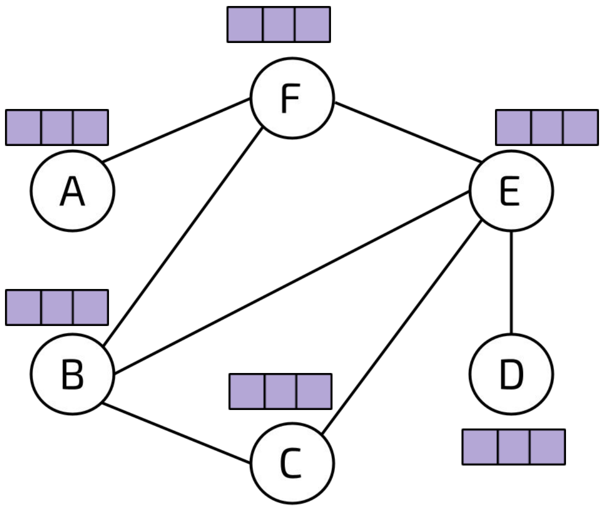
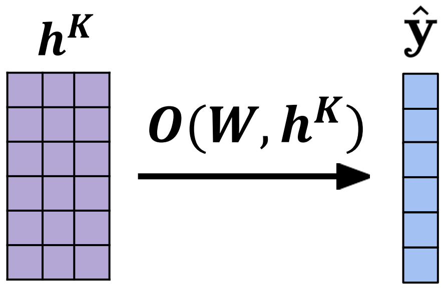
Graph Classification

Predizione dell'etichetta di un nodo

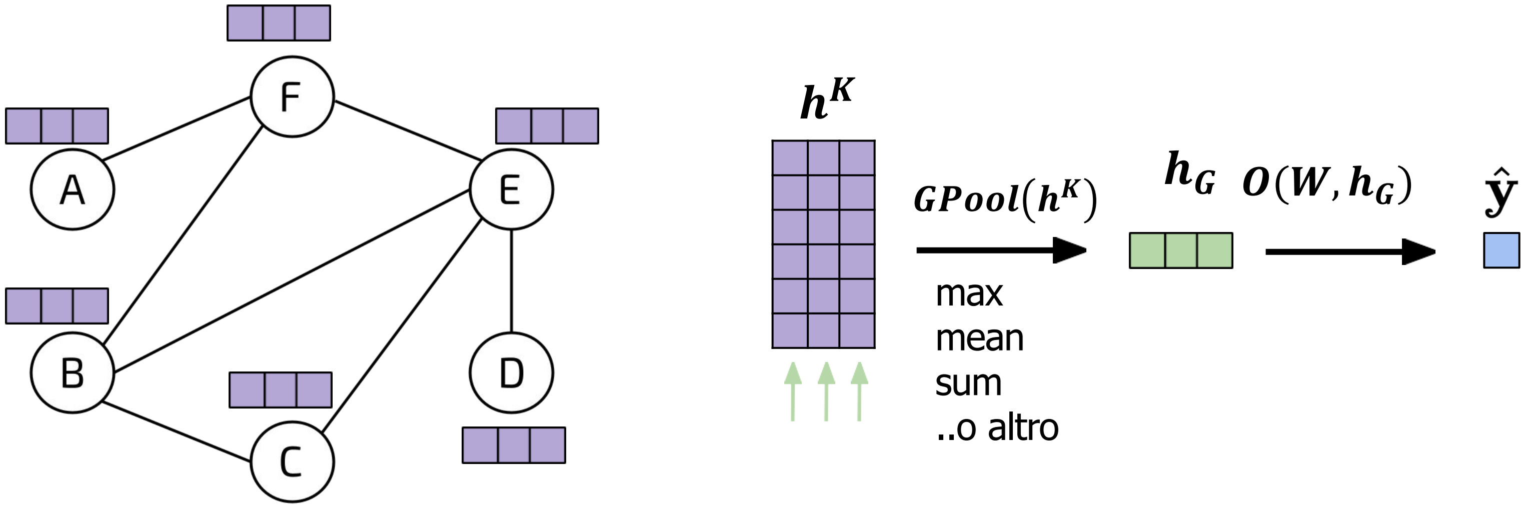




$$h^{k+1} = GNN\left(W^k, h^k, A\right)\ k = 0, \dots, K$$

$O(W, h^K)$ → Funzione node readout

Classificazione grafi



$$h^{k+1} = GNN(W^k, h^k, A)\ k = 0, \dots, K$$

$h_G = GPool(h^K)$ $\longrightarrow$ Funzione di pooling globale

$O(W, h_G)$ $\longrightarrow$ Funzione di graph readout

Graph Pooling?

