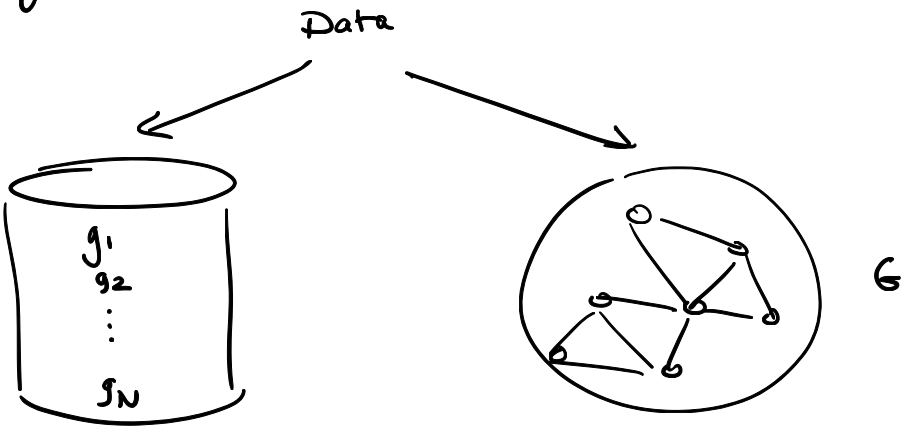
GNNs : Graph Neural Networks

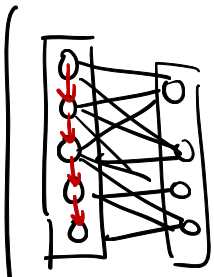


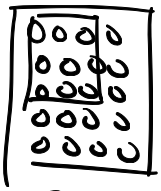
Tasks

1) graph classification/ regression

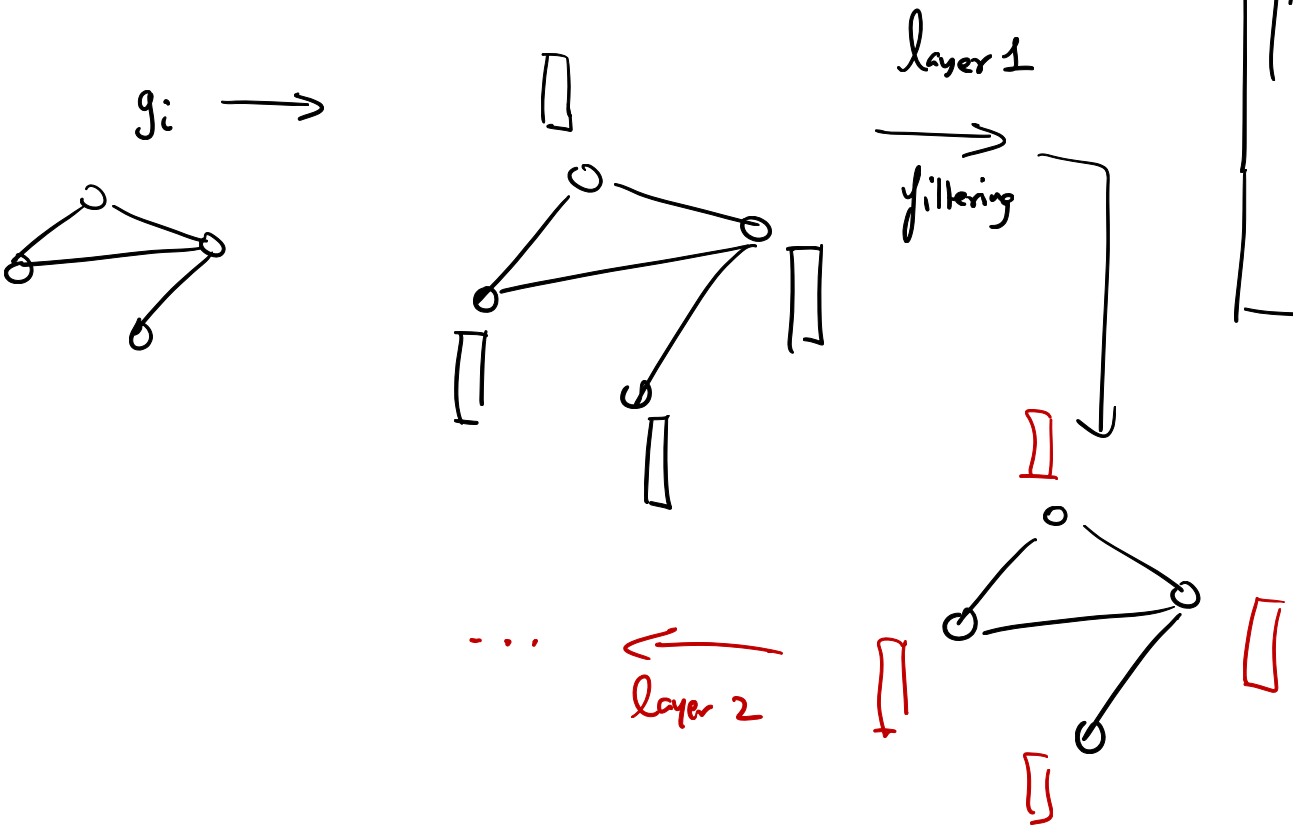
