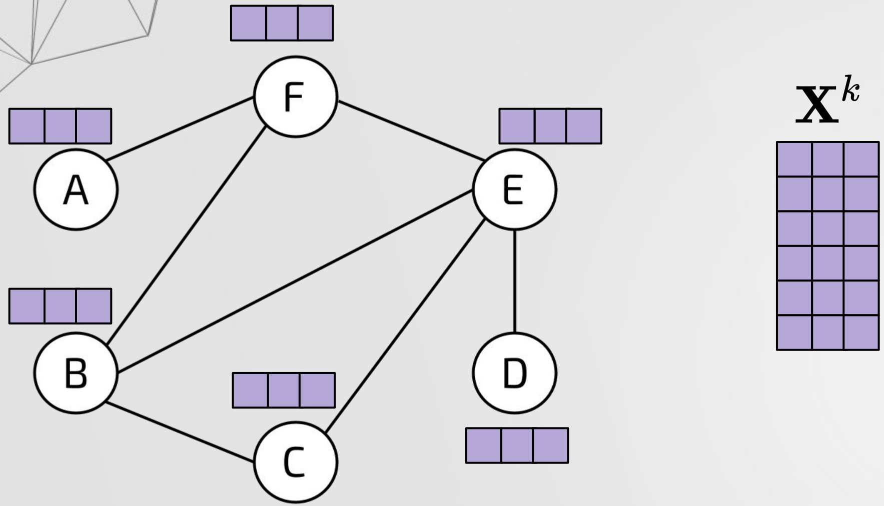
Graph Pooling: DIFFPOOL

Giovanni Pellegrini[1,2,3]

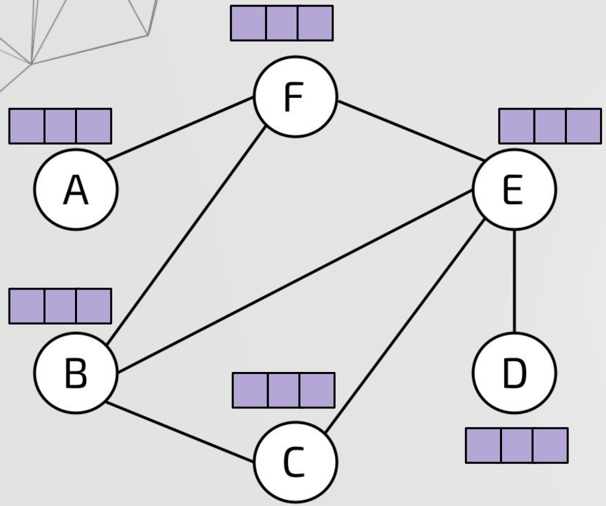
SML[1] Lab, University of Trento, Italy
TIM[2]
EIT DIGITAL[3]

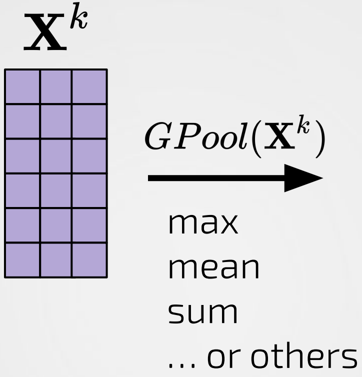
01 Graph prediction



$$\boldsymbol{X}^{t+1} = \mathrm{GConv}(\boldsymbol{A}, \boldsymbol{X}^t, \boldsymbol{W}^t) \quad t = 1, \ldots, k$$

