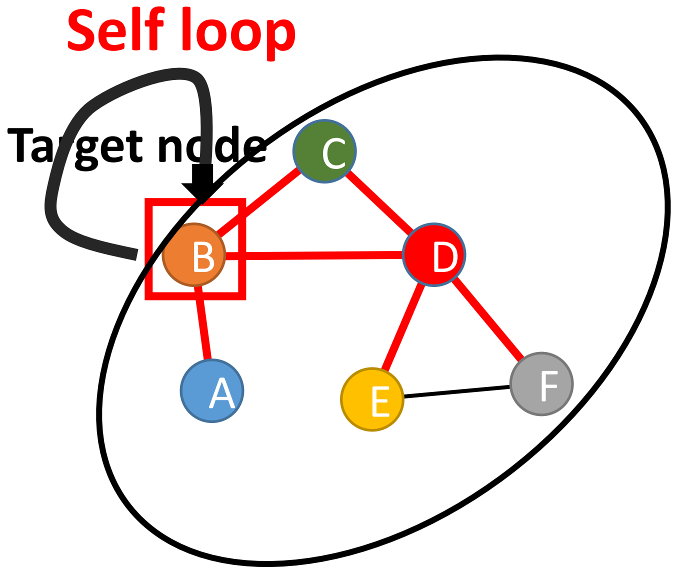


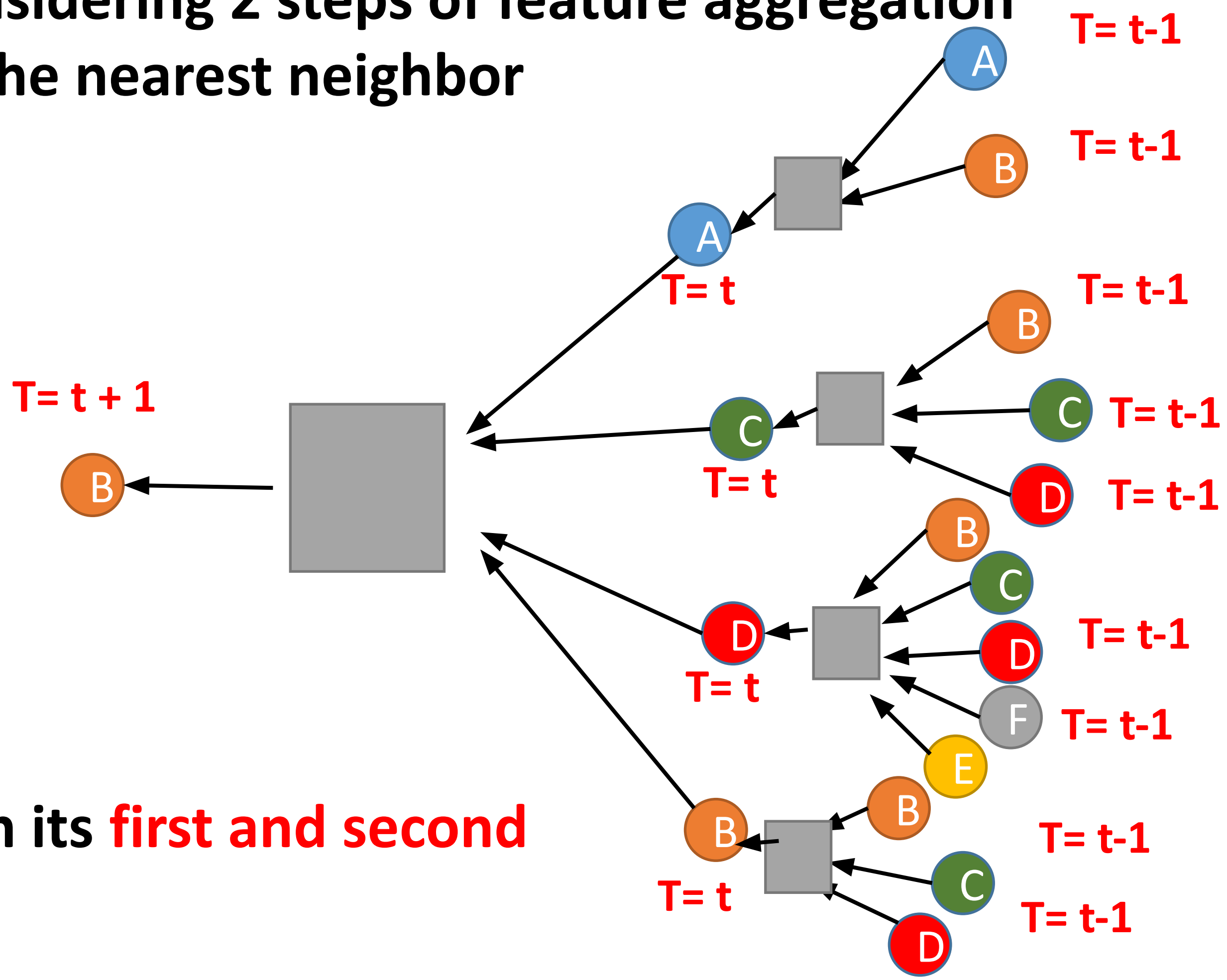
Also we don't want to lose information from B itself