2) node level class/regression

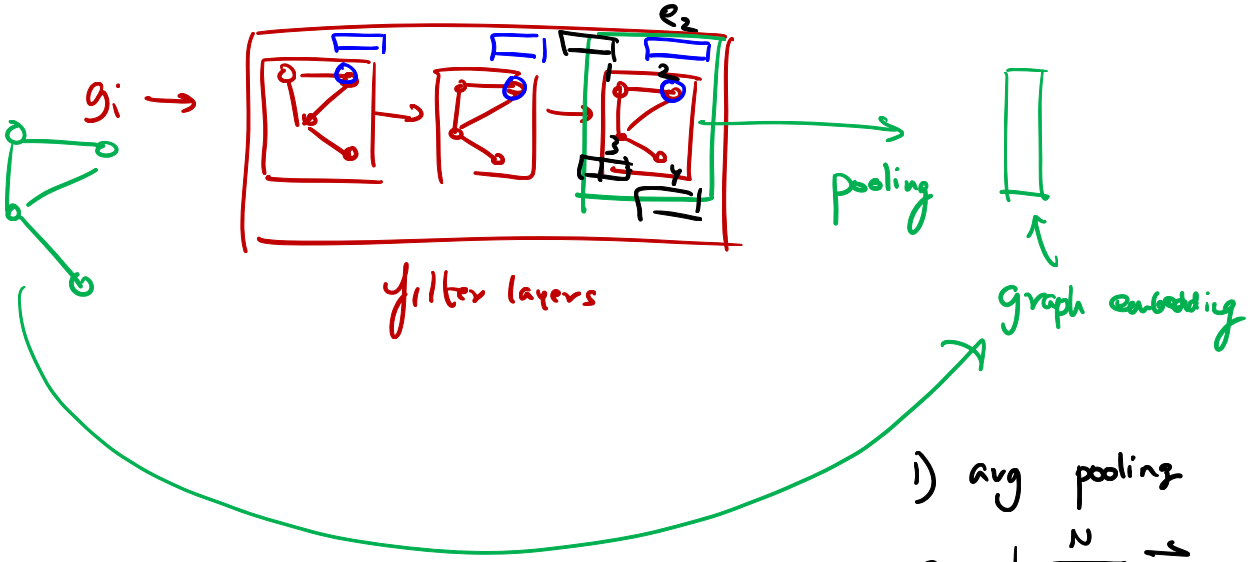
3) edge/ prediction link





CNN: 'regular' graph implicit

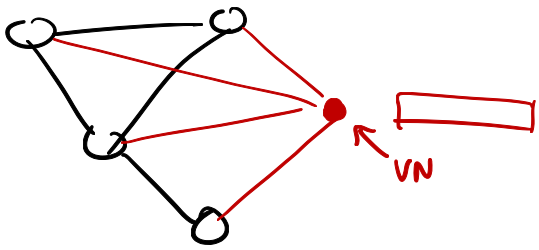




1) avg pooling

$$g = \frac{1}{N} \sum_{i=1}^{N} \vec{e}_i$$

2) Virtual node (CLS)



