Deep Learning for Graphs

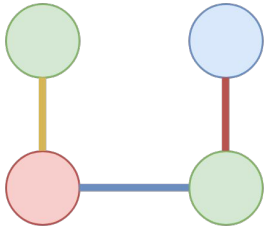
Trends & Open Questions

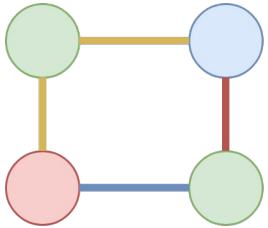
Federico Errica

Outline

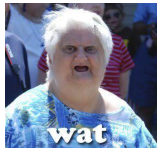
- What are graphs?
- Why ML on graphs?
- How?
- Some models
- Open questions (Projects / **Theses**)

The What

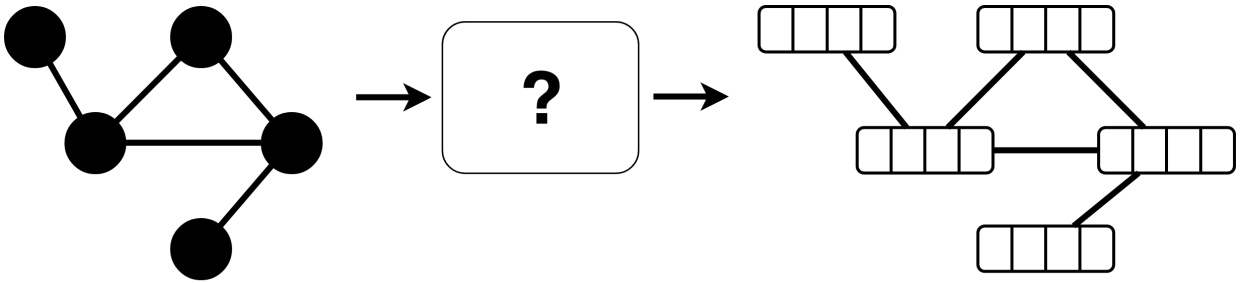




- Vertex $u \rightarrow$ Entity
  - Continuous / Discrete attributes
- Edge $(u,v) \rightarrow$ Relationship
  - Continuous / Discrete attributes ({0,1} **still** most common)
  - Directed / Undirected
- Generalizations: Multigraphs & Hypergraphs

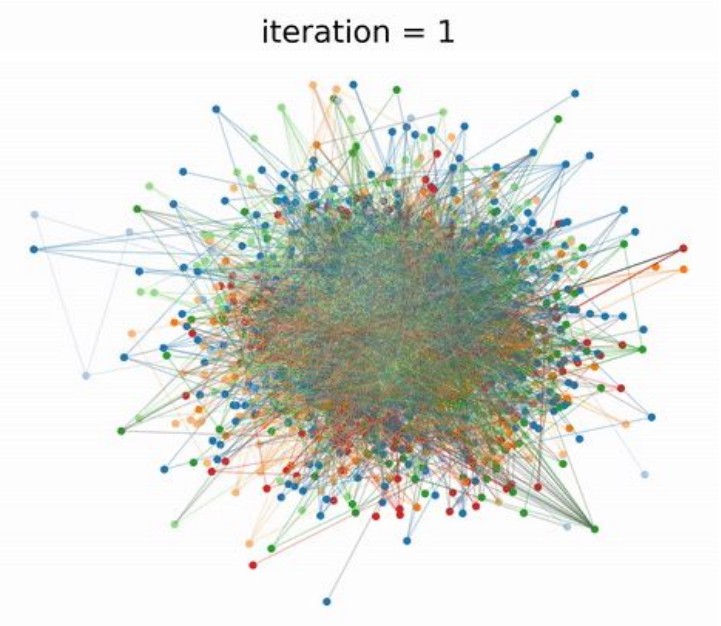


The What *(cont.)*



- Representation Learning on graphs
  - Vertex & Graph embeddings
- Supervised
  - Vertex/Graph **classification**/regression
- Unsupervised
  - Link Prediction
  - Clustering

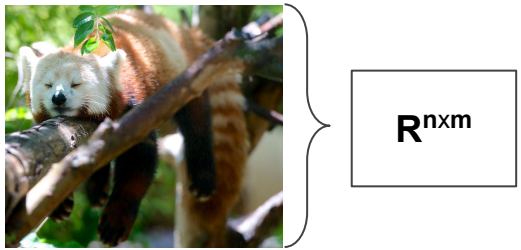
A visual example



Leow, Yao Yang, Thomas Laurent, and Xavier Bresson.
GraphTSNE: A Visualization Technique for Graph-Structured Data.
ICLR Workshop (2019).
https://leowyy.github.io/graphtsne