A single layer of GNN: Graph Convolution

Key idea: Generate node embedding based on local network neighborhoods

Considering 2 steps of feature aggregation of the nearest neighbor



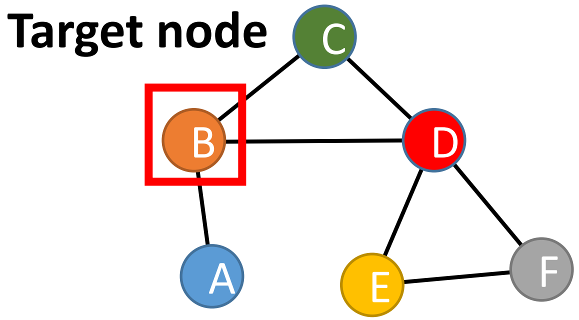


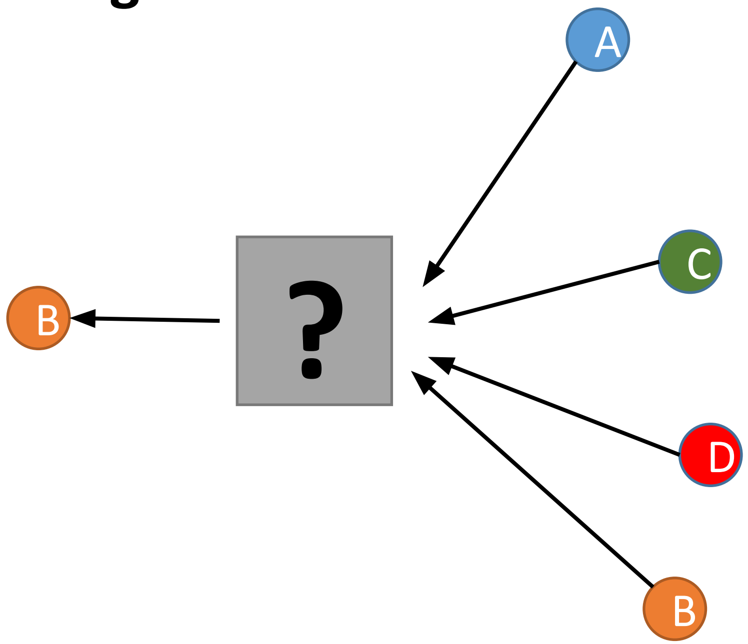
Now B have the information from its first and second nearest neighbors

A single layer of GNN: Graph Convolution

Key idea: Generate node embedding based on local network neighborhoods

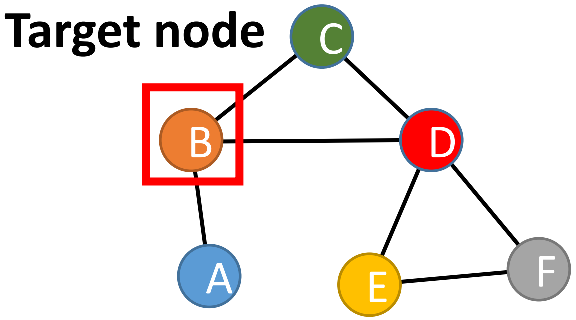
How to process and mix the information from neighbor?



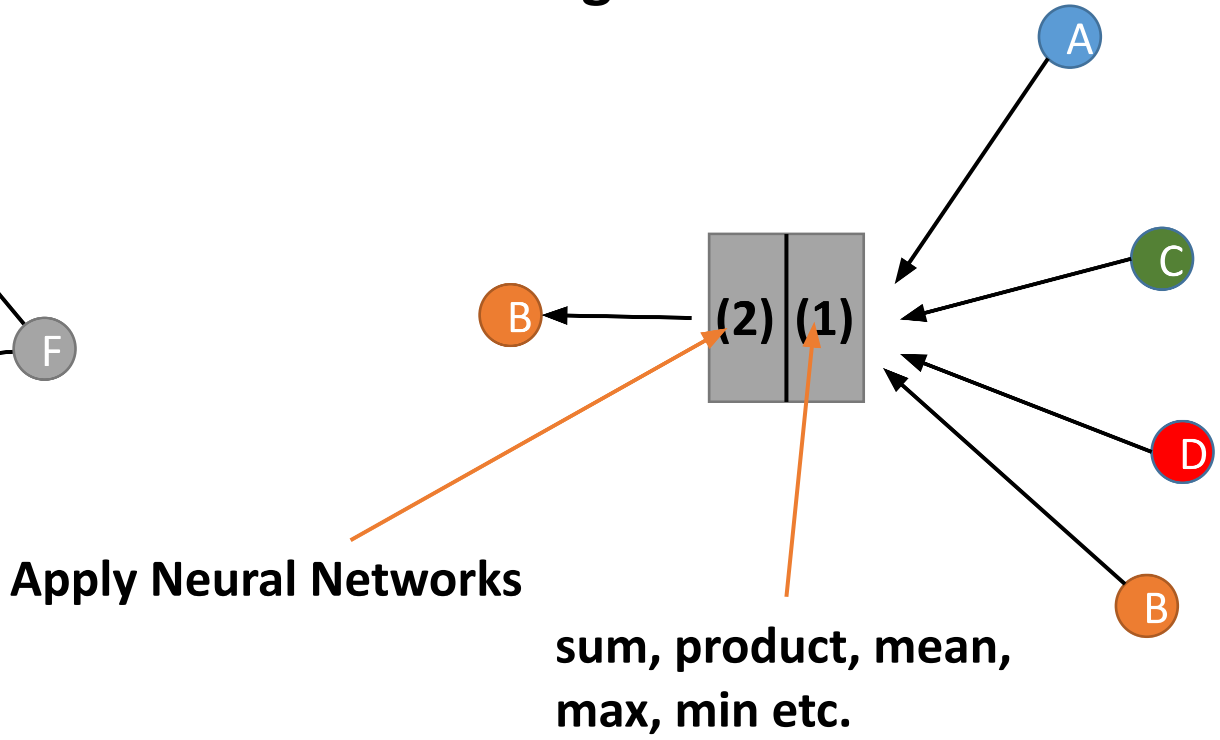


A single layer of GNN: Graph Convolution

Key idea: Generate node embedding based on local network neighborhoods



How to process and mix the information from neighbor?

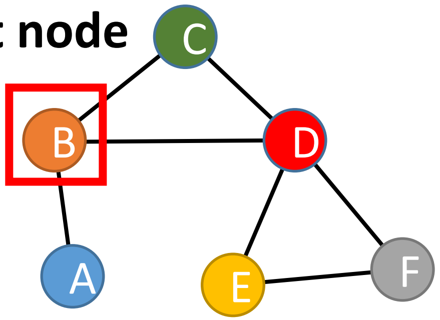


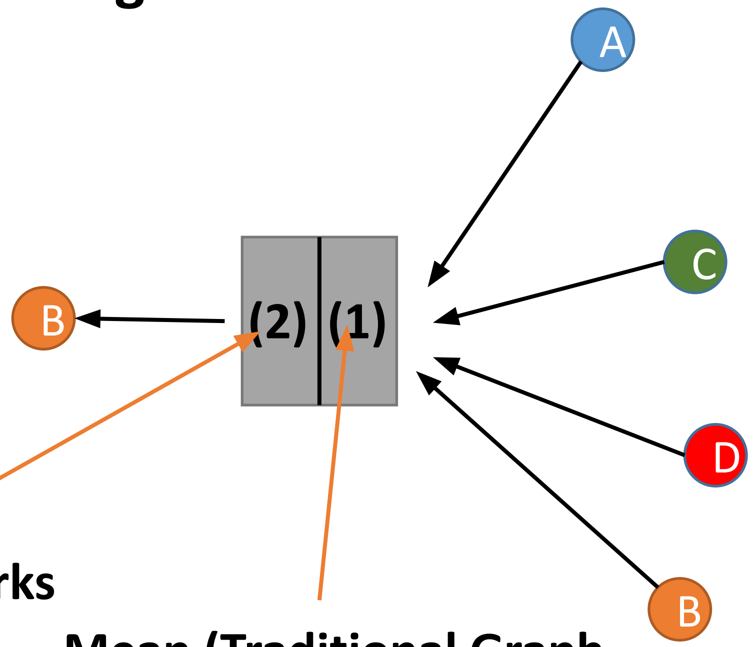
A single layer of GNN: Graph Convolution

Key idea: Generate node embedding based on local network neighborhoods

How to process and mix the information from neighbor?