after filtering

$\vec{g} = \vec{e}_{VN}$: is the embedding for the graph

2) weighted pooling

$$= \sum_{i=1}^{N} f(e_i \cdot W)$$

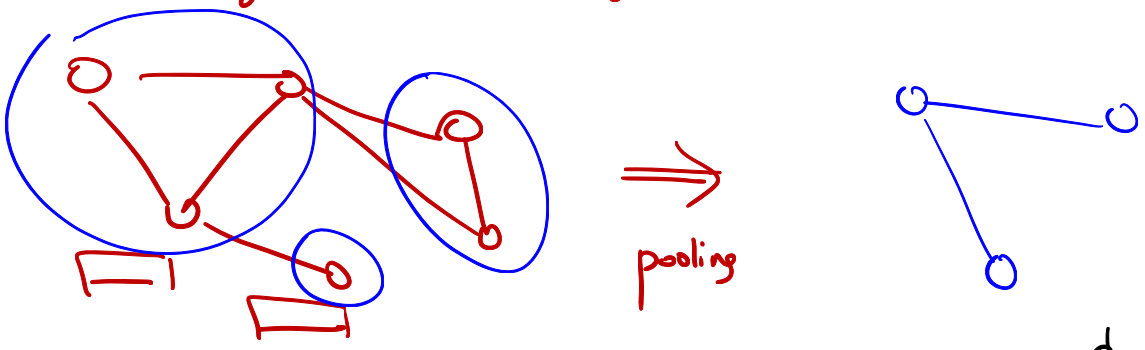
learnable matrix

$e_i \in \mathbb{R}^{d_{in}}$

$g \in \mathbb{R}^{d_{out}}$

$W: d_{in} \times d_{out}$

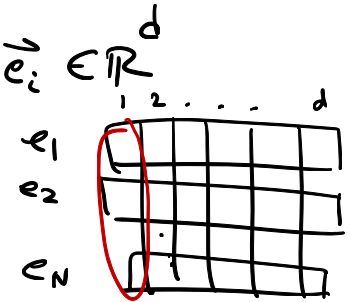
Pooling → Coarsening



4) max pooling $\vec{g} = (g_1 g_2 \dots g_d)^T$