- Processo di aggregazione delle informazioni a livello di nodi per ottenere una rappresentazione a livello di grafo
- **Obiettivo**: Ridurre la dimensionalità da node-level a graph-level
- **Necessità**: I grafi hanno dimensioni variabili, ma i classificatori richiedono input di dimensione fissa

$$\boldsymbol{h_G} = \boldsymbol{GPool}\left(\{\boldsymbol{h_v^K} \mid \boldsymbol{v} \in \boldsymbol{V}\}\right)$$

Vari tipi

- Sum Pooling

$$\boldsymbol{h_G} = \sum_{\boldsymbol{v \in V}} \boldsymbol{h_v^K}$$

- Mean Pooling:

$$\boldsymbol{h_G} = \frac{\boldsymbol{1}}{|\boldsymbol{V}|} \sum_{\boldsymbol{v \in V}} \boldsymbol{h_v^K}$$

- Max Pooling

$$\boldsymbol{h_G} = \boldsymbol{max_{v \in V} h_v^K}$$

- Attention Pooling

$$\boldsymbol{h_G} = \sum_{\boldsymbol{v} \in \boldsymbol{V}} \boldsymbol{\alpha_v h_v^K}$$

Come calcoliamo il mecccanismo di attenzione (semplificaizone)

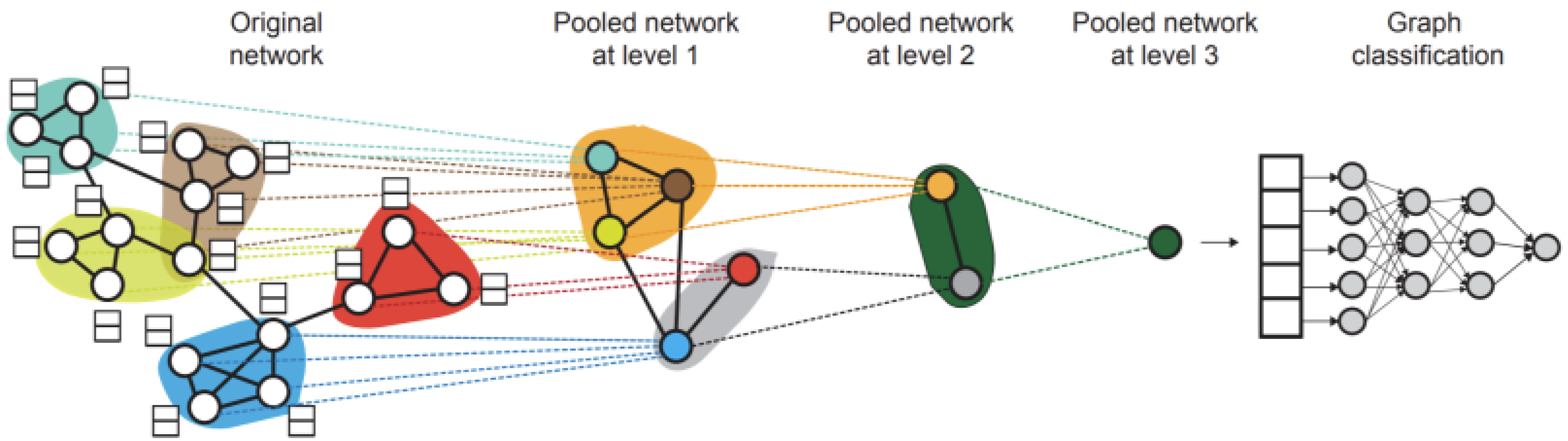
$$\boldsymbol{e_v} = \boldsymbol{W} \times \boldsymbol{h_v^K} + \boldsymbol{B}$$

$$\boldsymbol{\alpha}_{\mathbf{v}} = \frac{\mathbf{exp}(\mathbf{e}_{\mathbf{v}})}{\sum \boldsymbol{exp}(\boldsymbol{e_u})}$$