Target node





Apply Neural Networks

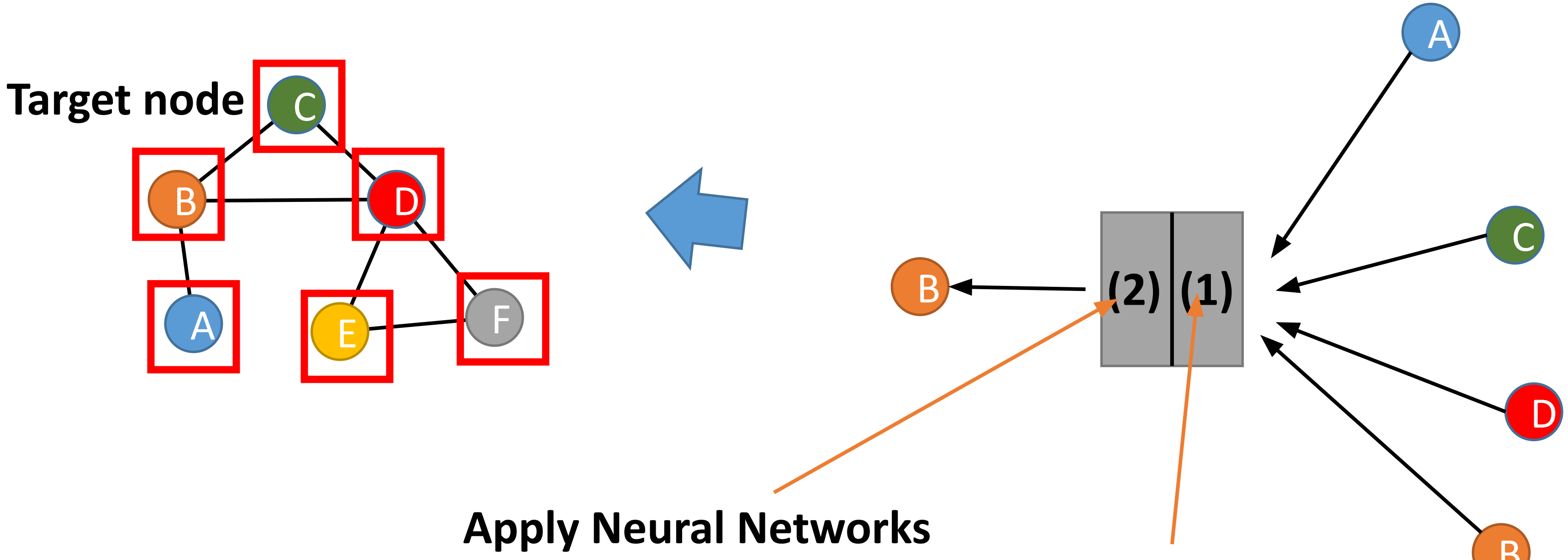
Mean (Traditional Graph Convolutional Neural Networks(GCN))

[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)



Mean (Traditional Graph Convolutional Neural Networks(GCN))

[Kipf and Welling, ICLR 2017]

A single layer of GNN: Graph Convolution-Forward

Math for a single layer of graph convolution

$$\boxed{h_v^0 = x_v^{(\mathbf{1}\times \boldsymbol{F})}}$$

0th layer embedding = node v initial feature

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

Number of time(layers)

Node v feature at time(layer) t+1

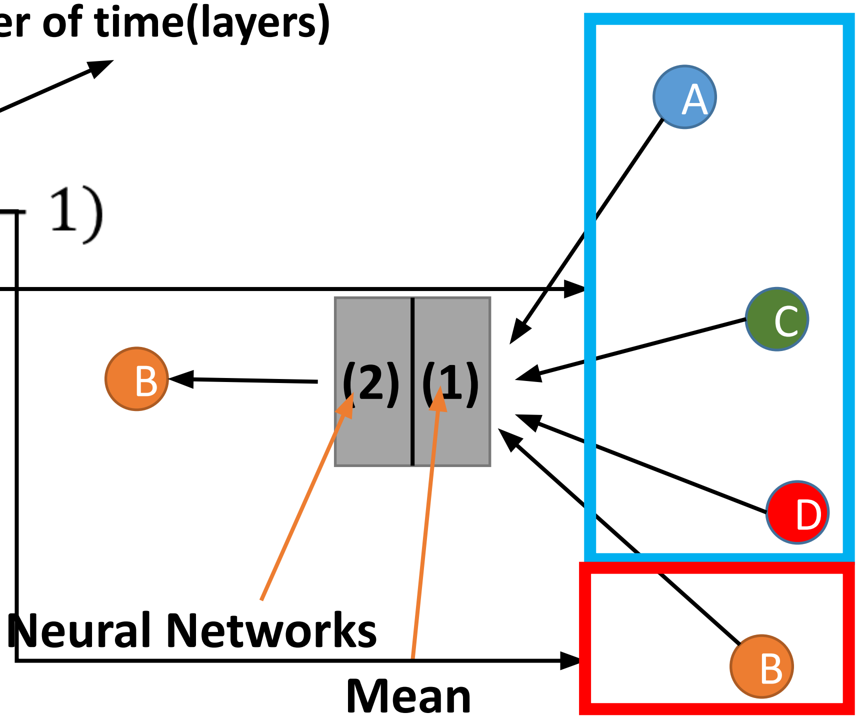
Non-linear activation (i.e. relu())