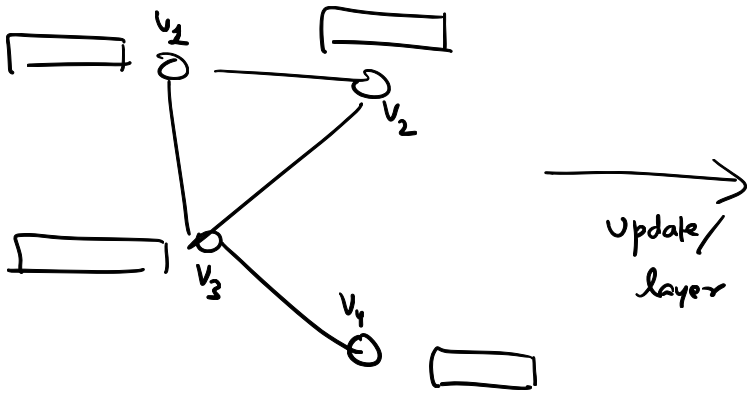
$$g_i = \max_{j=1}^{N} \{ e_{ji} \}$$

$\vec{e}_i \in \mathbb{R}^d$



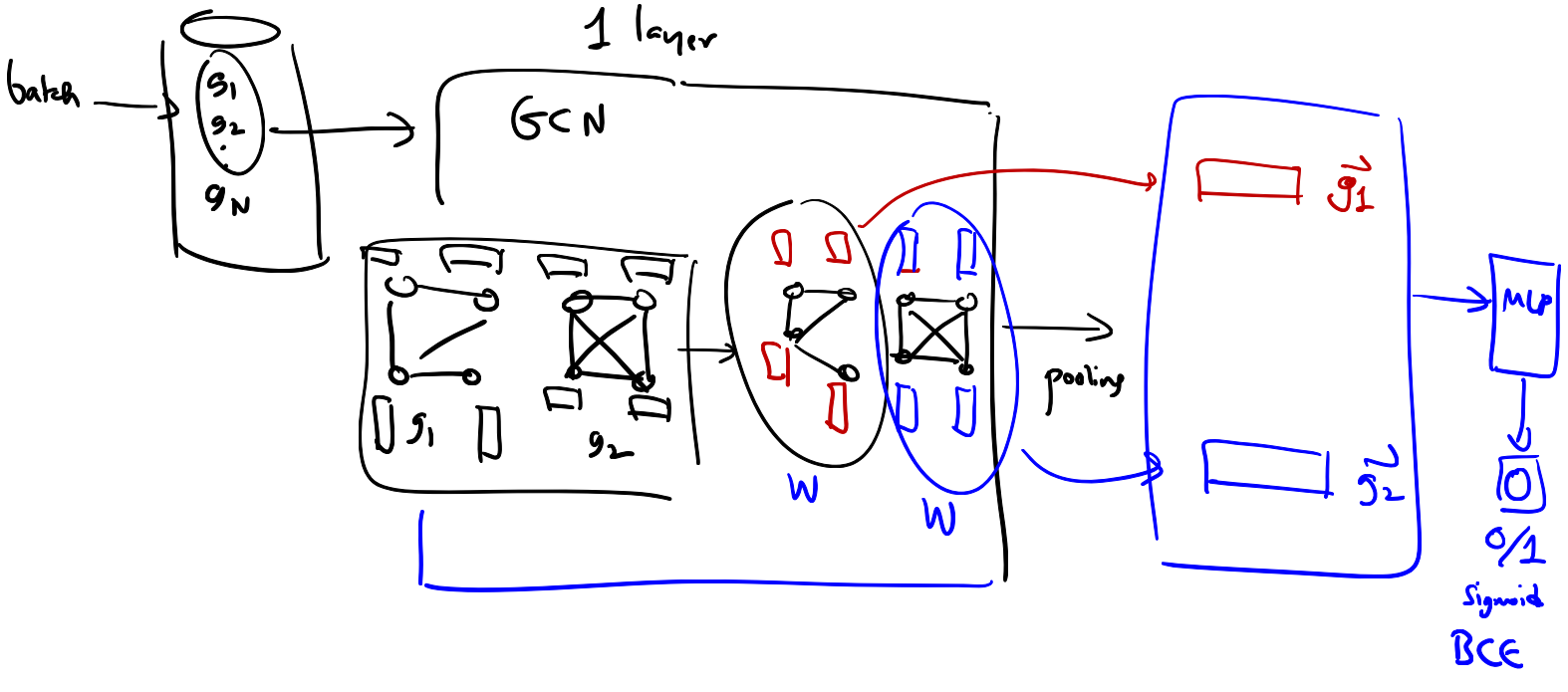
Graph filtering operations

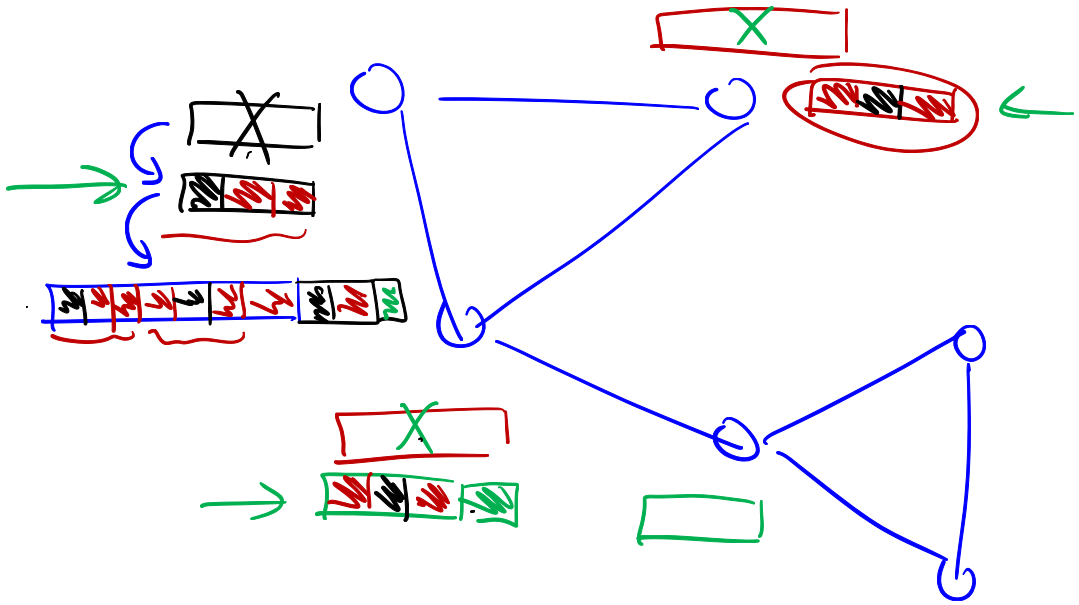
GCN: graph convolutional Network

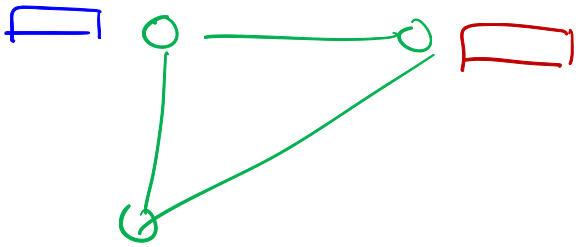


$$i = 1 \dots N \qquad \vec{e}_i' = \sum_{j \in N(i)} \frac{1}{\sqrt{d_i}\sqrt{d_j}} \vec{e}_j \cdot W$$

