Deep Learning for Graphs - Fundamentals

INTELLIGENT SYSTEMS FOR PATTERN RECOGNITION (ISPR)

DAVIDE BACCIU – DIPARTIMENTO DI INFORMATICA - UNIVERSITA' DI PISA

DAVIDE.BACCIU@UNIPI.IT

Lecture Outline

- Motivations
- Formalization of the learning task: graph prediction, induction, transduction and generation
- Historical perspective: contractive and contextual models
- A view on modern deep learning for graphs
  - Convolutional, feedforward, recurrent and attention-based approaches

Introduction

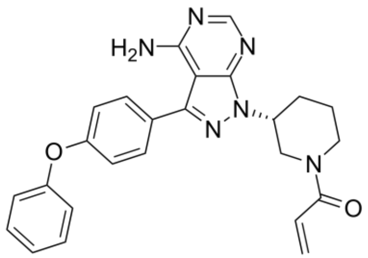
Why Graphs?

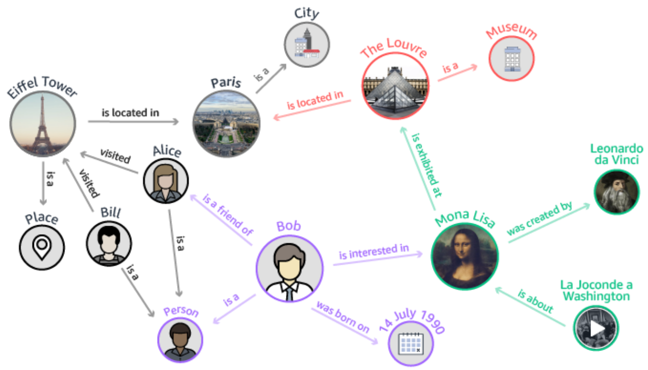
Why Graphs?

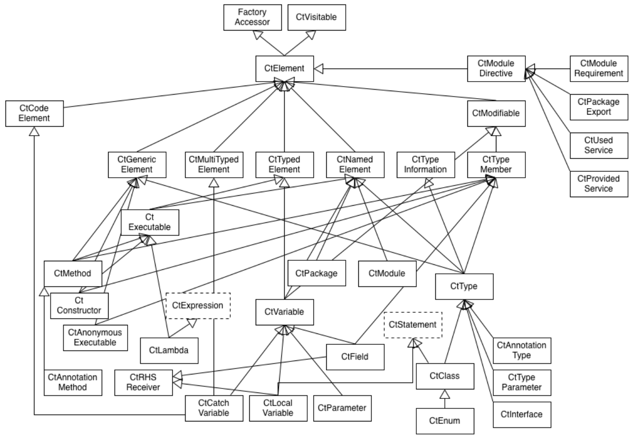
Context is fundamental for the correct interpretation of information

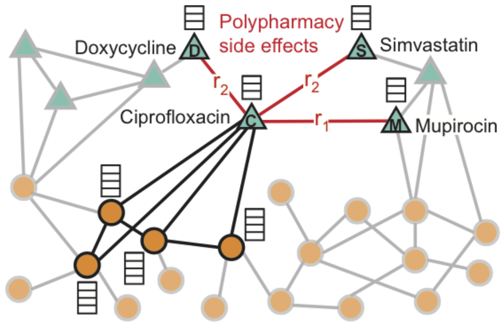
…well also for the plenty of applications

- Chemistry & Physics
- Knowledge graphs
- (Bio/social) networks
- Recommender systems
- Point clouds
- Code & ICT systems







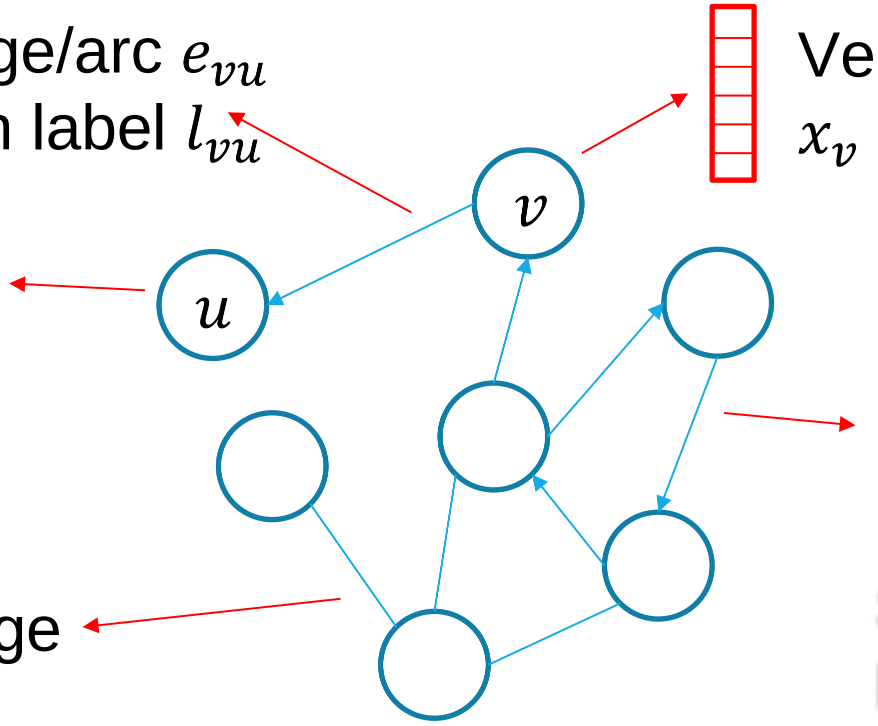


Graph Structured Data

Oriented edge/arc e_{vu}
possibly with label l_{vu}

Vectorial node label
x_v

Node/vertex u



Cycle

Undirected edge

Structures are useful because allow to represent relationships in the data

UNIVERSITÀ DI PISA

A Nomenclature Nightmare

Neural networks for graphs

Deep learning for graphs

CNN for/on graphs

Graph neural networks

Deep Graph Networks

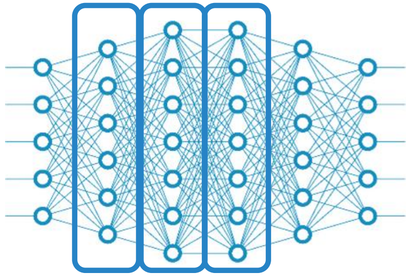
Graph CNN

Learning graph/node embedding

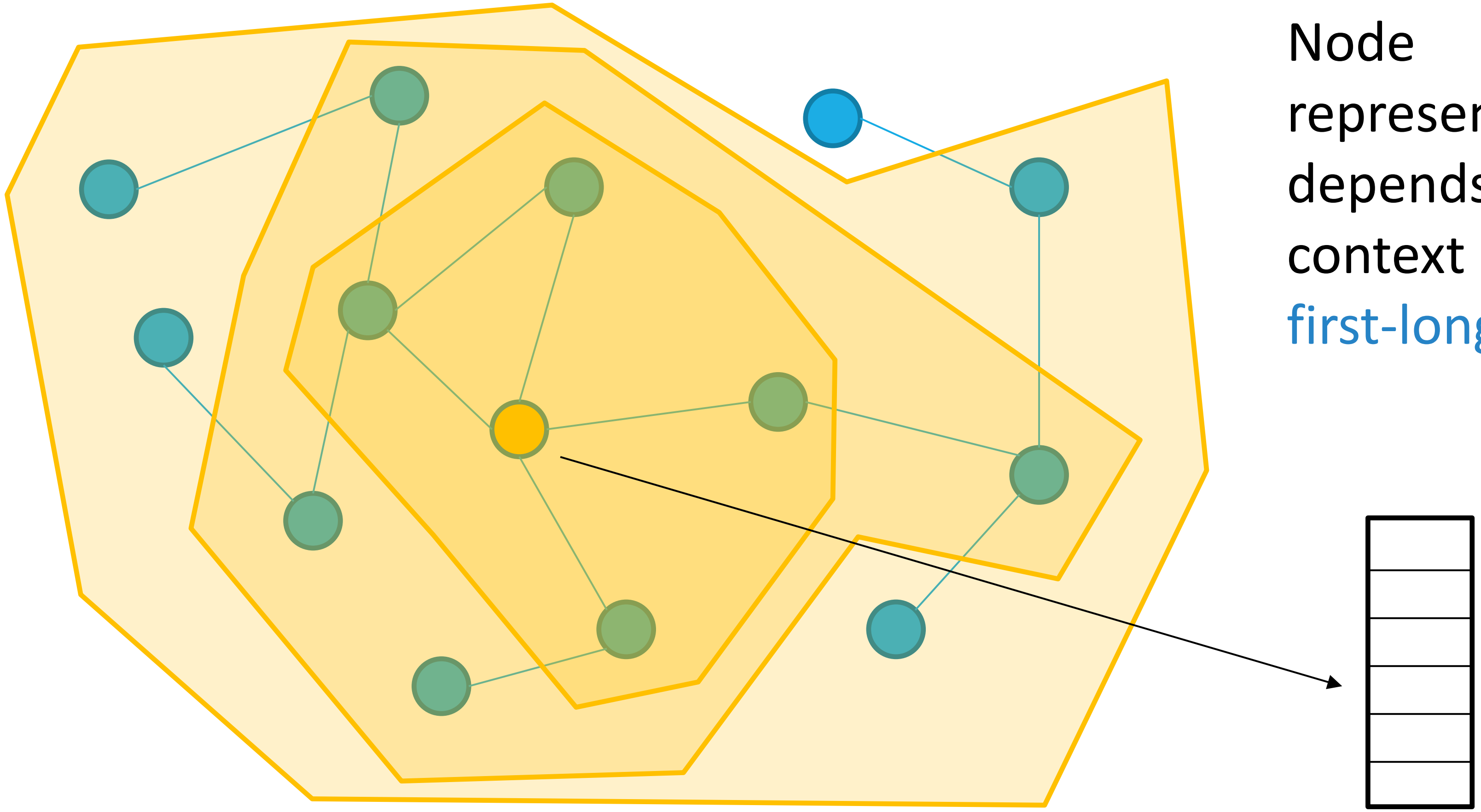
Geometric deep learning

Graph Convolutional Networks

Deep Learning with graphs

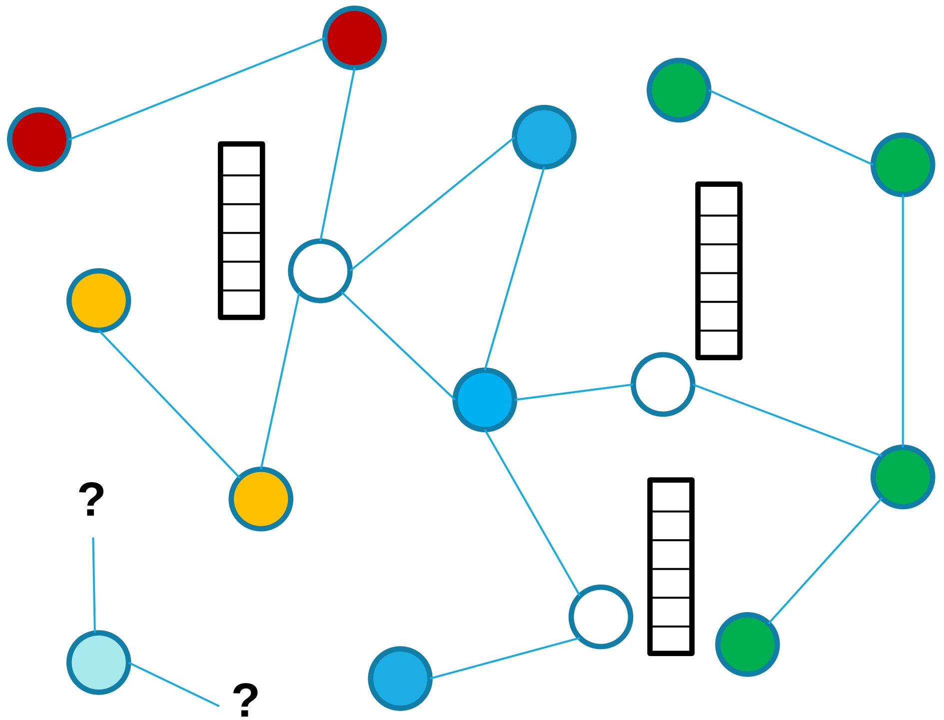


Hierarchical representation learning allows to efficiently diffuse information through graph structure