01 Graph prediction

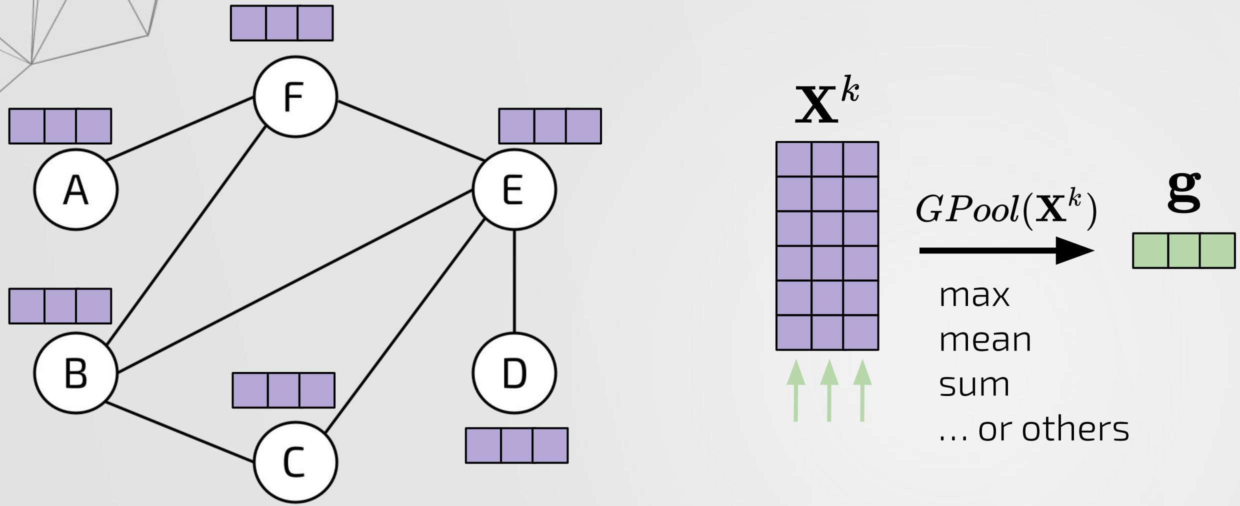




$$\boldsymbol{X}^{t+1} = \mathrm{GConv}(\boldsymbol{A}, \boldsymbol{X}^t, \boldsymbol{W}^t) \quad t = 1, \ldots, k$$

$GPool(\mathbf{X}^k) \longrightarrow$ global pooling function

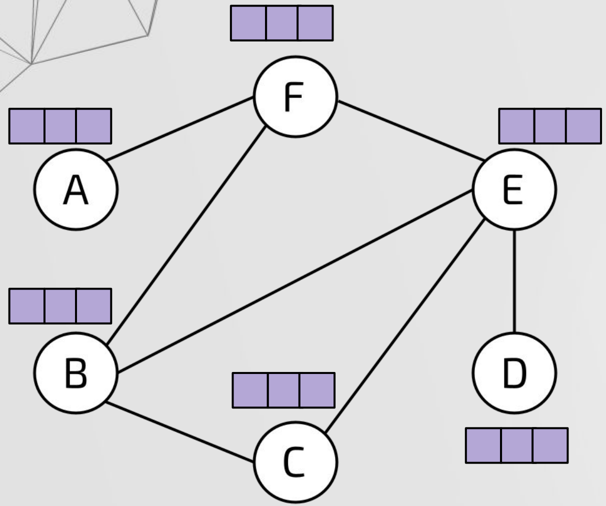
01 Graph prediction

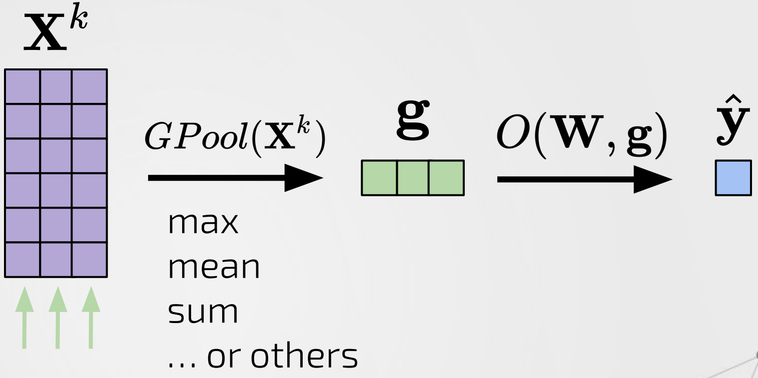


$$\boldsymbol{X}^{t+1} = \mathrm{GConv}(\boldsymbol{A}, \boldsymbol{X}^t, \boldsymbol{W}^t) \quad t = 1, \ldots, k$$

$GPool(\mathbf{X}^k) \longrightarrow$ global pooling function

01 Graph prediction





$$\boldsymbol{X}^{t+1} = \mathrm{GConv}(\boldsymbol{A}, \boldsymbol{X}^{t}, \boldsymbol{W}^{t}) \quad t = 1, \ldots, k$$

$GPool(\mathbf{X}^k) \longrightarrow$ global pooling function

$O(\mathbf{W}, \mathbf{g}) \longrightarrow$ graph readout function

01 Graph prediction

Equivalent to having a virtual “supernode” in the last layer connected to all the nodes in the graph