- $\vec{e}_i'$: embedding for node v_i, $\vec{e}_i' \in \mathbb{R}^{d_{out}}$
- $\vec{e}_j \in \mathbb{R}^{d_{in}}$
- W: $d_{in} \times d_{out}$, learnable (shared across all node)
- $d_{out} = d_{in}$







GCN ⟶ based on laplacian filter

symmetric normalized laplacian

$$L = D - A$$

$$L_n = D^{-1}(D - A) = D^{-1}L$$

$$L_S = D^{-1/2}(D - A)D^{-1/2}$$

$$= \left\{ \frac{l_{ij}}{\sqrt{d_i}\sqrt{d_j}} \right\}$$

$$\vec{e}_i' = \sum_j L_s \cdot e_j$$

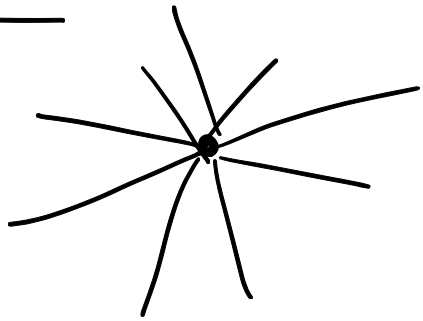
(Kipf et al GCN)

Vasre

$$\vec{e}_i' = \sum_j \frac{1}{\sqrt{d_i}\sqrt{d_j}} e_j \cdot W$$

---

graph SAGE



$N_s(i)$: sample the neighborhood of v_i

$$\vec{e}_i' = \text{agg}\left\{\vec{e}_j, \ \forall j \in N_s(i)\right\}$$

$$\vec{e}_i'' = \sigma\left\{\left(\vec{e}_i \| \vec{e}_j'\right) \cdot W \cdot \right\} \quad j = 1 \dots N_s(i)$$

Concat operation

$|N_s(i)| = r$

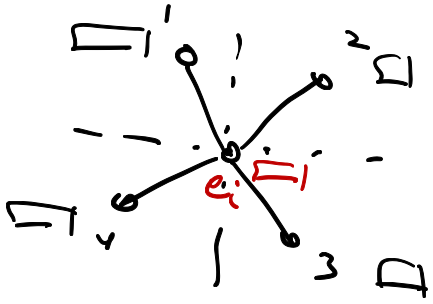
$$\underbrace{e_i \| e_{j_1}' \| e_{j_2}' \| \dots \| e_{j_s}'}$$

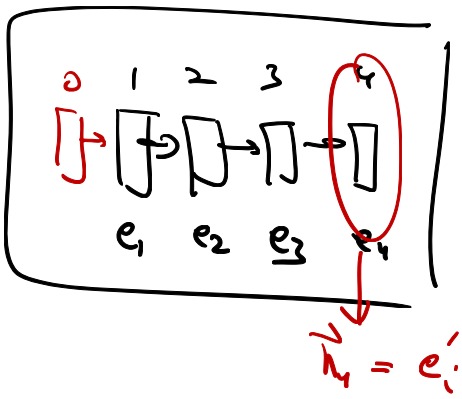
agg:

1) mean aggregate

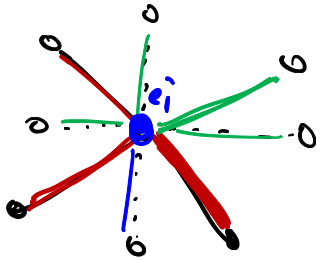
$$\frac{1}{N_s(i)} \sum_{j=1}^{N_s(i)} e_j$$

2) LSTM aggregator





GAT: Graph Attention Filter



$$\underline{\underline{e_i}} = \sum_{j \in N(i)} \alpha_{ij} \vec{e}_j$$

attention based aggregation

orig: d

proj: d_p

$$f_{ij} = \text{relu}\left(\vec{\omega}_a^T \left(\vec{e}_i \cdot W \,\|\, \vec{e}_j \cdot W\right)\right)$$

$d_p \,\|\, d_p$

$2 \cdot d_p$

$2 \cdot d_p$

vector

scalar

W: proj matrix

$d \times d_p$

$\vec{\omega}_a$: attn-vector

$$\alpha_{ij} = \frac{e^{\beta_{ij}}}{\sum_{k=1}^{N(i)} e^{\beta_{ik}}}$$

Softmax across neighbors!

we can do multiple heads

$(\underline{W}, \vec{\omega}_a)^h$ ← head index

$$e_i^{(h)} = \sum_{j=1}^{N(i)} \alpha_{ij}^h \, e_j$$

$e_i^1 \| e_i^2 \| \cdots \| e_i^h$

$h \cdot d_p$ vector

W_0 ←

↓

d-dim