Node representation depends on its context (shorter first-longer later)

Predictive Tasks – Network data

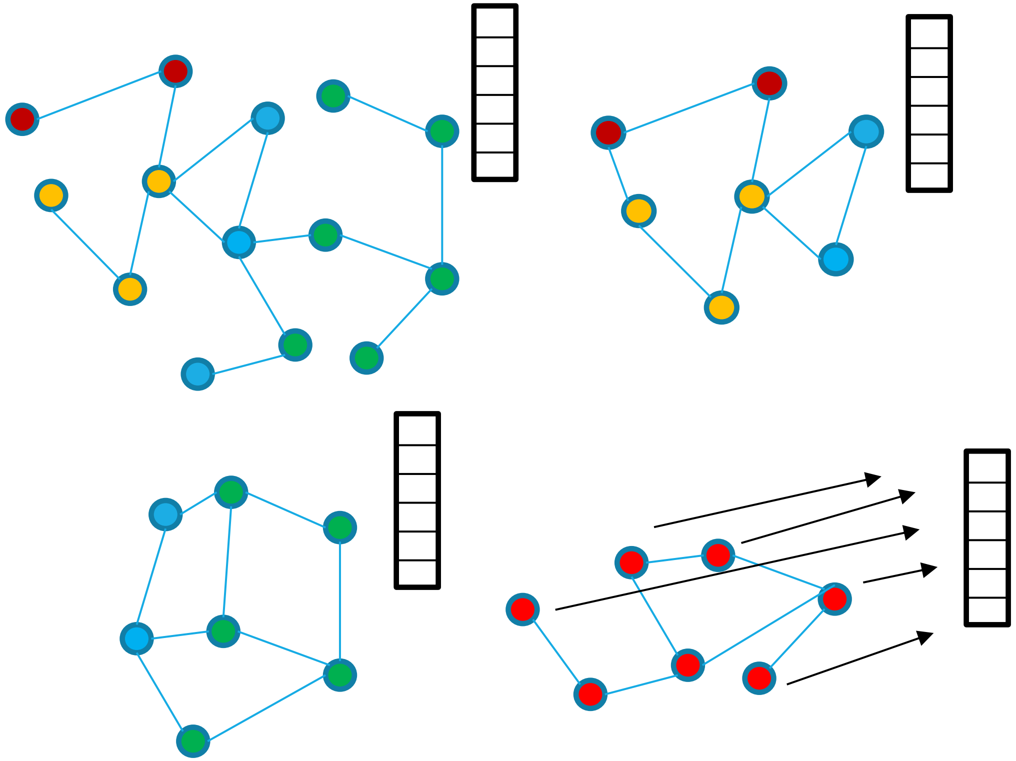


Node predictions
Predict a type or a continuous value for a given node

Link prediction
Predict whether two nodes are linked

Community/module detection
Identify clusters of linked nodes that are alike

Predictive Tasks – Graph Level



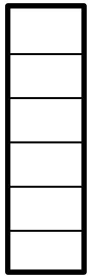
A dataset of i.i.d graphs

Graph classification
Assign whole structure to a specific class

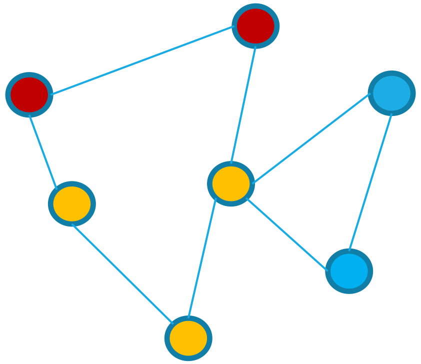
Graph regression
Regress a structure to a value (or a vector of values)

Transductive tasks

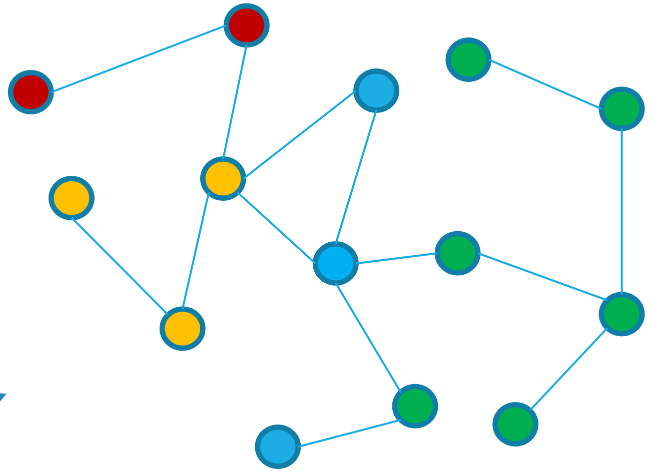
Given a vectorial and/or structured input



$\boldsymbol{x}$



$\boldsymbol{y}$



$\boldsymbol{y} \sim P(\boldsymbol{y}|\boldsymbol{x})$

Learn to generate a structured prediction

An Hystorical (and Geographical) Perspective

Early neural network approaches to deal with cyclic graphs of varying topology date back to 2005-2009

(Sperduti & Starita, TNN 1997)

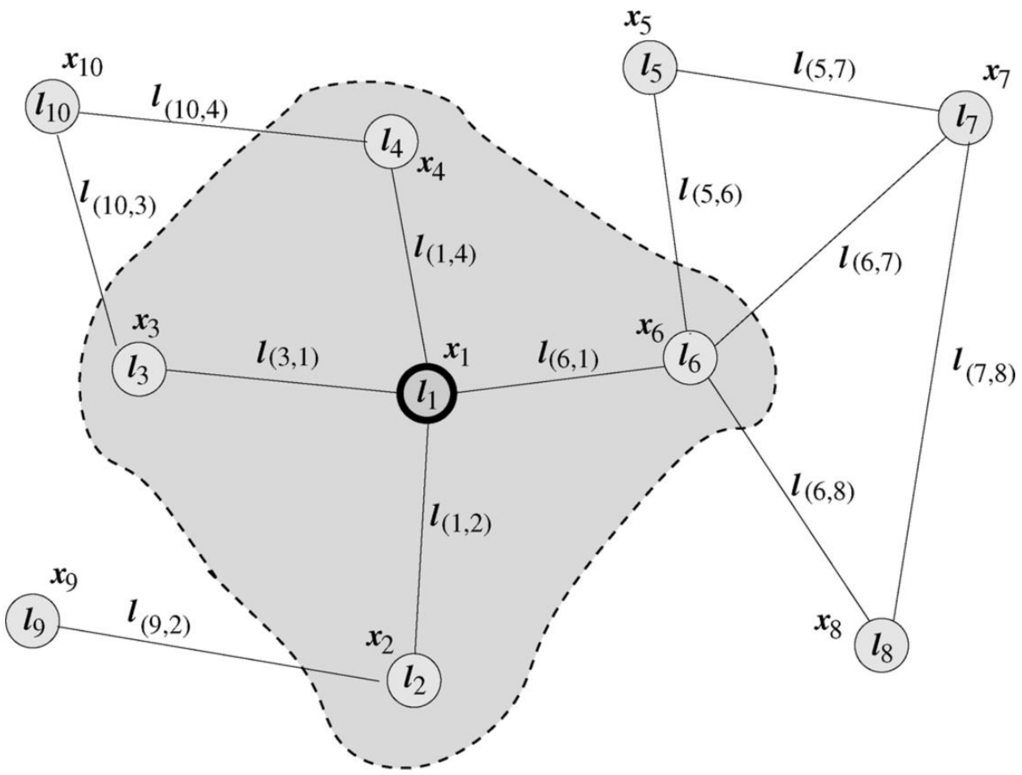
A. Micheli, TNN 2009

UNIVERSITÀ DI PISA

Scarselli et al, TNN 2009

UNIVERSITÀ DI SIENA 1240

Contractive - Graph Neural Networks (GNN)



$$x_1 = f_w(l_1, \underbrace{l_{(1,2)}, l_{(3,1)}, l_{(1,4)}, l_{(6,1)}}_{l_{co[1]}}, \underbrace{x_2, x_3, x_4, x_6}_{x_{ne[1]}}, \underbrace{l_2, l_3, l_4, l_6}_{l_{ne[n]}})$$

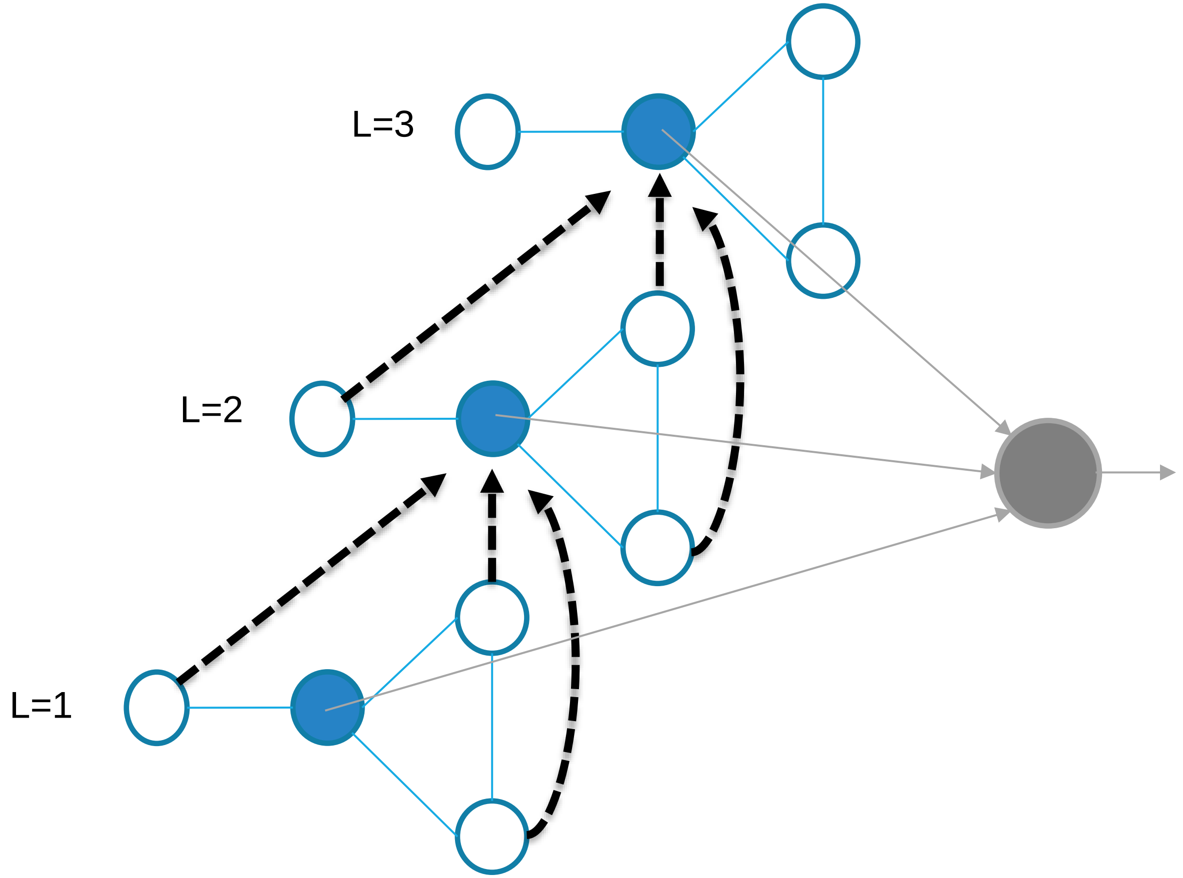
- Extend the Recurrent/Recursive Neural Network approach to cyclic graphs
- Handle loops through fixed points
- Impose dynamic weight constraints to yield a contractive state mapping