Learnable weight

Average the neighbor node feature at time(layer) t

Learnable weight

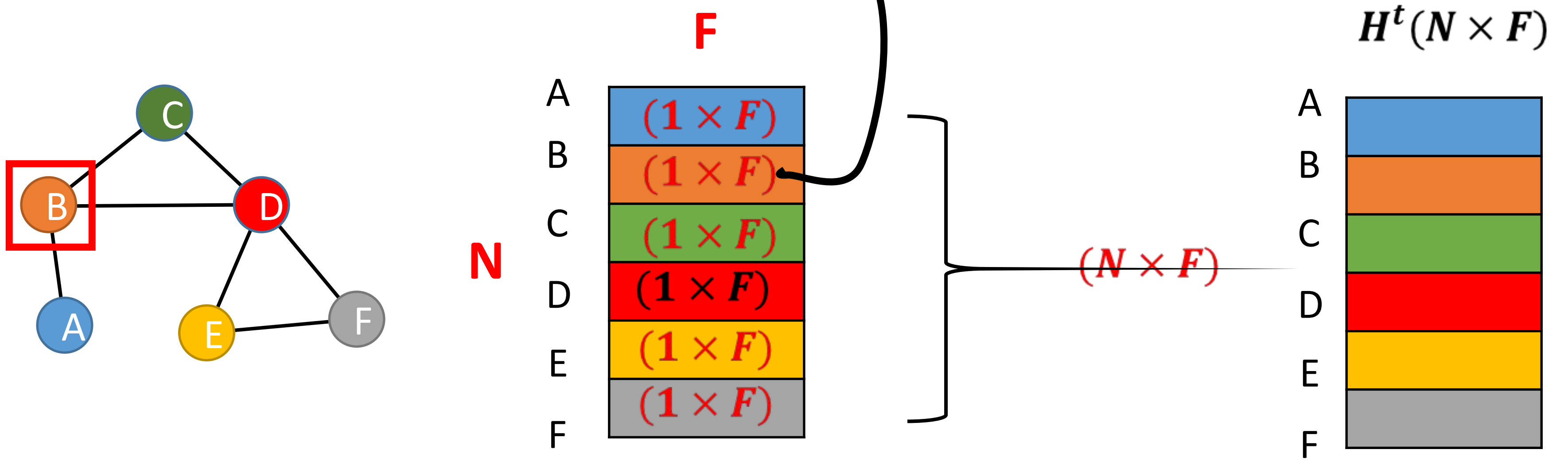
Node v feature at time(layer) t



A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution

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



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

A single layer of GNN: Graph Convolution-Forward

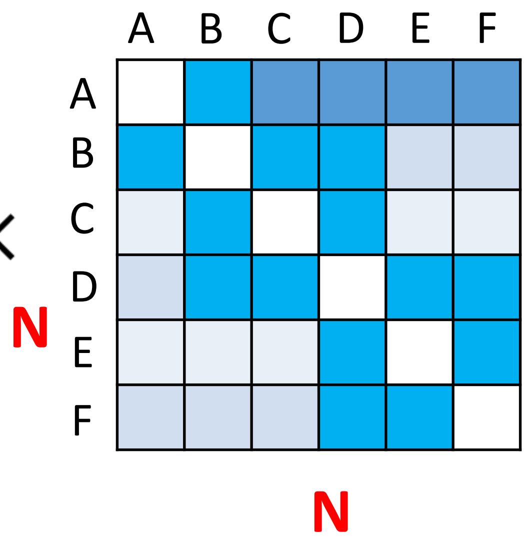
Matrix form for a single layer of graph convolution

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

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

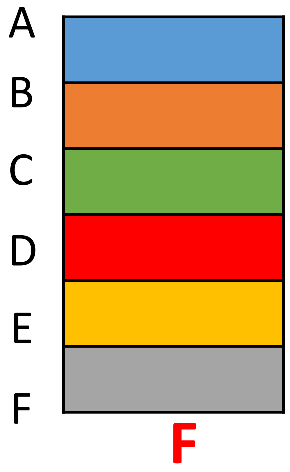
$$\mathbf{N}\underbrace{\begin{bmatrix} \frac{1}{|N(v)|} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{|N(v)|} \end{bmatrix}}_{\mathbf{N}} \times$$

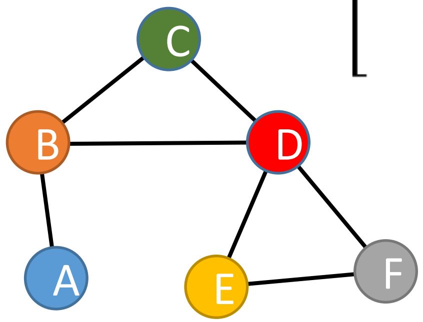
$\boldsymbol{A}(\boldsymbol{N}\times\boldsymbol{N})$



$\times$ **N**

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





A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution

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

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

Noted that W^T is a learnable weight matrix

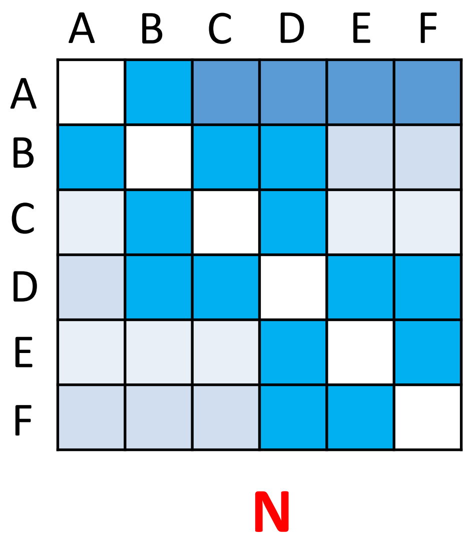
$\boldsymbol{D}^{-1}(\boldsymbol{N}\times\boldsymbol{N})$

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

N

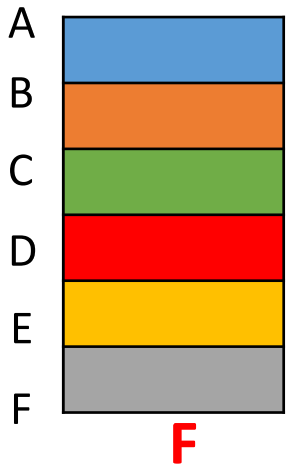
$\times$

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



$\times$ N

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

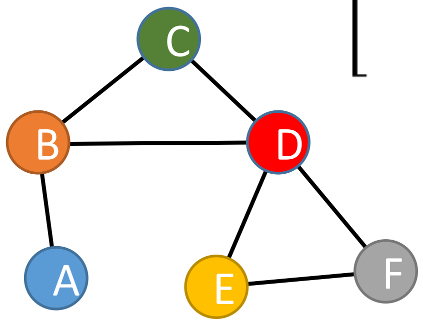


$\times$ F

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

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

F



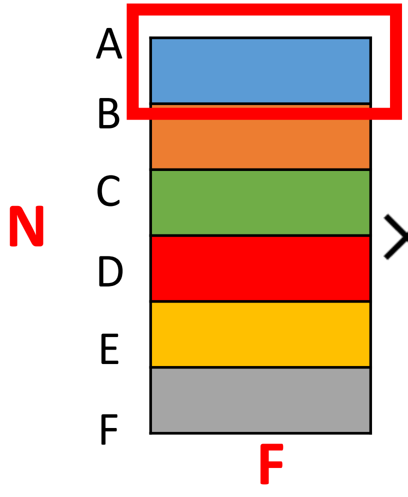
A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution

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

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

$H^t (N \times F)$



$\times$

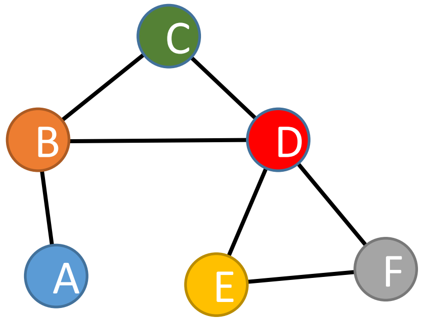
$W^T (F \times F)$

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

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

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

$W^T(N \times N)$

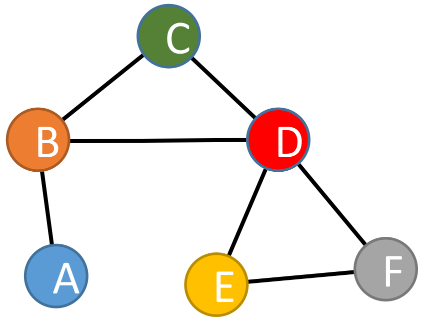
$$\mathbf{N}\quad \underbrace{\begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{n1} & \cdots & w_{nn} \end{bmatrix}}_{\mathbf{N}} \times \quad \mathbf{N}$$

$H^t(N \times F)$

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

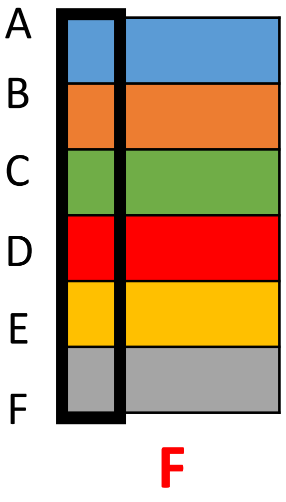
Matrix form for a single layer of graph convolution

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

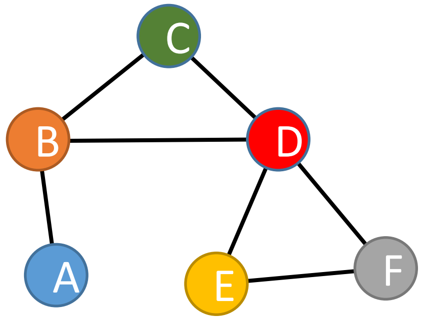
Seems like nothing goes wrong, the result matrix shape is still $(N \times F)$?

$$\begin{array}{cc} & W^T(N\times N) \\ N & \begin{bmatrix} \boxed{w_{11} \quad \cdots \quad w_{1n}} \\ \vdots \quad \ddots \quad \vdots \\ w_{n1} \quad \cdots \quad w_{nn} \end{bmatrix} \\ & N \end{array} \times \quad N$$

$H^t(N \times F)$



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

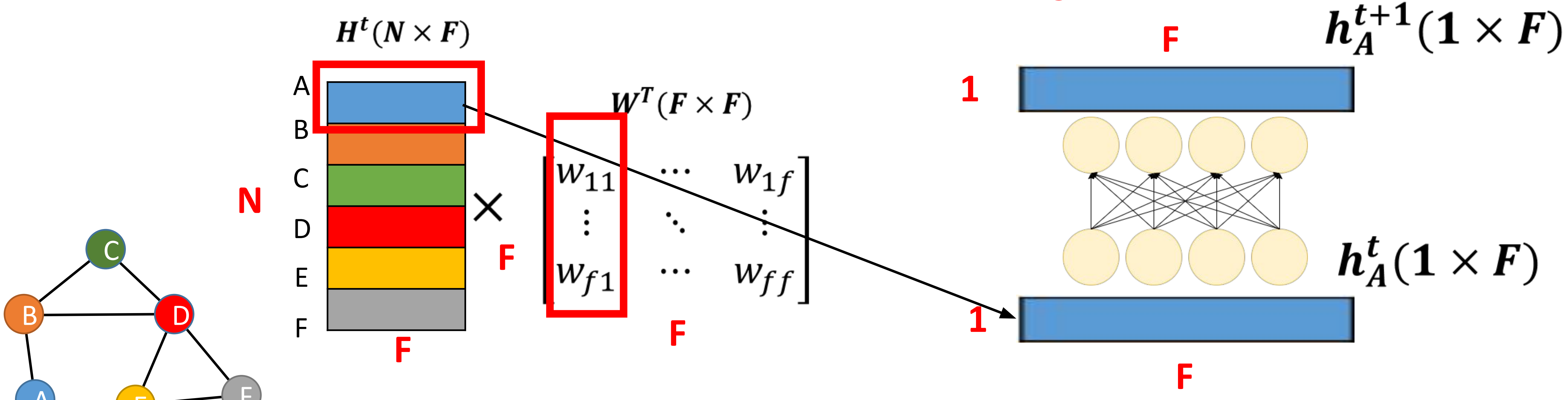


A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution

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

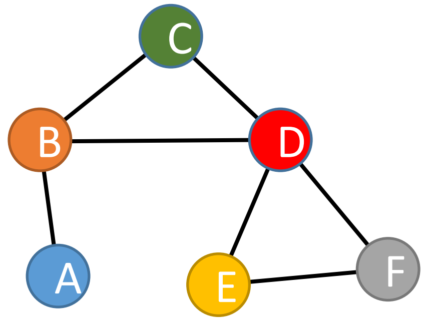
Learnable weight is used to mix information along the feature within a single node