Pooling the node embeddings all together do not capture hierarchical representations of the network

01 Graph prediction

Equivalent to having a virtual “supernode” in the last layer connected to all the nodes in the graph

Pooling the node embeddings all together do not capture hierarchical representations of the network

DIFFPOOL: hierarchical nodes pooling strategy

02 DIFFPOOL[1]

DIFFerentiable **POOL**ing: Compute an hierarchical representation of the graph by aggregating "close" nodes

Ying et al, *Hierarchical Graph Representation Learning with Differentiable Pooling,* NIPS '18

02 DIFFPOOL[1]

DIFFerentiable **POOL**ing: Compute an hierarchical representation of the graph by aggregating “close” nodes

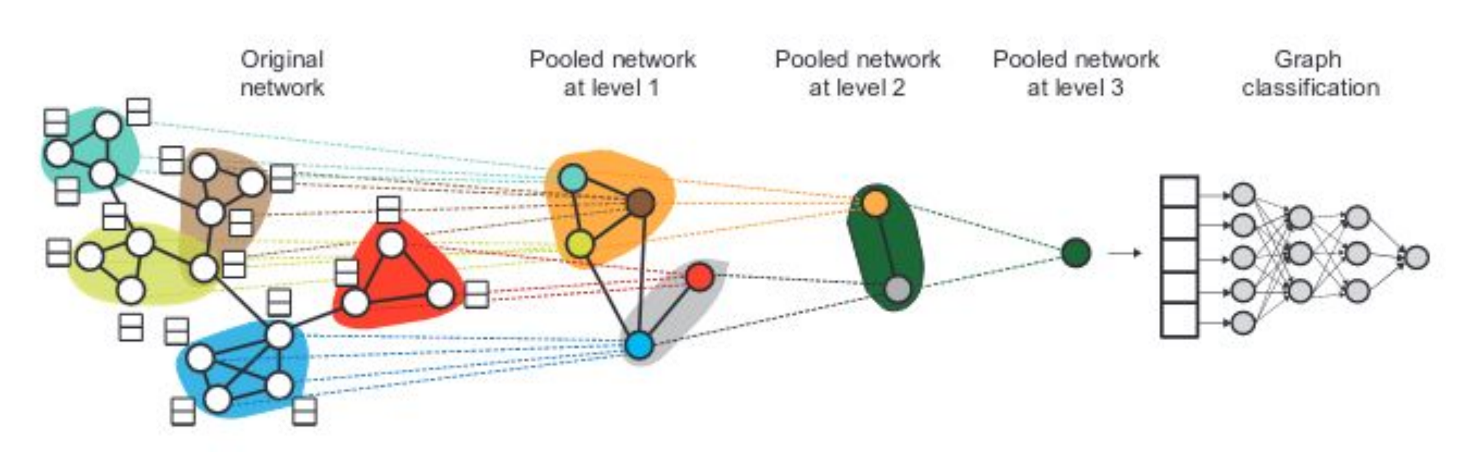
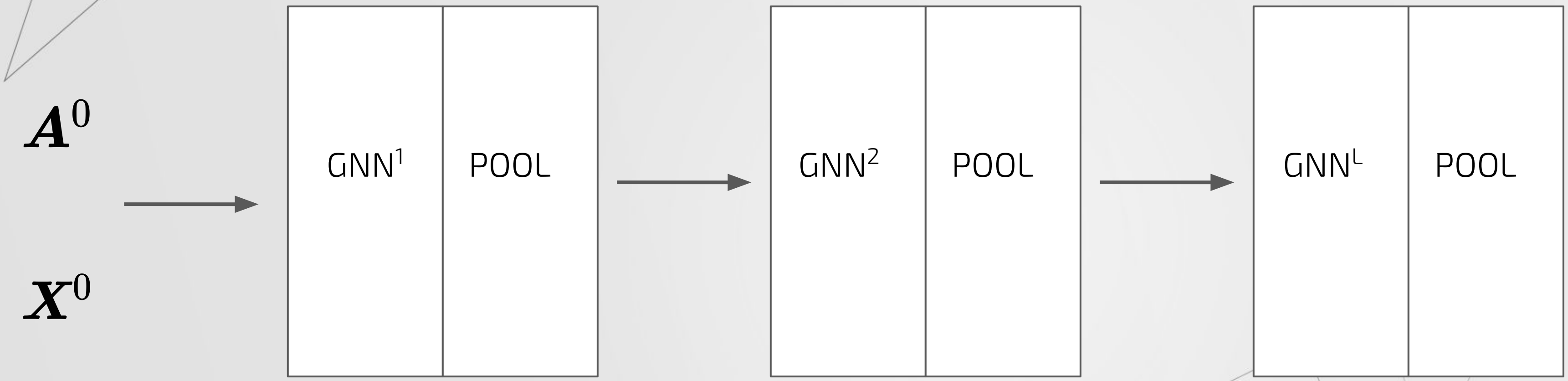


Image taken from the original publication.