The Why

Flat

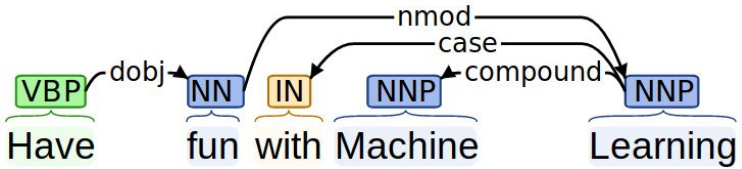


$R^{n \times m}$

Sequences

Have → fun → with → Machine → Learning

Trees



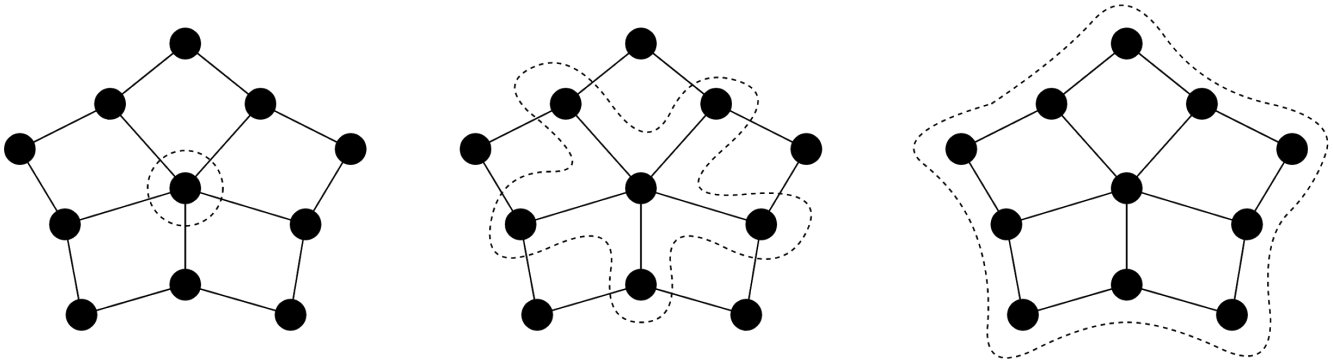
Graphs

The Why *(cont.)*

- Handle cyclic structures
  - No recursion!
- Variable **size**
- Variable **shape**
- **Efficiency**
- **No more** feature engineering
  - i.e. kernel methods

The **How** (in a nutshell)

- **Neighborhood Aggregation** to the rescue
- Use **layering** to spread **context** between vertices

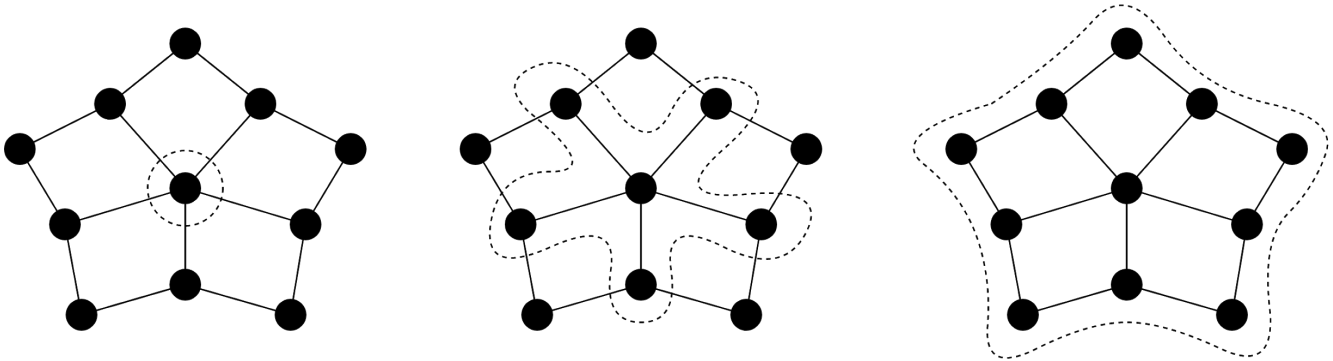


Layer 1 *Layer 2* *Layer 3*

- How can we aggregate neighbors?
  - Permutation-invariant functions over (multi)-sets
- How many layers do we need?