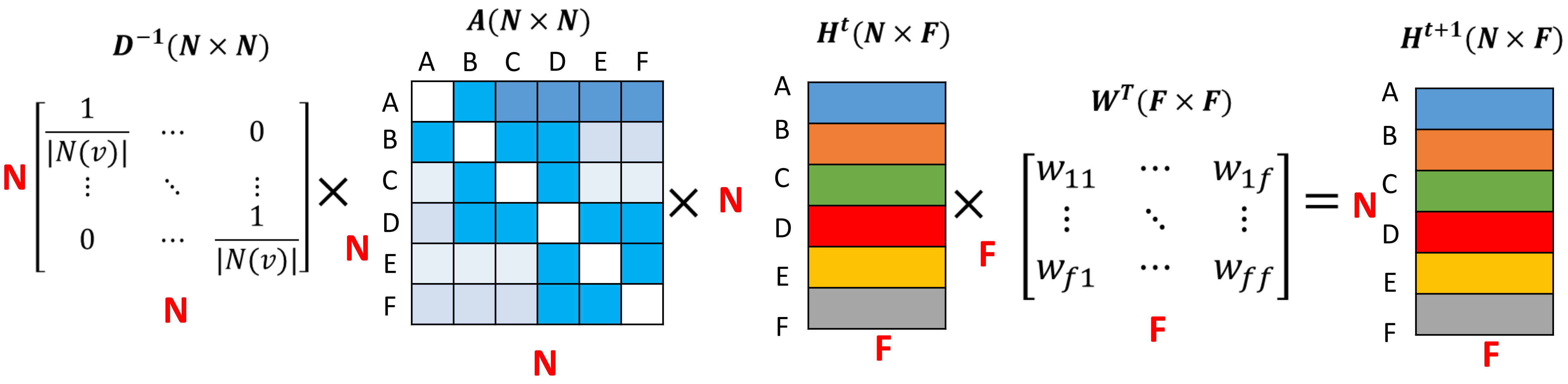
W term should be on the right hand site!

A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution

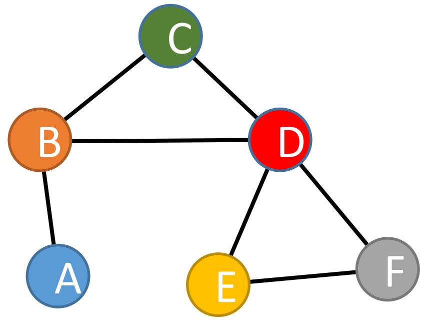


$$\overset{(1\times F)}{h_v^{t+1}} = \sigma\left(\boxed{W_k \sum_{u\in N(v)} \overset{(1\times F)}{\frac{h_u^t}{|N(v)|}}} + \overset{(1\times F)}{B_k h_v^t}\right), \forall t \in (\,0, \ldots\ldots, T-1)$$



A single layer of GNN: Graph Convolution-Forward

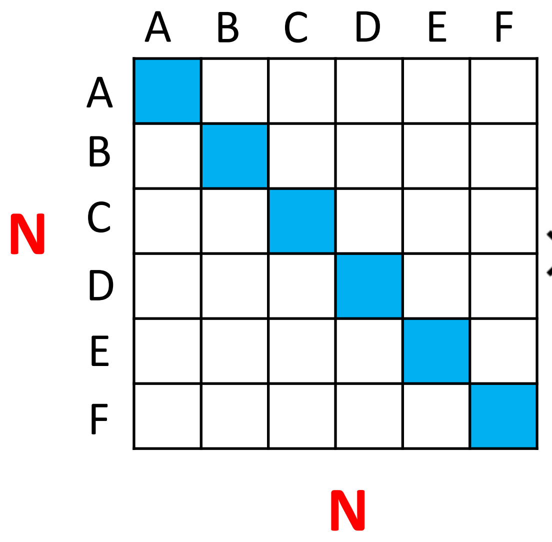
Matrix form for a single layer of graph convolution



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

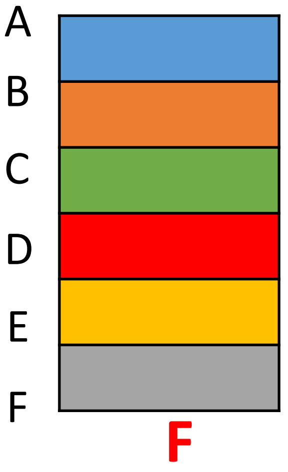
Noted that B^T is a learnable weight matrix

$A'(N \times N)$ Indentity matrix



$\times$ N

$H_v^t \ (N \times F)$



$\times$ F

$B^T(F \times F)$

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

F

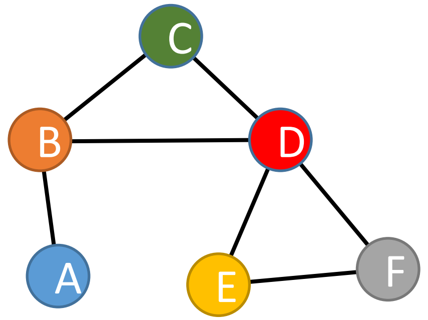
$=$ N

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

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



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

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

Non-Linear Activation

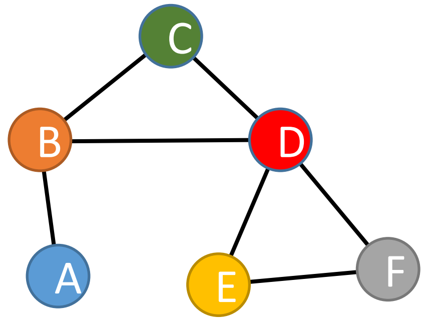
$$H^{t+1} = \sigma(D^{-1}AH^tW^T + A'\overset{(N\times F)(F\times F)}{H_v^t B^T})$$

Aggregating neighbor node feature

Aggregating self node feature

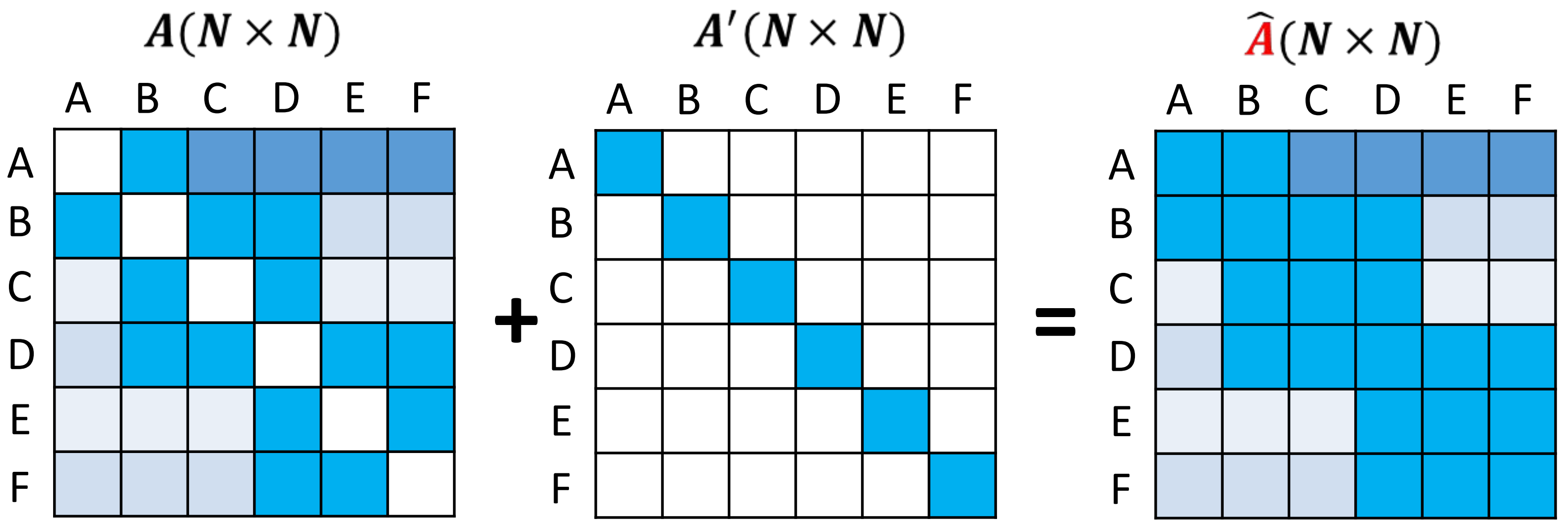
A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution



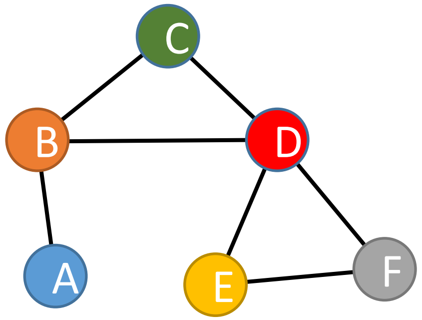
$$\overset{(1\times F)}{h_v^{t+1}} = \sigma\left(W_k \sum_{u \in N(v)} \frac{\overset{(1\times F)}{h_u^t}}{|N(v)|} + B_k \overset{(1\times F)}{h_v^t} \right), \forall t \in (0, \ldots\ldots, T-1)$$

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



A single layer of GNN: Graph Convolution-Forward

Matrix form for a single layer of graph convolution

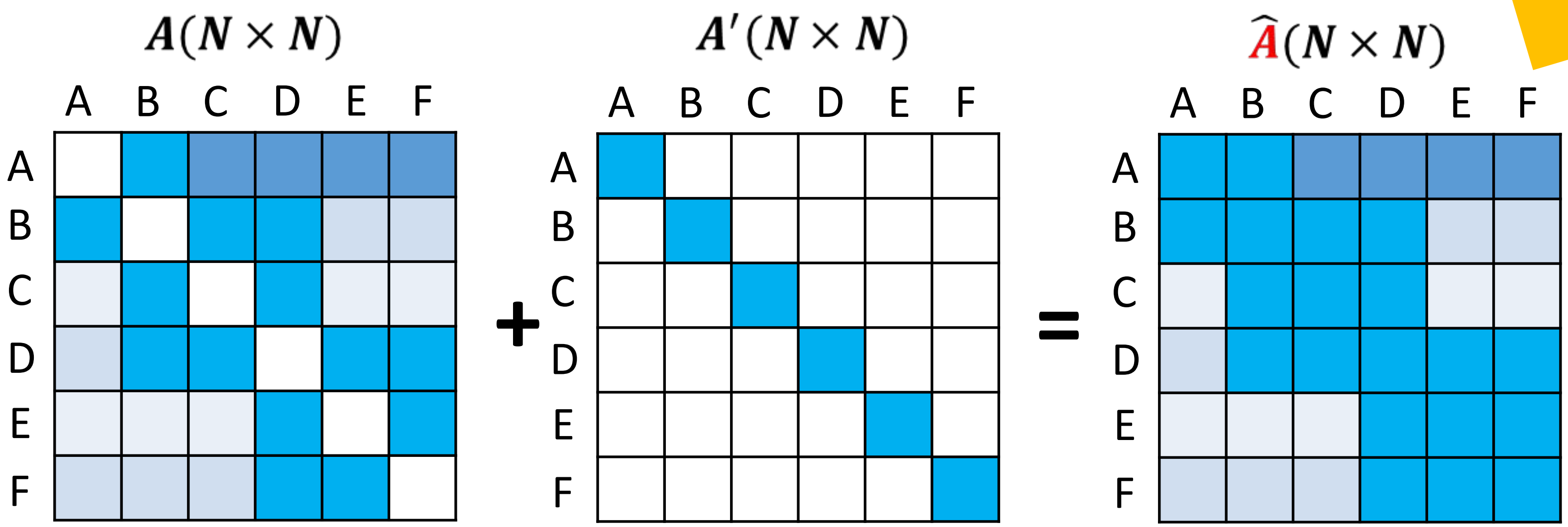


$$\overset{(1\times F)}{h_v^{t+1}} = \sigma\left(W_k \sum_{u \in N(v)} \frac{\overset{(1\times F)}{h_u^t}}{|N(v)|} + B_k \overset{(1\times F)}{h_v^t} \right), \forall t \in (0, \dots\dots, T-1)$$

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

$(N\times N)(N\times N)(N\times F)(F\times F)$

Forward equation for GCN



A single layer of GNN: Graph Convolution-Backward

First, let's recall…

Y = X @ W

4x2 4x3 3x2

How to express dL/dX and dL/dW with dL/dY?

A single layer of GNN: Graph Convolution-Backward

First, let's recall…

Y = X @ W

4x2 4x3 3x2

How to express dL/dX and dL/dW with dL/dY?

dL/dX = dL/dY @ dY/dX

= dL/dY @ W^T

4x2 2x3

A single layer of GNN: Graph Convolution-Backward

First, let's recall…

$$\underset{4\times2}{Y} = \underset{4\times3}{X} \;@\; \underset{3\times2}{W}$$

How to express dL/dX and dL/dW with dL/dY?

$$dL/dX = dL/dY \;@\; dY/dX$$
$$= \underset{4\times2}{dL/dY} \;@\; \underset{2\times3}{W^T}$$

What about dL/dW?