Ying et al, *Hierarchical Graph Representation Learning with Differentiable Pooling,* NIPS ‘18

02 DIFFPOOL

Idea: stack several GNN and pooling layer on top of each other



02 DIFFPOOL

$$G = (\boldsymbol{A}, \boldsymbol{X})$$

adjacency matrix

node features matrix

02 DIFFPOOL

$$G = (\boldsymbol{A}, \boldsymbol{X})$$

adjacency matrix

node features matrix

$$\boldsymbol{D} = \{(\boldsymbol{G}_1, y_1), \ldots, (\boldsymbol{G}_n, y_n)\}$$

02 DIFFPOOL

$$G = (\boldsymbol{A}, \boldsymbol{X})$$

adjacency matrix

node features matrix

$$\boldsymbol{D} = \{(\boldsymbol{G}_1, y_1), \ldots, (\boldsymbol{G}_n, y_n)\}$$

$$\boldsymbol{X}^{t+1} = \mathrm{GConv}(\boldsymbol{A}, \boldsymbol{X}^t, \boldsymbol{W}^t)$$

$$\boldsymbol{Z} = \mathrm{GNN}(\boldsymbol{A}, \boldsymbol{X})$$

02 DIFFPOOL

$$G = (\boldsymbol{A}, \boldsymbol{X})$$

adjacency matrix

node features matrix

$$\boldsymbol{D} = \{(\boldsymbol{G}_1, y_1), \ldots, (\boldsymbol{G}_n, y_n)\}$$

$$\boldsymbol{X}^{t+1} = \mathrm{GConv}(\boldsymbol{A}, \boldsymbol{X}^t, \boldsymbol{W}^t)$$

$$\boldsymbol{Z} = \mathrm{GNN}(\boldsymbol{A}, \boldsymbol{X})$$

arbitrary GNN that computes K iterations, $Z = X^K$

02 DIFFPOOL

Given $\boldsymbol{A}$ and $\boldsymbol{Z}$, find a coarse representation of the graph

02 DIFFPOOL

Given $\boldsymbol{A}$ and $\boldsymbol{Z}$, find a coarse representation of the graph

$$\boldsymbol{A} \in R^{n \times n}$$

$$\boldsymbol{Z} = R^{n \times d}$$

02 DIFFPOOL

Given $\boldsymbol{A}$ and $\boldsymbol{Z}$, find a coarse representation of the graph

$$\boldsymbol{A} \in R^{n\times n} \qquad \boldsymbol{A'} \in R^{m\times m}$$

$$\longrightarrow \qquad \text{with} \quad m < n$$

$$\boldsymbol{Z} = R^{n\times d} \qquad \boldsymbol{Z'} = R^{m\times d}$$

02 DIFFPOOL

Given $\boldsymbol{A}$ and $\boldsymbol{Z}$, find a coarse representation of the graph

$$\boldsymbol{A} \in R^{n\times n} \longrightarrow \boldsymbol{A'} \in R^{m\times m}$$

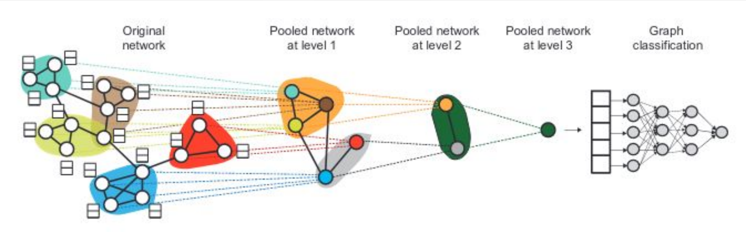
with $m < n$

$$\boldsymbol{Z} = R^{n\times d} \qquad \boldsymbol{Z'} = R^{m\times d}$$

Solution -> learn a cluster assignment for each node

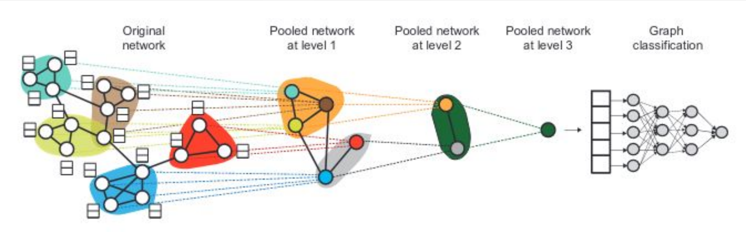
03 Pooling in DIFFPOOL

The final representation is obtained by coarsening the graph in L hierarchical steps. To do that, at each step a **cluster assignment matrix** is learned.