Scarselli et al, TNN 2009

https://sailab.diism.unisi.it/gnn/

UNIVERSITÀ DI PISA

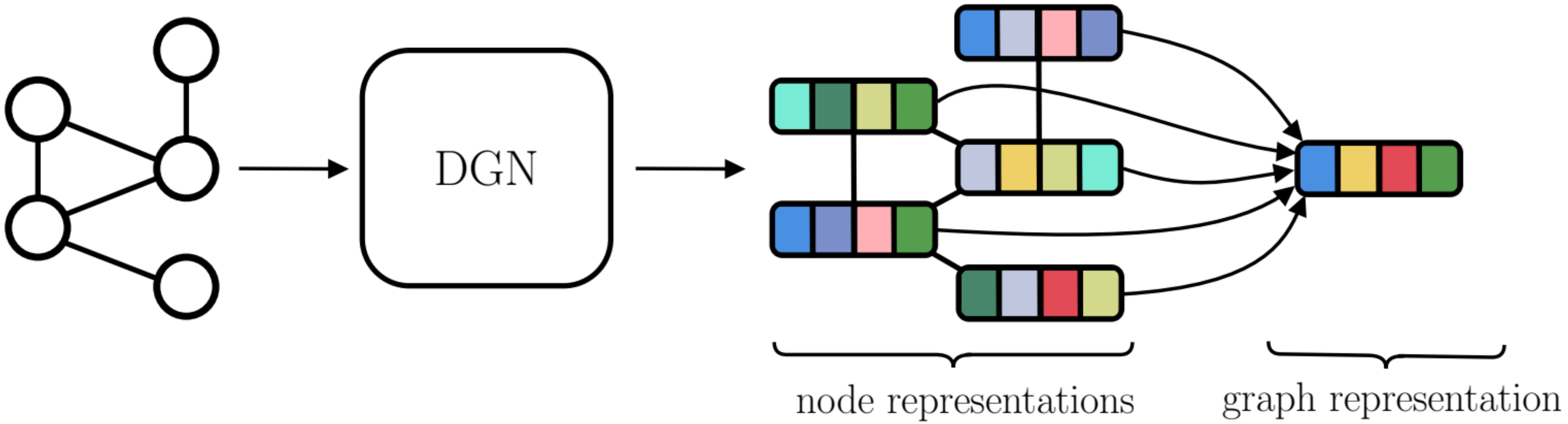
Contextual - Neural Networks for Graphs (NN4G)



- A feedforward approach to process graphs
- Handle loops through layering
- Uses context from frozen earlier layers compute the state on the node at current layer
- Layerwise training

A. Micheli, TNN 2009

Deep Graph Networks



- Encode vertices and the graph itself into a vector space by means of an adaptive (learnable) mapping
- Use the learned encodings to solve predictive and explorative tasks

A Survey of Recent Approaches

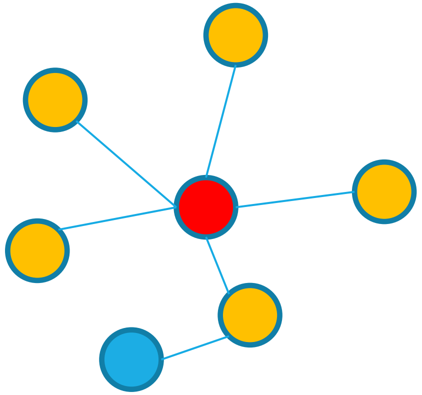
- Convolutional Neural Networks for Graphs
  - Spectral
  - Spatial
- Message Passing Graph Processing
  - The message passing paradigm
  - Overview of relevant feedforward approaches
  - Graph reduction
- Recurrent (randomized) graph processing
- Attention-based graph processing (Graph Transformers)

Convolutional Neural Networks for Graphs

How to Perform Convolutions on Graphs?

SPATIAL DOMAIN

What is the equivalent of sliding a kernel to aggregate local spatial information?



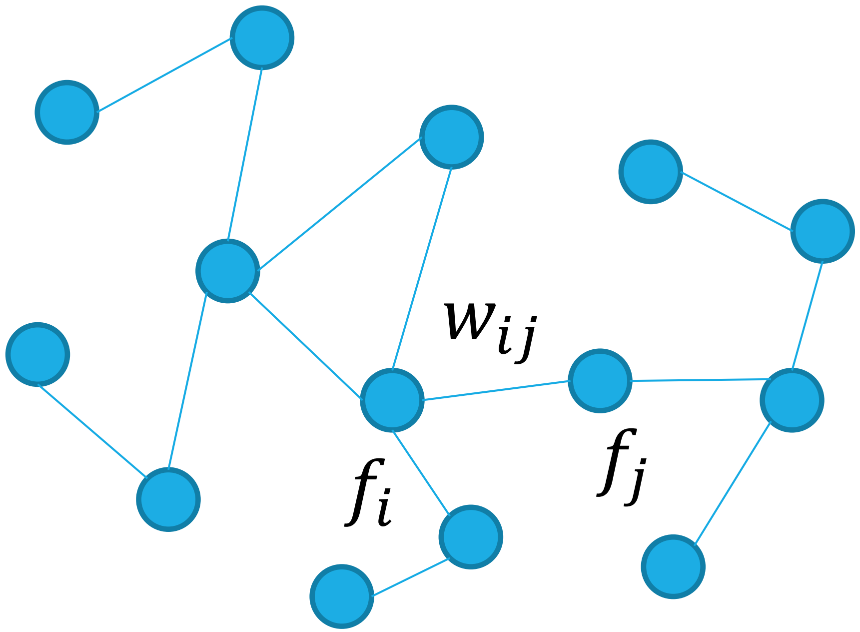
SPECTRAL DOMAIN

$$\mathcal{F}(f * g) = \mathcal{F}(f) \times \mathcal{F}(g)$$

Exploit the Convolution Theorem and Fourier analysis to perform convolutions in the spectral domain

Decompose a function f as a combination of vectors $\boldsymbol{e}_k$ from an orthonormal basis

The Spectral Scenario



- Single weighted undirected graph
  - $w_{ij} > 0$ weight of the i-j edge
- Functions f_i attaching values (i.e. labels/signals x_i) to nodes i
- Task: process the signals defined on the graph structure

UNIVERSITÀ DI PISA

Spectral Graph Convolution in 1 Slide

- Given a graph G, the eigendecomposition of its Laplacian provides an orthonormal basis U which allow to compute the graph convolution of its node signals $\boldsymbol{f}$ with a filter

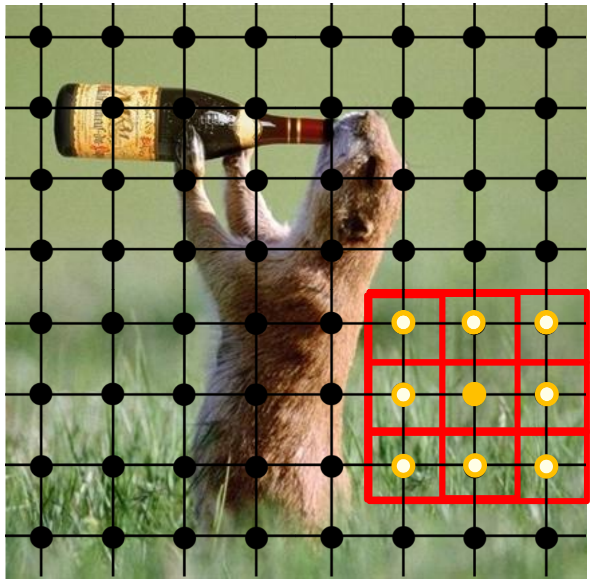
$$(\boldsymbol{f} *_G \boldsymbol{g}) = \mathcal{F}^{-1}\big(\mathcal{F}(\boldsymbol{f})\,\mathcal{F}(\boldsymbol{g})\big) = U\underbrace{\mathbf{W}(\lambda)U^T}\boldsymbol{f}$$

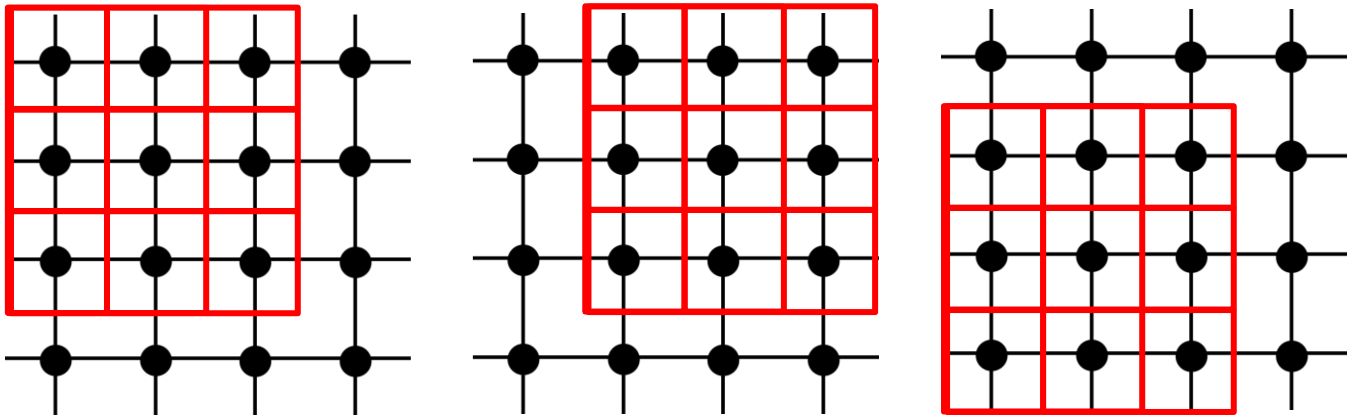
Convolutional filter **g** in spectral domain