Varianti

- Self-attention
- Multi-head attention
- GAT
- Gerarchico

Pooling gerarchico



Approccio multi-livello: Riduce gradualmente la dimensione del grafo

Preserva struttura: Mantiene le connessioni importanti

Selezione intelligente: Sceglie i nodi più rilevanti

Coarsening: Processo iterativo di semplificazione

SAG Pooling

pool.SAGPooling

class SAGPooling(in_channels: int, ratio: ~typing.Union[float, int] = 0.5, GNN: ~torch.nn.modules.module.Module = <class 'torch_geometric.nn.conv.graph_conv.GraphConv'>, min_score: ~typing.Optional[float] = None, multiplier: float = 1.0, nonlinearity: ~typing.Union[str, ~typing.Callable] = 'tanh', **kwargs) [source]

Bases: `Module`

The self-attention pooling operator from the "Self-Attention Graph Pooling" and "Understanding Attention and Generalization in Graph Neural Networks" papers.

If `min_score` $\tilde{\alpha}$ is `None`, computes:

$$\begin{aligned}\mathbf{y} &= \mathrm{GNN}(\mathbf{X}, \mathbf{A}) \\ \mathbf{i} &= \mathrm{top}_k(\mathbf{y}) \\ \mathbf{X}' &= (\mathbf{X} \odot \tanh(\mathbf{y}))_{\mathbf{i}} \\ \mathbf{A}' &= \mathbf{A}_{\mathbf{i},\mathbf{i}}\end{aligned}$$

If `min_score` $\tilde{\alpha}$ is a value in `[0, 1]`, computes:

$$\begin{aligned}\mathbf{y} &= \mathrm{softmax}(\mathrm{GNN}(\mathbf{X}, \mathbf{A})) \\ \mathbf{i} &= \mathbf{y}_i > \tilde{\alpha} \\ \mathbf{X}' &= (\mathbf{X} \odot \mathbf{y})_{\mathbf{i}} \\ \mathbf{A}' &= \mathbf{A}_{\mathbf{i},\mathbf{i}}.\end{aligned}$$

Projections scores are learned based on a graph neural network layer.

PARAMETERS:

- **in_channels** (*int*) – Size of each input sample.
- **ratio** (*float or int*) – Graph pooling ratio, which is used to compute $k = \lceil \mathrm{ratio} \cdot N \rceil$, or the value of k itself, depending on whether the type of `ratio` is `float` or `int`. This value is

TopK Pooling

pool.TopKPooling

class **TopKPooling** (**in_channels**: int, **ratio**: Union[int, float] = **0.5**, **min_score**: Optional[float] = **None**, **multiplier**: float = **1.0**, **nonlinearity**: Union[str, Callable] = **'tanh'**) [source]

Bases: `Module`

top_k pooling operator from the "Graph U-Nets", "Towards Sparse Hierarchical Graph Classifiers" and "Understanding Attention and Generalization in Graph Neural Networks" papers.

If `min_score` $\tilde{\alpha}$ is `None`, computes:

$$\mathbf{y} = \sigma\left(\frac{\mathbf{X}\mathbf{p}}{\|\mathbf{p}\|}\right)$$
$$\mathbf{i} = \mathrm{top}_k(\mathbf{y})$$
$$\mathbf{X}' = (\mathbf{X} \odot \mathrm{tanh}(\mathbf{y}))_{\mathbf{i}}$$
$$\mathbf{A}' = \mathbf{A}_{\mathbf{i},\mathbf{i}}$$

If `min_score` $\tilde{\alpha}$ is a value in `[0, 1]`, computes:

$$\mathbf{y} = \mathrm{softmax}(\mathbf{X}\mathbf{p})$$
$$\mathbf{i} = \mathbf{y}_i > \tilde{\alpha}$$
$$\mathbf{X}' = (\mathbf{X} \odot \mathbf{y})_{\mathbf{i}}$$
$$\mathbf{A}' = \mathbf{A}_{\mathbf{i},\mathbf{i}},$$

where nodes are dropped based on a learnable projection score $\mathbf{p}$.

PARAMETERS:

- **in_channels** (*int*) – Size of each input sample.
- **ratio** (*float or int*) – The graph pooling ratio, which is used to compute $k = \lceil \mathrm{ratio} \cdot N \rceil$, or the value of k itself, depending on whether the type of `ratio` is `float` or `int`. This value is ignored if `min_score` is not `None`. (default: `0.5`)