03 Pooling in DIFFPOOL

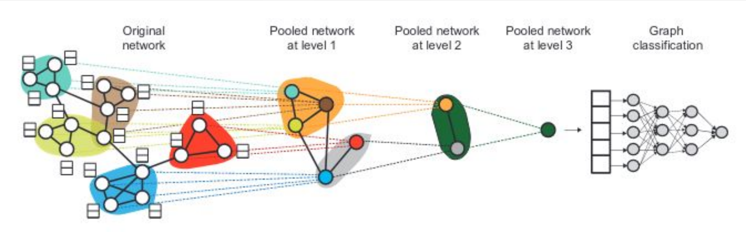
The final representation is obtained by coarsening the graph in L hierarchical steps. To do that, at each step a **cluster assignment matrix** Is learned.



$$\boldsymbol{S}^l \in \boldsymbol{R}^{n^l \times n^{l+1}}$$

03 Pooling in DIFFPOOL

The final representation is obtained by coarsening the graph in L hierarchical steps. To do that, at each step a **cluster assignment matrix** Is learned.



$$\boldsymbol{S}^l \in \boldsymbol{R}^{n^l \times n^{l+1}}$$

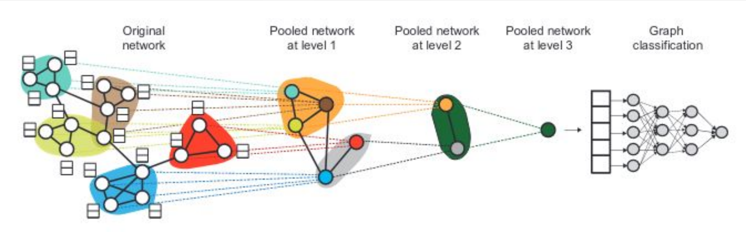
$\boldsymbol{n}^{l+1}$ = number of cluster at the next step

$\boldsymbol{n}^{l}$ = number of clusters at the current step

$\boldsymbol{n}^{0}$ = number of nodes in the graph

03 Pooling in DIFFPOOL

The final representation is obtained by coarsening the graph in L hierarchical steps. To do that, at each step a **cluster assignment matrix** Is learned.



$$\boldsymbol{S}^{l} \in R^{n^{l} \times n^{l+1}}$$

n^{l+1} = number of cluster at the next step

n^{l} = number of clusters at the current step

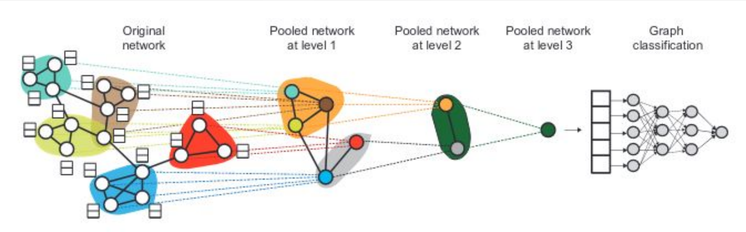
n^{0} = number of nodes in the graph

transposed

$$\boldsymbol{X}^{l+1} = \boldsymbol{S}^{l^{T}} \boldsymbol{Z}^{l} \longrightarrow R^{n^{l+1} \times d}$$

03 Pooling in DIFFPOOL

The final representation is obtained by coarsening the graph in L hierarchical steps. To do that, at each step a **cluster assignment matrix** Is learned.