Graph equivalent of the learnable CNN filter matrix **W**

Spectral convolution matrix **W** contains information on the graph Laplacian

A Graph View on (Image) Convolutions





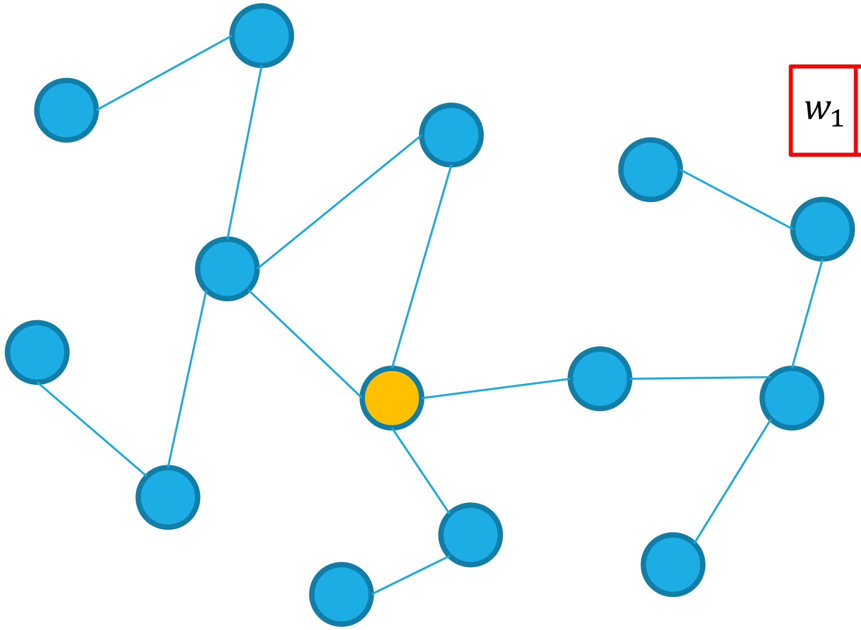
Visual convolutions are graph convolutions on a regular grid

Plus some key assumptions which make it difficult to directly apply them to graphs

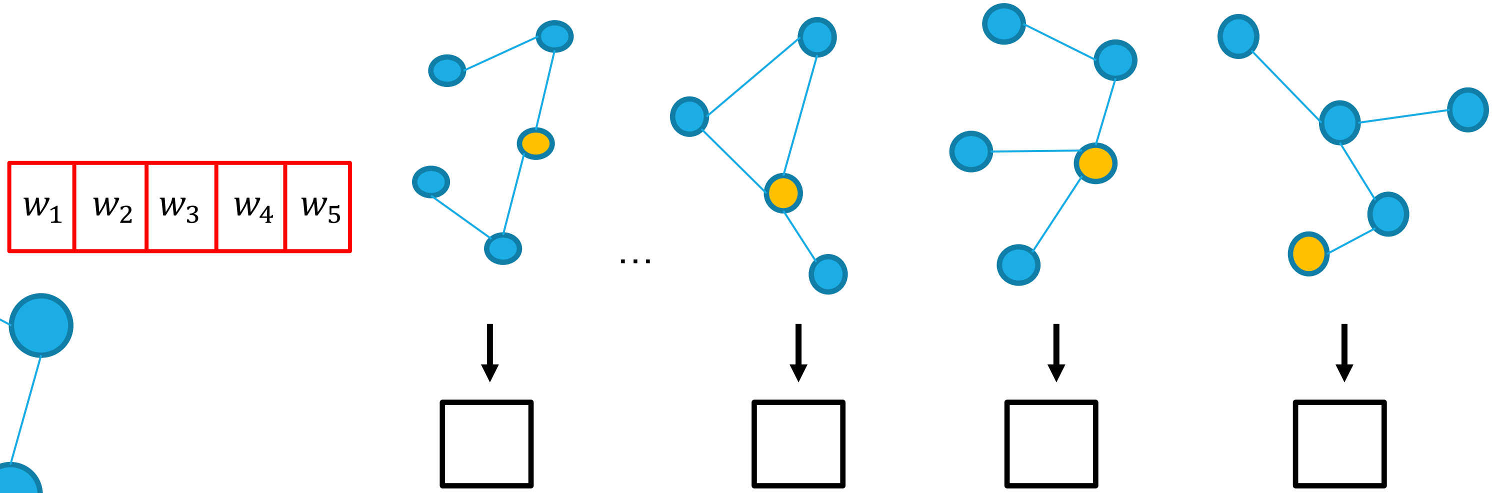
- Regular neighborhood
- Existence of a total node ordering

Node Neighborhoods

Example of 4-neighborhoods



convolutions



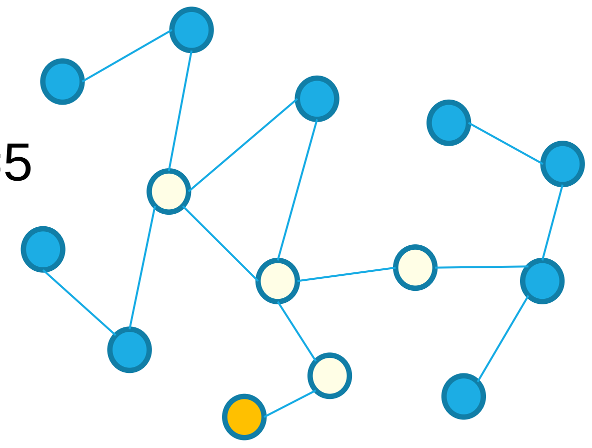
Neighborhoods depend on node ordering: how can I get coherent node ordering across multiple graphs?

PATCHY-SAN

Niepert, Ahmed, Kutzkov, ICML 2016

Leverage graph labelling techniques (e.g. Weisfeiler-Lehman) to determine a coherent ordering within the graph and between the graphs

Neighborhood for k=5



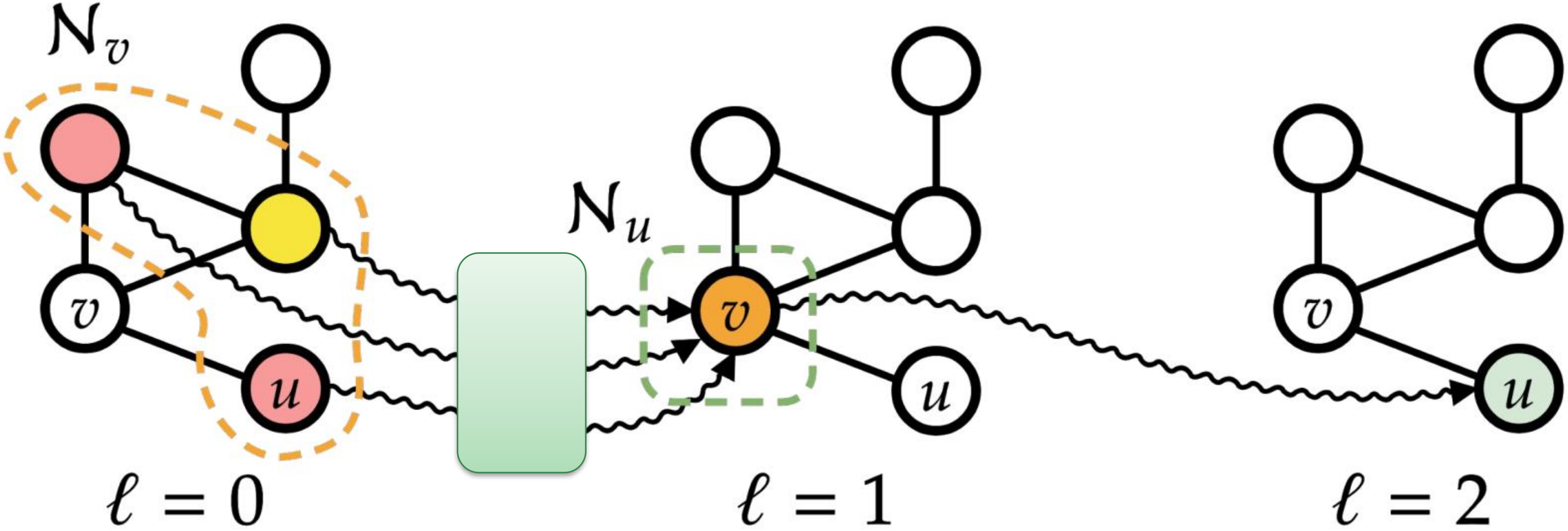
Parametric convolutional filter of size k

w_1	w_2	w_3	w_4	w_5

Determining a coherent ordering to match nodes to filter parameters in NP complete (graph normalization)

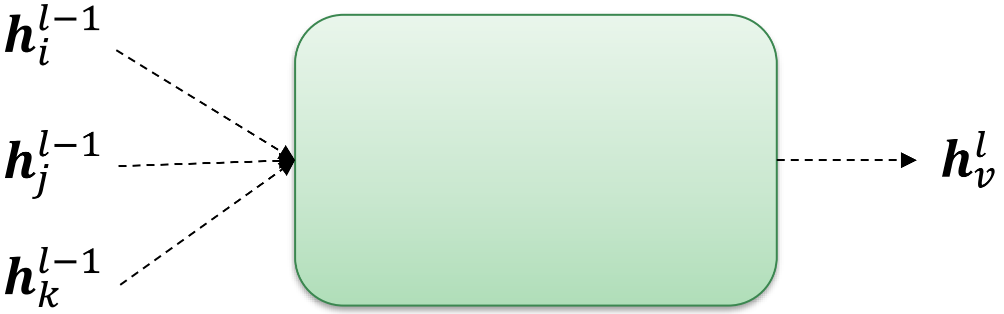
Message-Passing Graph Processing

Neighborhood Aggregation & Layering



What is inside of the Box?

A learning model of course (e.g. a neural network) including an aggregation function to handle size-varying neighborhoods



A simple model

$$\boldsymbol{h}_v^{l} = \sigma\left(\boldsymbol{W}_l AGG\left(\left\{\boldsymbol{h}_i^{l-1} : i \in N(v)\right\}\right), \widehat{\boldsymbol{W}}_l \boldsymbol{h}_v^{l-1}\right)$$

The graph convolutional layer

MLP/Linear

$$\underbrace{\mathbf{h}_v^{\ell+1}}_{\text{state}} = \overbrace{\phi^{\ell+1}}\Big(\mathbf{h}_v^{\ell},\ \underbrace{\Psi}_{\text{perm. invariant function}}(\{\overbrace{\psi^{\ell+1}}(\mathbf{h}_u^{\ell}) \mid u \in \mathcal{N}_v\})\Big)$$

Variants/extensions:

- **Edge-aware convolution**
- **Attention over neighbors**
- **Laplacian-normalized**