Notice $Y^T = W^T @ X^T$

So $dL/dW^T = dL/dY^T @ dY^T/dW^T$

$= dL/dY^T @ (X^T)^T$

$= dL/dY^T @ X$

So $dL/dW = ((dL/dW)^T)^T = (dL/dW^T)^T$

$$= (\underset{2\times4}{dL/dY^T} \;@\; \underset{4\times3}{X})^T$$

A single layer of GNN: Graph Convolution-Backward

Now, let's derive the backward equation

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

NxF NxN NxN NxF FxF

N: # of nodes
F: # of features

Let's define

$H^{\sim} = D^{-1}A^{\wedge}H^{t'}W'^{T}$ (what's inside the brackets)

$H_0^{\sim} = H^{t'}W'^{T}$

Want to derive:

$dL/dH^{t'}$

dL/dW'^{T}

A single layer of GNN: Graph Convolution-Backward

Let's draw the computational graph

With these definitions

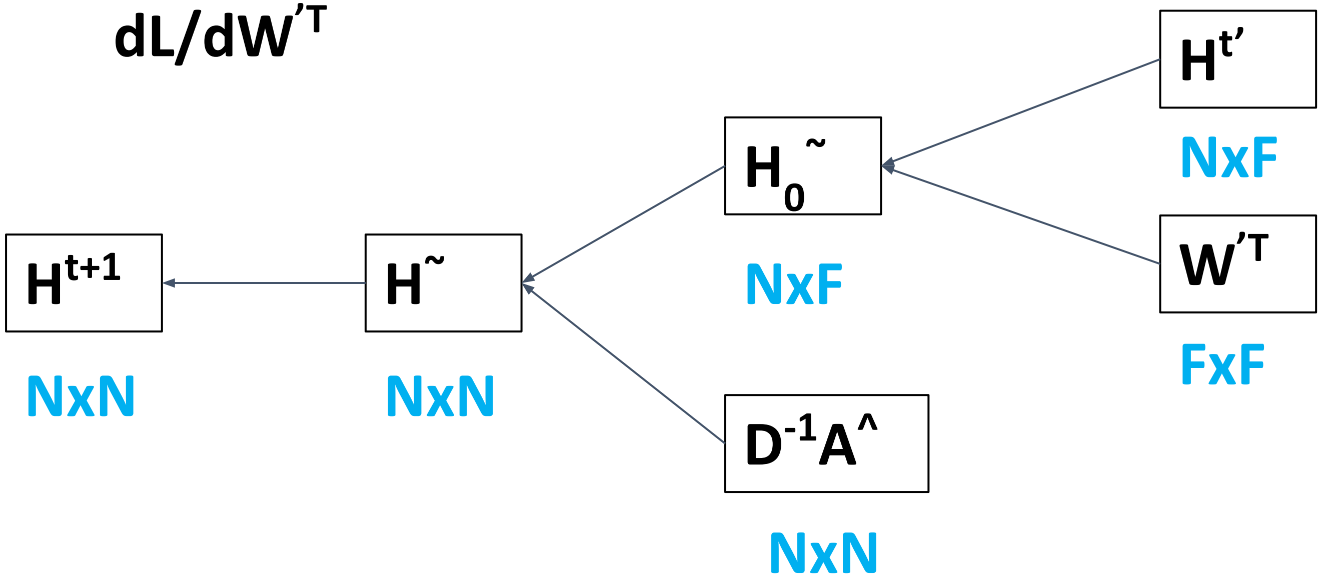
$H^{\sim} = D^{-1}A^{\wedge}H^{t'}W'^{T}$ (what's inside the brackets)

$H_0^{\sim} = H^{t'}W'^{T}$

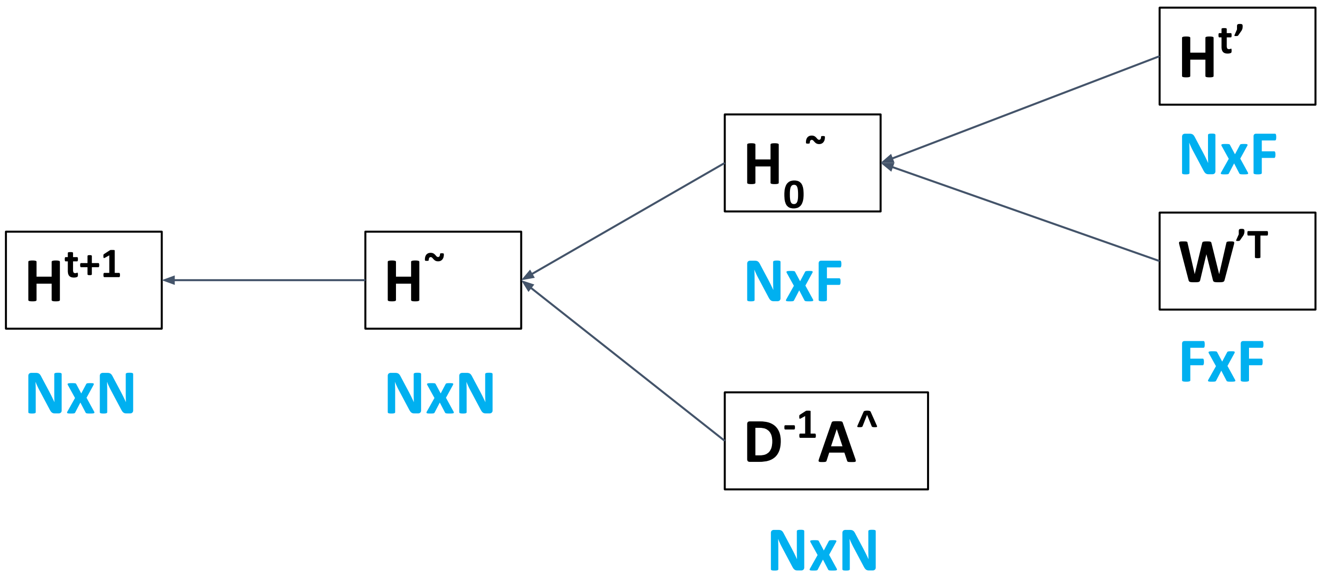
Want to derive:

$dL/dH^{t'}$

dL/dW'^{T}



A single layer of GNN: Graph Convolution-Backward



$dL/dH^{\sim} = dL/dH^{t+1} * dH^{t+1}/dH^{\sim}$

$dL/dH_0^{\sim} = (dL/dH^{\sim} @ (D^{-1}A^{\wedge}))^T$ (recall that $H^{\sim} = (D^{-1}A^{\wedge})H_0^{\sim}$)

Now also recall that $H_0^{\sim} = H^{t'}W'^{T}$

$dL/dH^{t'} = (dL/dH^{\sim} @ (D^{-1}A^{\wedge}))^T @ W'$

$dL/dW'^{T} = (dL/dH^{\sim} @ (D^{-1}A^{\wedge}) @ H^{t'})^T$