$$\boldsymbol{S}^l \in R^{n^l \times n^{l+1}}$$

n^{l+1} = number of cluster at the next step

n^l = number of clusters at the current step

n^0 = number of nodes in the graph

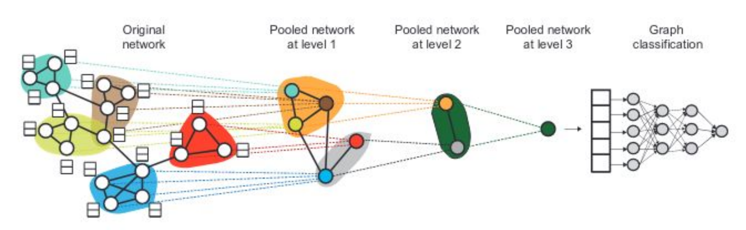
transposed

$$\boldsymbol{X}^{l+1} = \boldsymbol{S}^{l^T} \boldsymbol{Z}^l \longrightarrow R^{n^{l+1} \times d}$$

$$\boldsymbol{A}^{l+1} = \boldsymbol{S}^{l^T} \boldsymbol{A} \boldsymbol{S}^l \longrightarrow R^{n^{l+1} \times n^{l+1}}$$

03 Pooling in DIFFPOOL

The final representation is obtained by coarsening the graph in L hierarchical steps. To do that, at each step a **cluster assignment matrix** Is learned.



$$\boldsymbol{S}^l \in R^{n^l \times n^{l+1}}$$

n^{l+1} = number of cluster at the next step

n^l = number of clusters at the current step

n^0 = number of nodes in the graph

The number of clusters at each step is an hyperparameter!

transposed

$$\boldsymbol{X}^{l+1} = \boldsymbol{S}^{l^T} \boldsymbol{Z}^l \longrightarrow R^{n^{l+1} \times d}$$

$$\boldsymbol{A}^{l+1} = \boldsymbol{S}^{l^T} \boldsymbol{A} \boldsymbol{S}^l \longrightarrow R^{n^{l+1} \times n^{l+1}}$$

04 Learning the assignment matrix

How is the cluster matrix learned?

04 Learning the assignment matrix

How is the cluster matrix learned?

$$\boldsymbol{Z}^{l} = \mathrm{GNN}^{l}_{\mathrm{emb}}(\boldsymbol{A}^{l}, \boldsymbol{X}^{l})$$

04 Learning the assignment matrix

How is the cluster matrix learned?

$$\boldsymbol{Z}^{l} = \mathrm{GNN}^{l}_{\mathrm{emb}}(\boldsymbol{A}^{l}, \boldsymbol{X}^{l})$$

$$\boldsymbol{S}^{l} = \mathrm{softmax}(\mathrm{GNN}^{l}_{\mathrm{pool}}(\boldsymbol{A}^{l}, \boldsymbol{X}^{l}))$$

04 Learning the assignment matrix

How is the cluster matrix learned?

$$\boldsymbol{Z}^l = \mathrm{GNN}^l_{\mathrm{emb}}(\boldsymbol{A}^l, \boldsymbol{X}^l)$$

$$\boldsymbol{S}^l = \mathrm{softmax}(\mathrm{GNN}^l_{\mathrm{pool}}(\boldsymbol{A}^l, \boldsymbol{X}^l))$$

$$\mathrm{GNN}^l_{\mathrm{pool}} \xrightarrow{\text{outputs}} \boldsymbol{R}^{n^l \times n^{l+1}}$$

Softmax is applied row-wise, it assigns probabilities to which clusters to belong to in the next step.

04 Prediction

How is the cluster matrix learned?

$$\boldsymbol{Z}^l = \mathrm{GNN}_{\mathrm{emb}}^l(\boldsymbol{A}^l, \boldsymbol{X}^l)$$

$$\boldsymbol{S}^l = \boxed{\mathrm{softmax}}(\mathrm{GNN}_{\mathrm{pool}}^l(\boldsymbol{A}^l, \boldsymbol{X}^l))$$

differentiable!

$$\mathrm{GNN}_{\mathrm{pool}}^l \xrightarrow{\text{outputs}} \boldsymbol{R}^{n^l \times n^{l+1}}$$

Softmax is applied row-wise, it assigns probabilities to which clusters to belong to in the next step.

05 Final prediction

At the last step the graph is condensed in one single node (vector). The final prediction is the output of a MLP.

$$\hat{y} = \mathrm{MLP}(\boldsymbol{Z}^L)$$

$$E = L(y, \hat{y})$$

05 Final prediction

At the last step the graph is condensed in one single node (vector). The final prediction is the output of a MLP.

$$\hat{y} = \mathrm{MLP}(\boldsymbol{Z}^L)$$

$$E = L(y, \hat{y})$$

Let's switch to the notebook...