Model	Neighborhood Aggregation $\mathbf{h}_v^{\ell+1}$
NN4G [88]	$\sigma\Big(\mathbf{w}^{\ell+1^T}\mathbf{x}_v + \sum_{i=0}^{\ell}\sum_{c_k \in \mathcal{C}}\sum_{u \in \mathcal{N}_v^{c_k}} w_{c_k}^i * \mathbf{h}_u^i\Big)$
GNN [104]	$\sum_{u \in \mathcal{N}_v} MLP^{\ell+1}\Big(\mathbf{x}_u, \mathbf{x}_v, \mathbf{a}_{uv}, \mathbf{h}_u^{\ell}\Big)$
GraphESN [44]	$\sigma\Big(\mathbf{W}^{\ell+1}\mathbf{x}_u + \hat{\mathbf{W}}^{\ell+1}[\mathbf{h}_{u_1}^{\ell}, \ldots, \mathbf{h}_{u_{\mathcal{N}_v}}^{\ell}]\Big)$
GCN [72]	$\sigma\Big(\mathbf{W}^{\ell+1}\sum_{u \in \mathcal{N}(v)} \mathbf{L}_{vu}\mathbf{h}_u^{\ell}\Big)$
GAT [120]	$\sigma\Big(\sum_{u \in \mathcal{N}_v} \alpha_{uv}^{\ell+1} * \mathbf{W}^{\ell+1}\mathbf{h}_u\Big)$
ECC [111]	$\sigma\Big(\frac{1}{\lvert\mathcal{N}_v\rvert}\sum_{u \in \mathcal{N}_v} MLP^{\ell+1}(\mathbf{a}_{uv})^T\mathbf{h}_u^{\ell}\Big)$
R-GCN [105]	$\sigma\Big(\sum_{c_k \in \mathcal{C}}\sum_{u \in \mathcal{N}_v^{c_k}} \frac{1}{\lvert\mathcal{N}_v^{c_k}\rvert}\mathbf{W}_{c_k}^{\ell+1}\mathbf{h}_u^{\ell} + \mathbf{W}^{\ell+1}\mathbf{h}_v^{\ell}\Big)$
GraphSAGE [54]	$\sigma\Big(\mathbf{W}^{\ell+1}(\frac{1}{\lvert\mathcal{N}_v\rvert}[\mathbf{h}_v^{\ell}, \sum_{u \in \mathcal{N}_v} \mathbf{h}_u^{\ell}])\Big)$
CGMM [3]	$\sum_{i=0}^{\ell} w^i * \Big(\sum_{c_k \in \mathcal{C}} w_{c_k}^i * (\frac{1}{\lvert\mathcal{N}_v^{c_k}\rvert}\sum_{u \in \mathcal{N}_v^{c_k}} \mathbf{h}_u^i)\Big)$
GIN [131]	$MLP^{\ell+1}\Big((1 + \epsilon^{\ell+1})\mathbf{h}_v^{\ell} + \sum_{u \in \mathcal{N}_v} \mathbf{h}_u^{\ell}\Big)$

A Message-Passing view on Deep Graph Networks

Algorithm 13.1: Simple message-passing neural network

Input: Undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
Initial node embeddings $\{\mathbf{h}_n^{(0)} = \mathbf{x}_n\}$
$\mathrm{Aggregate}(\cdot)$ function
$\mathrm{Update}(\cdot, \cdot)$ function
Output: Final node embeddings $\{\mathbf{h}_n^{(L)}\}$

```
// Iterative message-passing
```

for $l \in \{0, \dots, L-1\}$ **do**

$\mathbf{z}_n^{(l)} \leftarrow \mathrm{Aggregate}\left(\left\{\mathbf{h}_m^{(l)} : m \in \mathcal{N}(n)\right\}\right)$

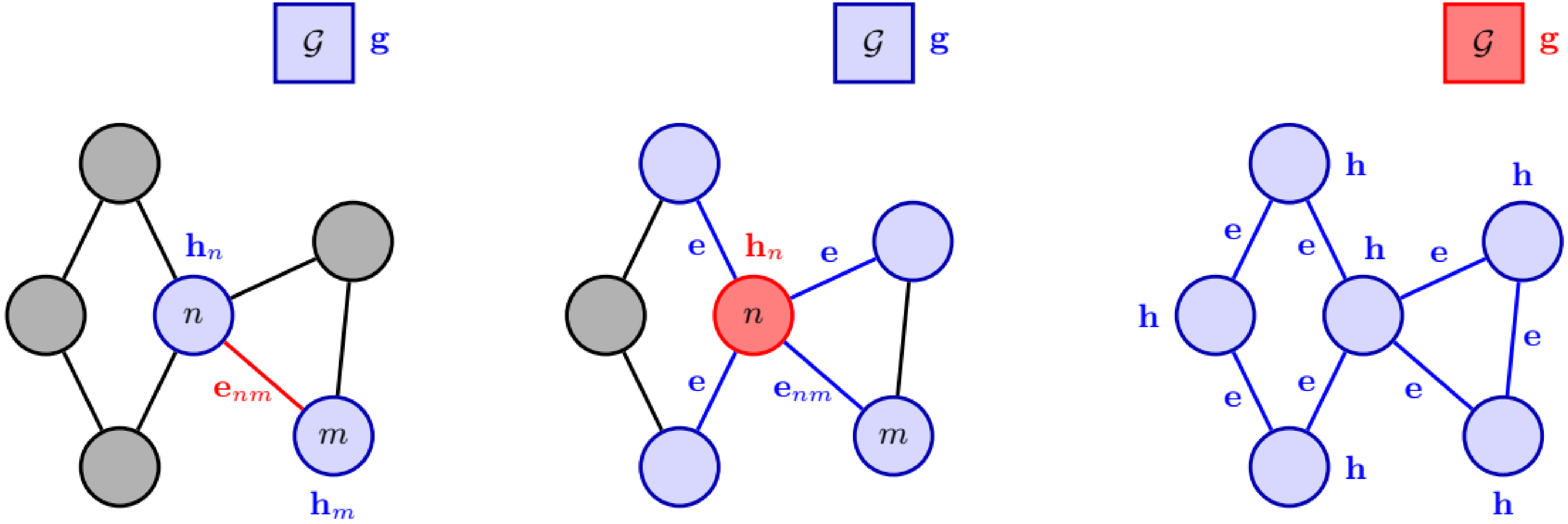
$\mathbf{h}_n^{(l+1)} \leftarrow \mathrm{Update}\left(\mathbf{h}_n^{(l)}, \mathbf{z}_n^{(l)}\right)$

end for

return $\{\mathbf{h}_n^{(L)}\}$

IN SUPREMÆ DIGNITATIS 1343 UNIVERSITÀ DI PISA

Different kinds of message-passing updates



Edge Node Graph

Graph Isomorphism Network (a.k.a. sum is better)

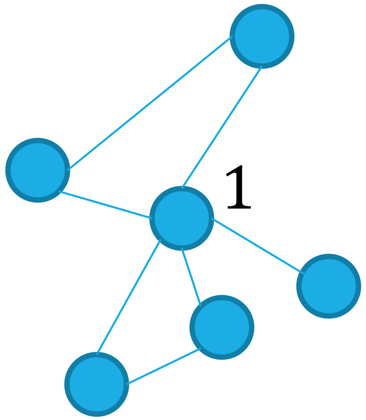
Xu et al, ICLR 2019

- A study of GNN expressivity w.r.t. WL test of graph isomorphism
- Choice of aggregation functions influences what structures can be recognized
- Propose a simple aggregation and concatenation model

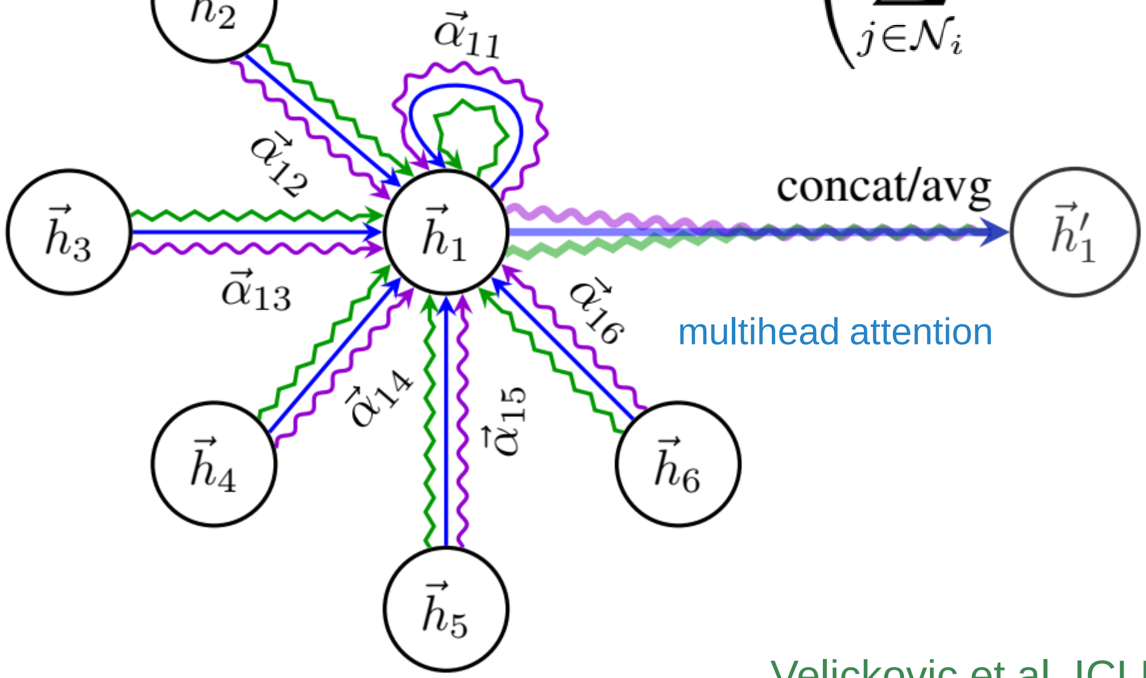
$$h_v^{(k)} = \mathrm{MLP}^{(k)}\left(\left(1+\epsilon^{(k)}\right) \cdot h_v^{(k-1)} + \sum_{u \in \mathcal{N}(v)} h_u^{(k-1)} \right)$$

$$h_G = \mathrm{CONCAT}\left(\mathrm{READOUT}\left(\left\{h_v^{(k)} | v \in G\right\}\right) | k = 0, 1, \cdots, K\right)$$

Graph Attention



$$\vec{h}'_i = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W} \vec{h}_j\right)$$

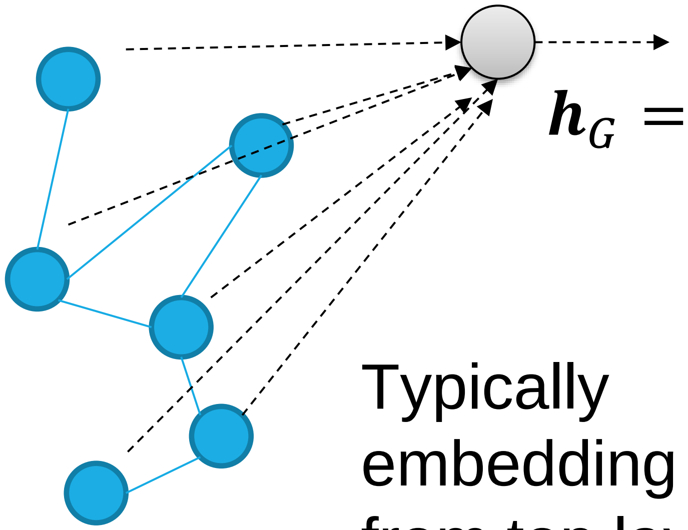


Learning to weight contribution of other nodes when aggregating to form the node embedding

Velickovic et al, ICLR 2018

Using Node Embedding

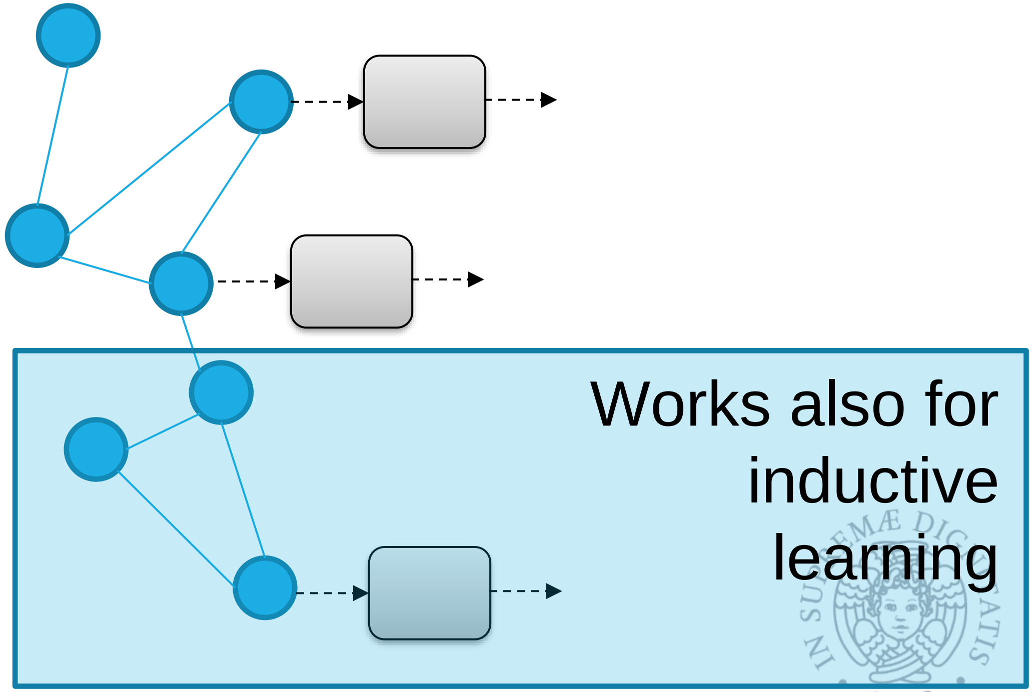
Aggregate all node embeddings to compute graph level predictions



$$\boldsymbol{h}_G = \sum_{i \in G} h_i^L$$

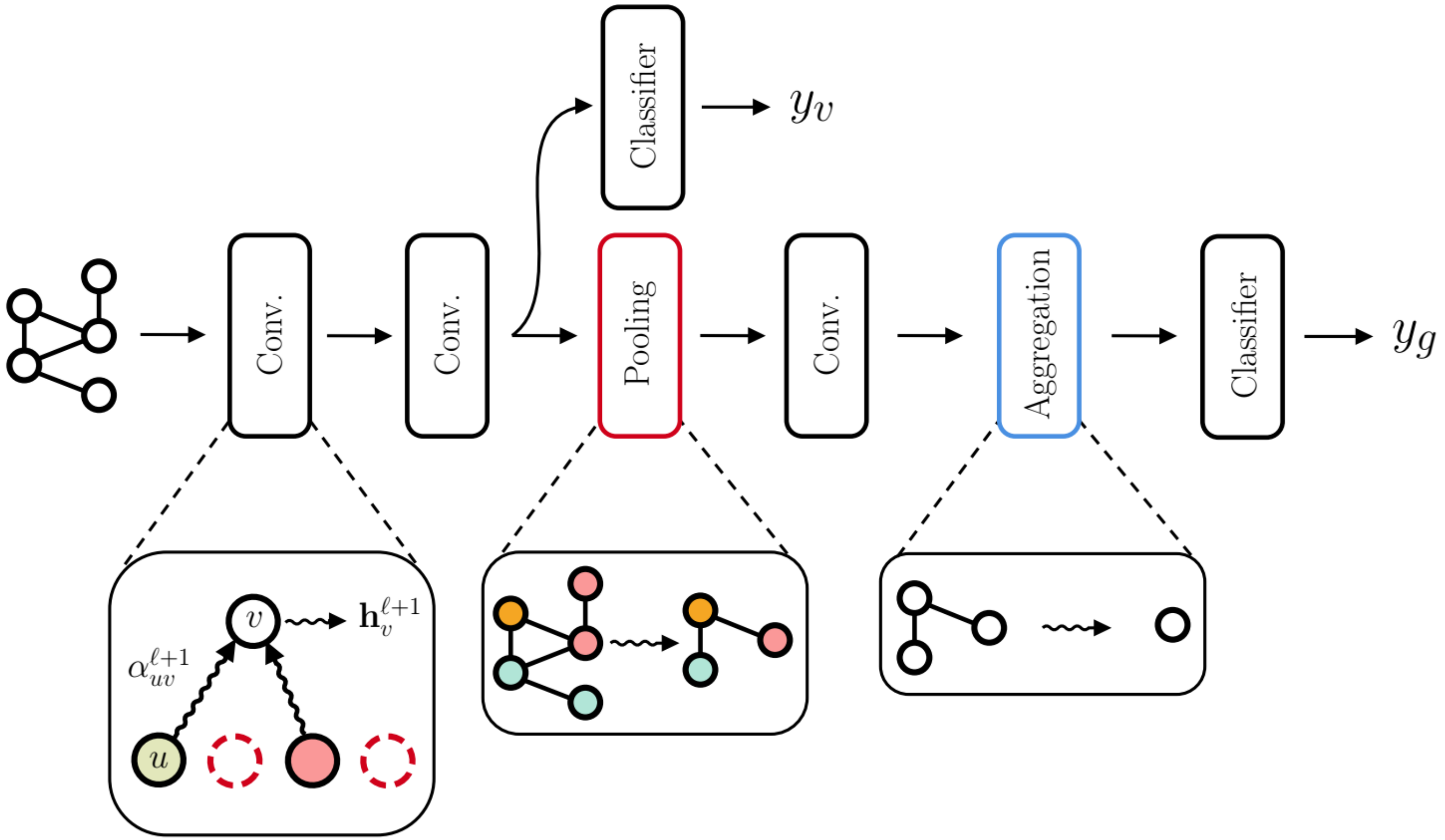
Typically embedding from top layer

Train node level predictors



Works also for inductive learning

Deep Graph Networks - The Complete Picture



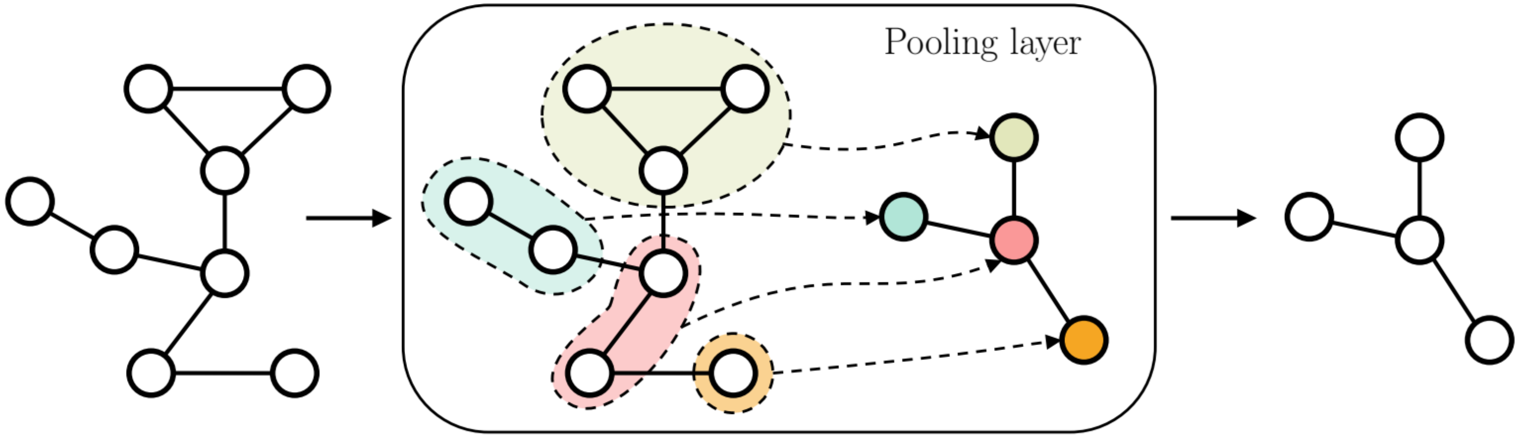
UNIVERSITÀ DI PISA

What About Pooling?

- Standard aggregation operates of predefined node subsets
- Ignore community/hierarchical structure in the graph
- Need graph coarsening (pooling) operators
  - Differentiable
  - Graph theoretical
  - Graph signature

Rex Ying et al, NIPS 2018

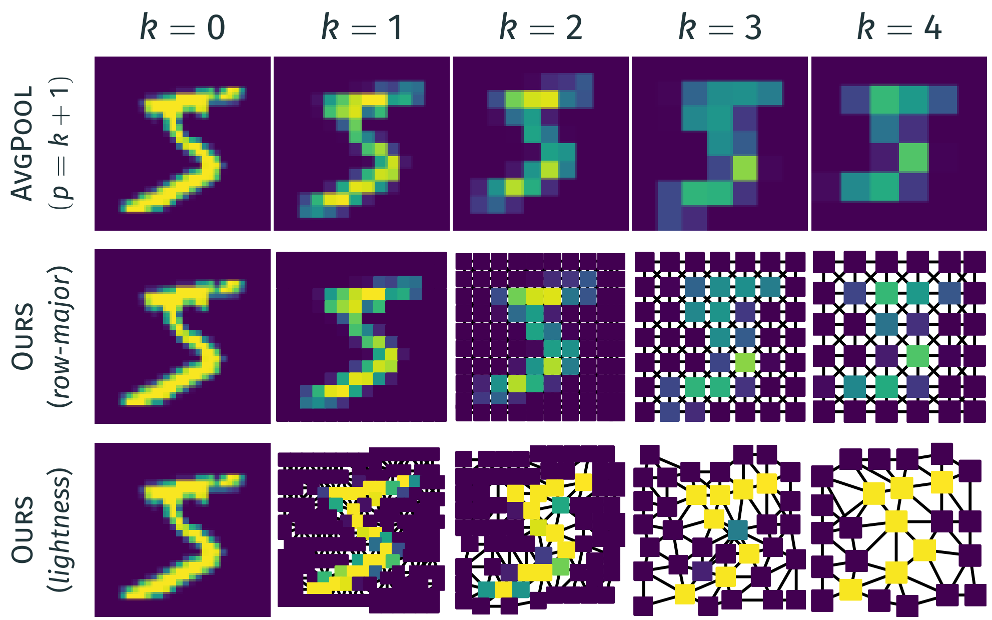
Bacciu et al, AAAI 2023

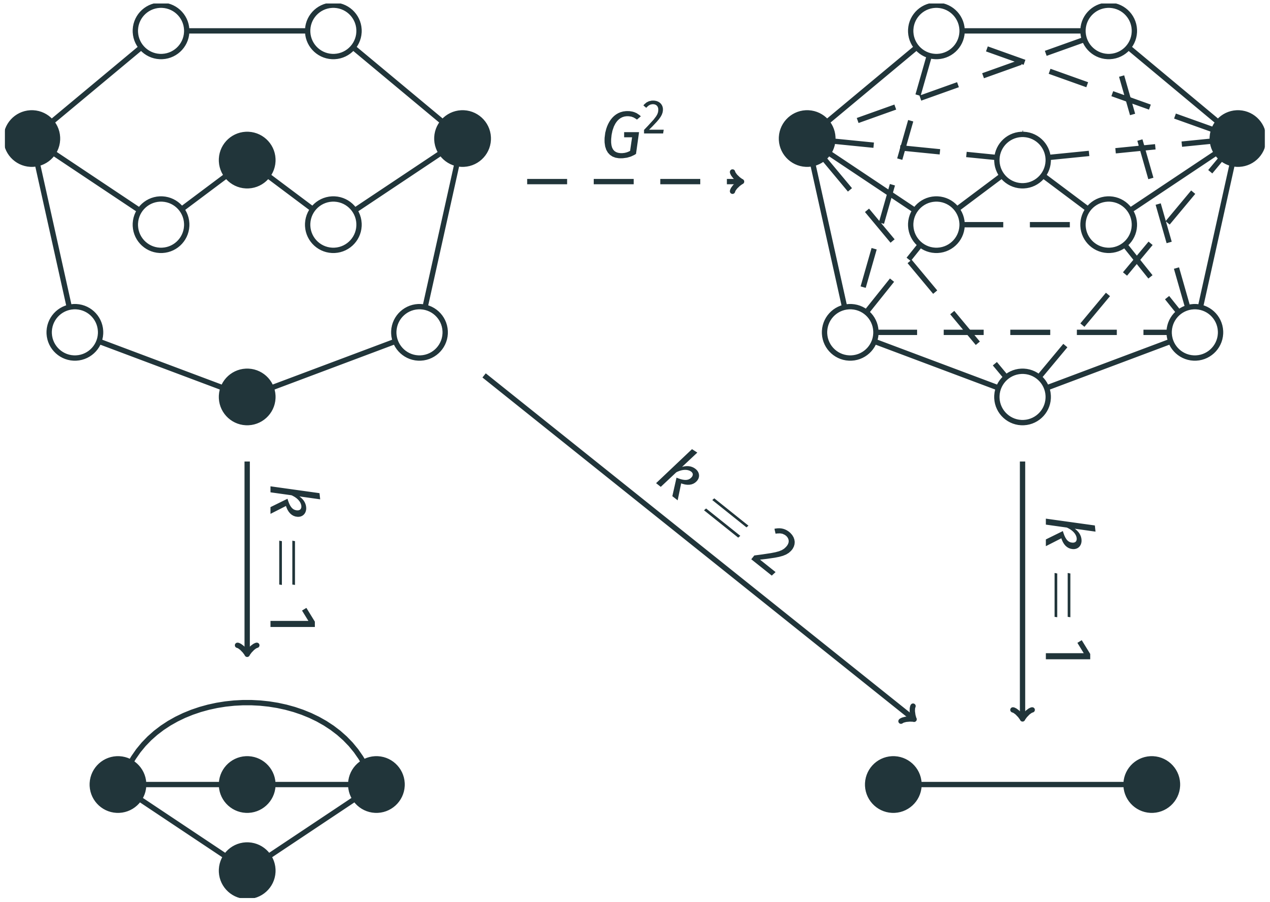


K-MIS Graph Coarsening

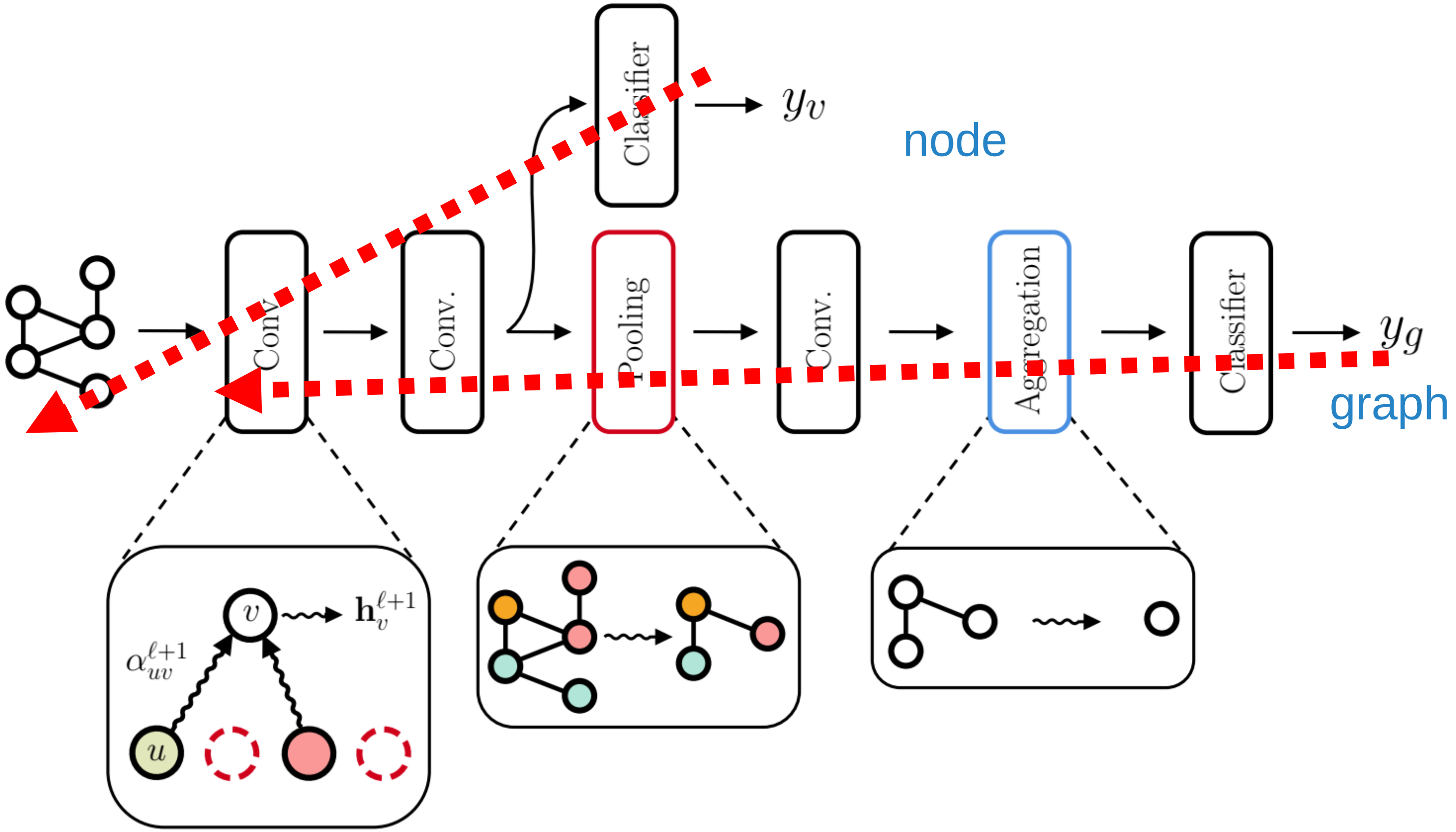
Bacciu et al, AAAI 2023

A proper extension of image-pooling to graphs with theoretical guarantees and scalability





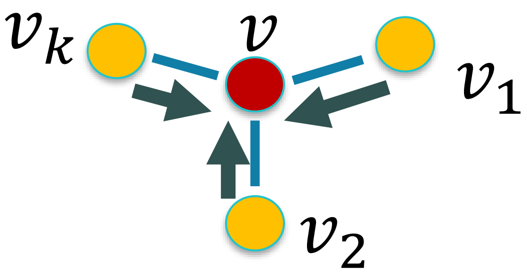
Training the Embedding

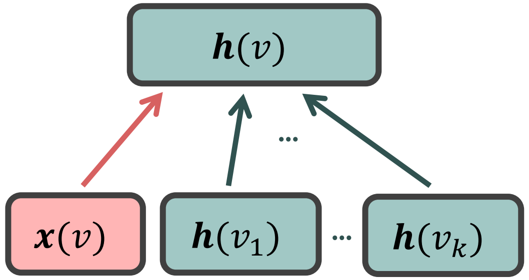


Backpropagate from the (graph or node level) error computed from the top layer embeddings to the early layers

Recurrent Graph Processing

Graph embedding by learning-free neurons





- Each vertex in an input graph is encoded by the hidden layer

input weight matrix

hidden weight matrix

$$\boxed{\mathbf{h}(v)} = \tanh\left(\boxed{\mathbf{V}}\,\boxed{\mathbf{x}(v)} + \sum_{v' \in N(v)} \boxed{\mathbf{W}}\,\boxed{\mathbf{h}(v')}\right)$$

embedding (state) of vertex v

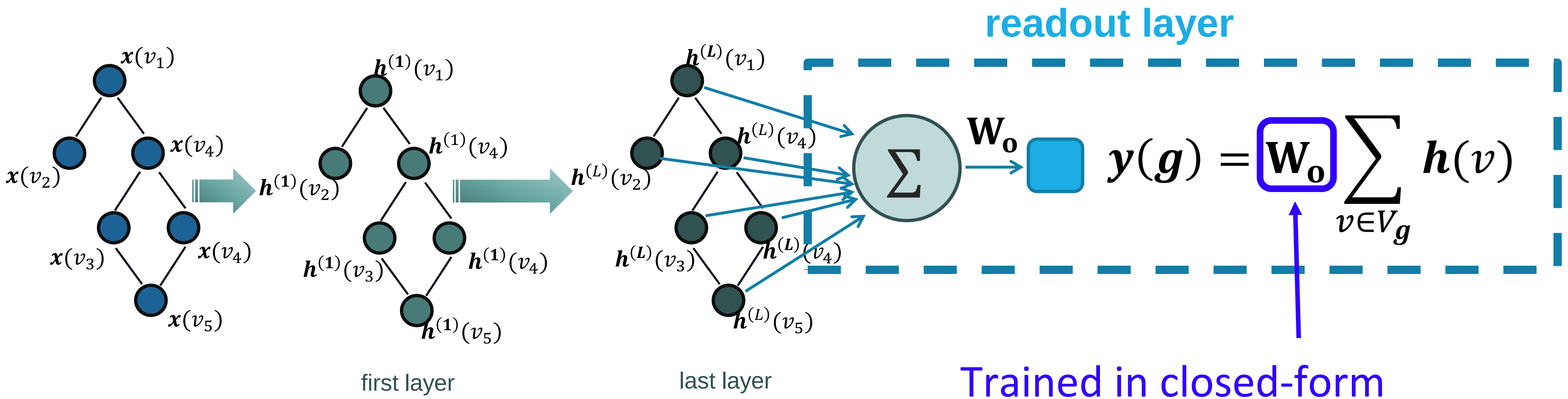
input feature of vertex v

embedding (state) of neighbors of vertex v

Need this to be contractive to ensure convergence of embedding

Deep Reservoirs for Graphs

Gallicchio & Micheli. *AAAI* 2020.



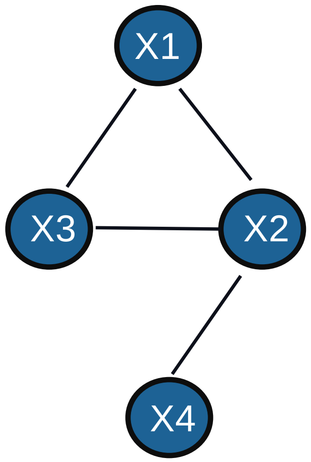
Deep reservoir embedding

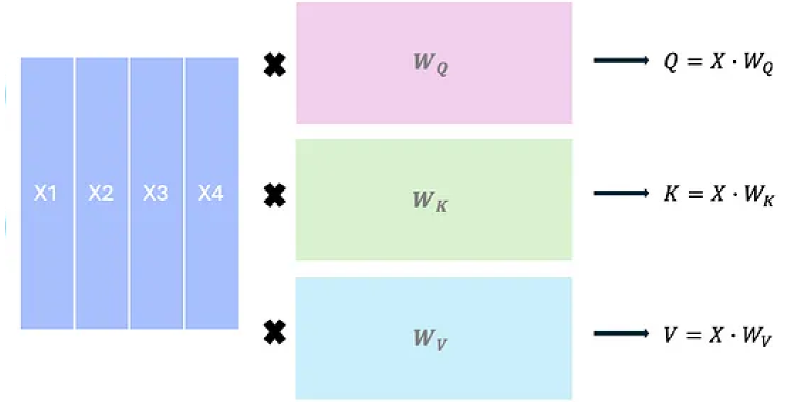
Trained in closed-form (e.g., pseudo-inversion, ridge regression)

Graph Transformers

Global Graph Attention

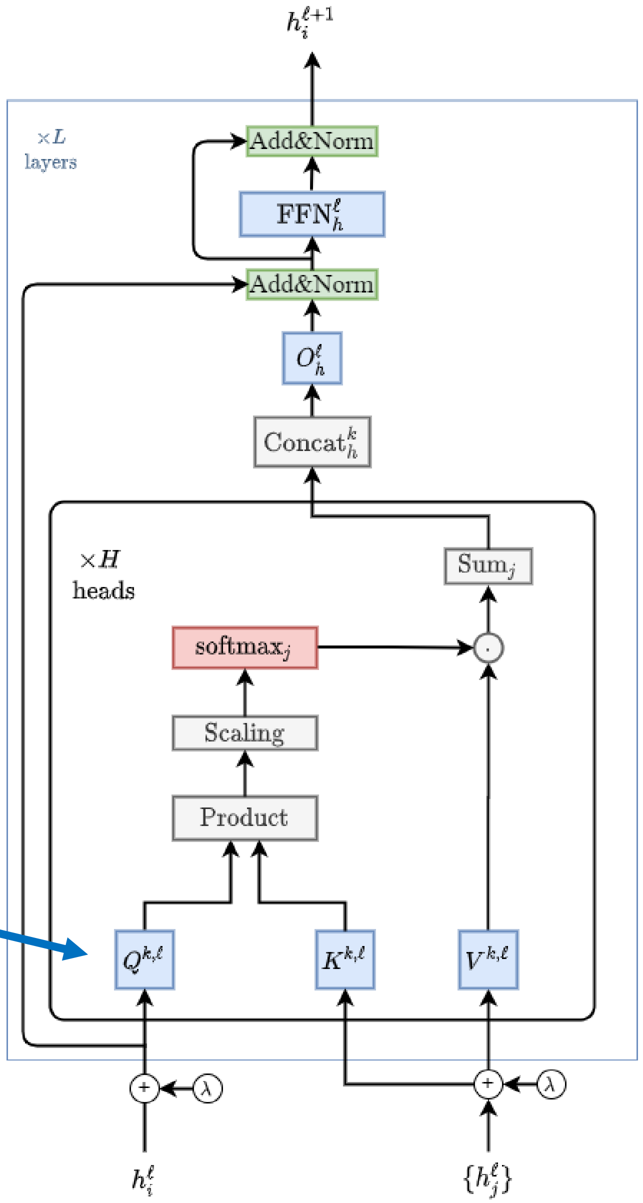
A direct generalization of standard attention from sequence tokens to nodes





$$Attention(Q, K, V) = \mathrm{softmax}\left(\frac{Q\,K^T}{\sqrt{d_k}}\right)V$$

Img adapted from Medium

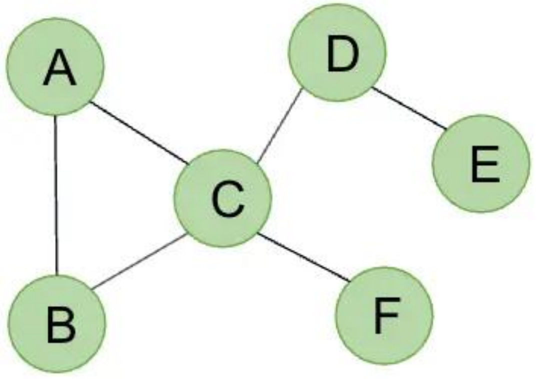


Graph Transformer Layer

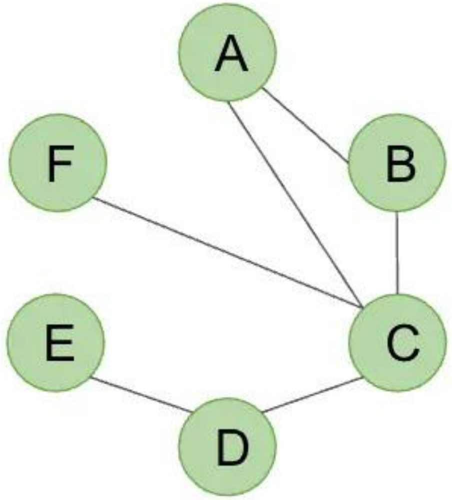
Dwivedi & Bresson, AAAI-WS 2021

Università di Pisa

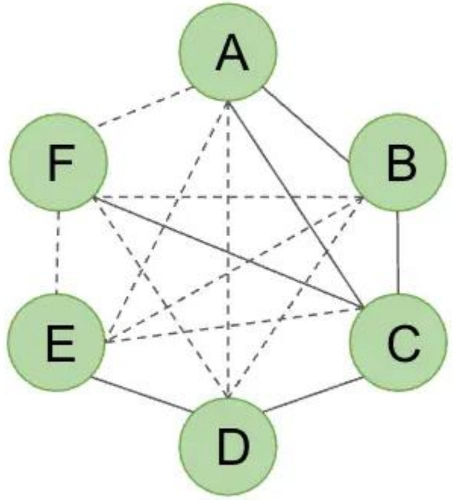
Wait! What is the inductive bias here?



Original graph



GAT Model

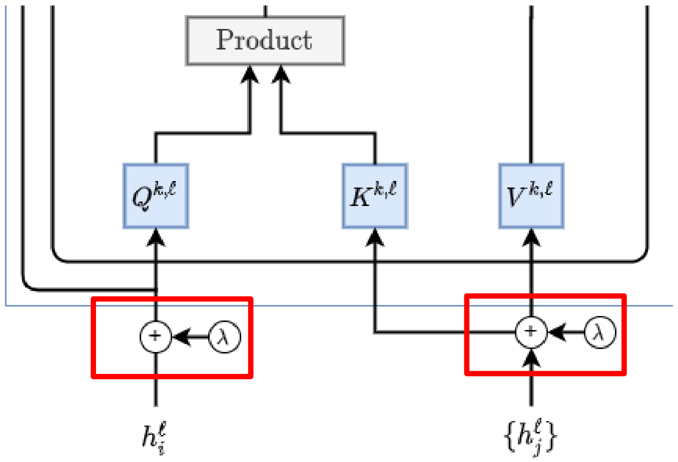


Graph Transformer

Img source: [Kumo AI]

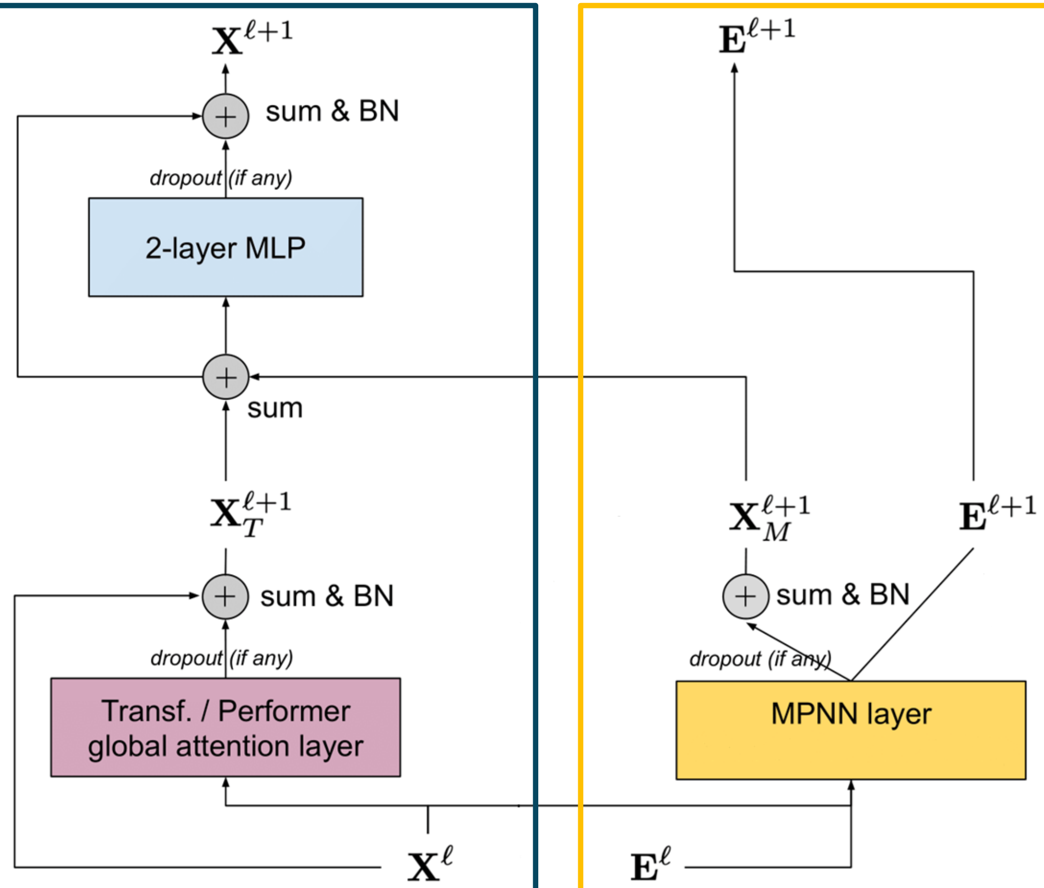
The return of positional encodings

- Transformers incorporate positional encodings to provide directional sense in a sequence (complete ordering)
- Graph have no complete ordering, but positional encodings can be used to reintroduce structural bias
  - Local PEs (Node-Level): Reflect a node's position relative to a specific substructure or cluster within the graph (e.g. reachability in random walks)
  - Global PEs (Node-Level): Node's position w.r.t. entire graph (e.g. Laplacian eigenvectors)
  - Relative PEs (Edge-Level): Represent the positional relationship between pairs of nodes (e.g. pairwise distances in random walks)
- Can be complemented with structural encodings providing insights into the local and global architecture of the graph



Graph Transformer Layer

GPS – Best of 2 worlds



Global graph attention easing information transfer

Local message passing with strong structure bias

Rampášek et al, NeurIPS 2022

Wrap-Up

Software

You can find most of the foundational models in this lecture implemented here

PyTorch geometric

DeepGraphLibrary

PyDGN

Our Python library for Deep Graph Networks

github.com/diningphil/PyDGN

Data (Benchmarks)

OPEN GRAPH BENCHMARK

- Pytorch Geometric and DGL integration
- Standardized splits and evaluators + leader-board
- Node, link and graph property prediction tasks

TUDataset

- Standardise assessment of existing benchmarks rather than inventing new ones
- Chemical, social, vision, synthetic, bioinformatics (with leader-board)
- Pytorch Geometric and DGL integration

Conclusions

- Deep learning for graphs is a now a consolidated research area
  - DGNs as natural extensions of convolutional and recurrent architectures to graphs
  - A candidate AI model for the integration of symbolic knowledge, numerical data and reasoning
- First wave of works (now almost over?) focusing mainly on
  - Different ways of implementing message passing and aggregation on static graphs
  - Graph reductions and pooling
  - Expressivity properties associated with different aggregation functions
  - Efficiency and efficacy of context creation and propagation by mixing local and global message passing

Next Lecture

- Generative graph learning
  - Probabilistic models on graphs
  - Graph VAE, graph language models and graph diffusion models
- Issues with information propagation on graphs
  - Oversmoothing, oversquashing and undereaching
  - Topological approaches
  - Dynamical systems approaches
- Spatio-temporal and dynamic graphs
- Applications