Resemblance to CNNs

- Convolution as neighborhood aggregation
  - On **regular grids**

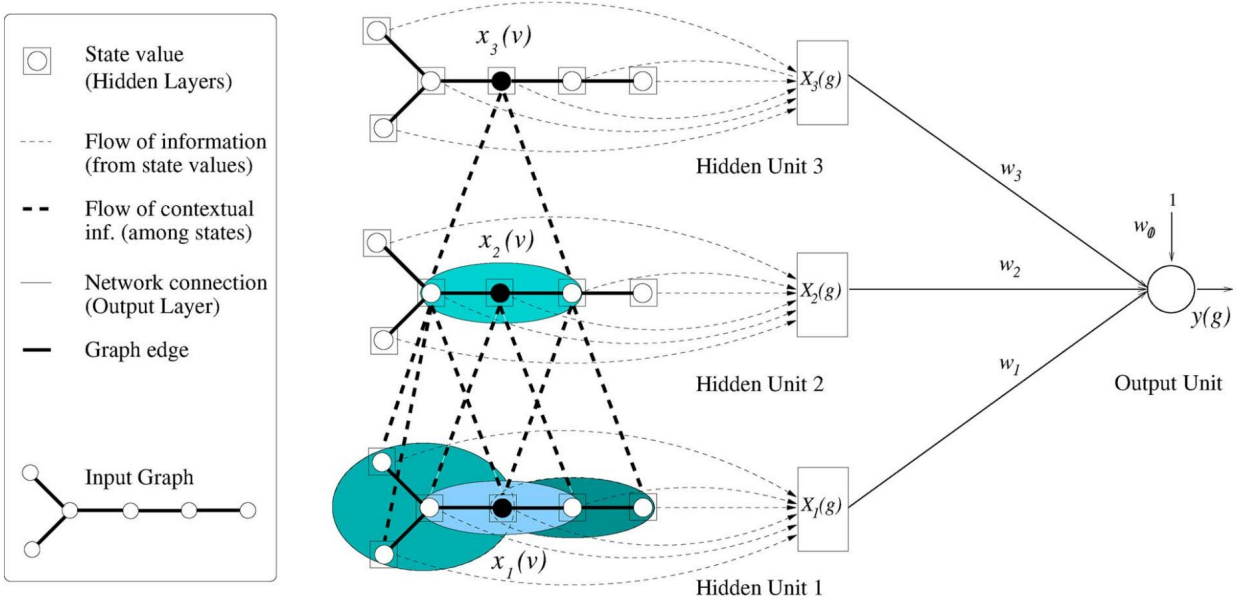


Layer 1 *Layer 2* *Layer 3*

- Layers increases the **local receptive field** of each vertex

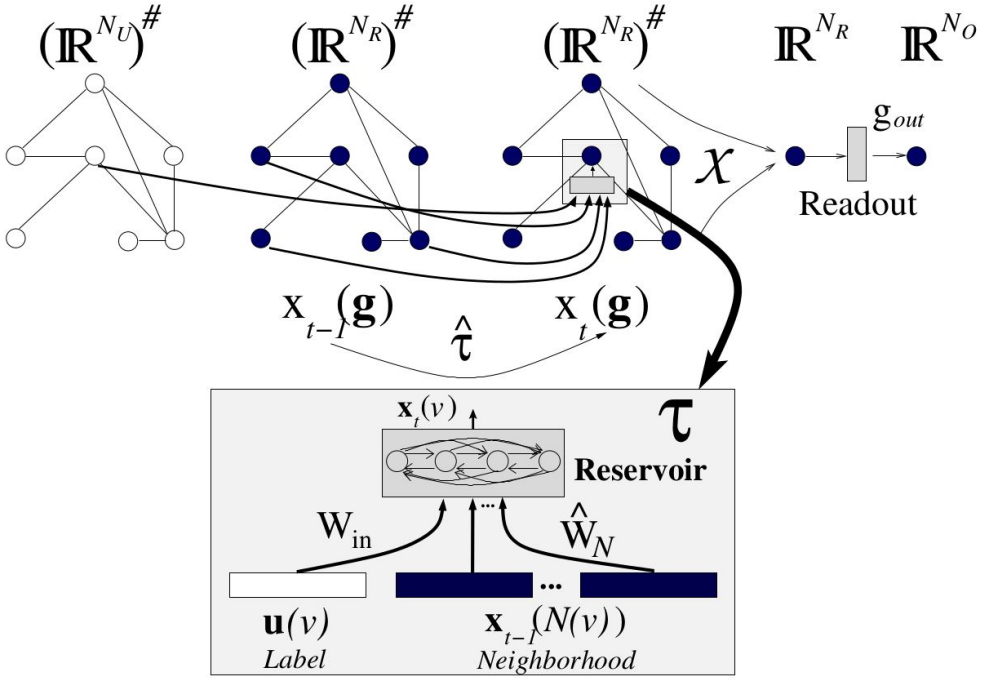
What we do: *NN4G (Micheli, 2009)*

- Constructive approach
  - Cascade Correlation
- Aggregation function
  - Sum
- The **first** GNN!



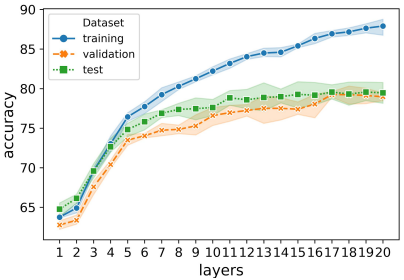
What we do: *GraphESN (Gallicchio & Micheli, 2010)*

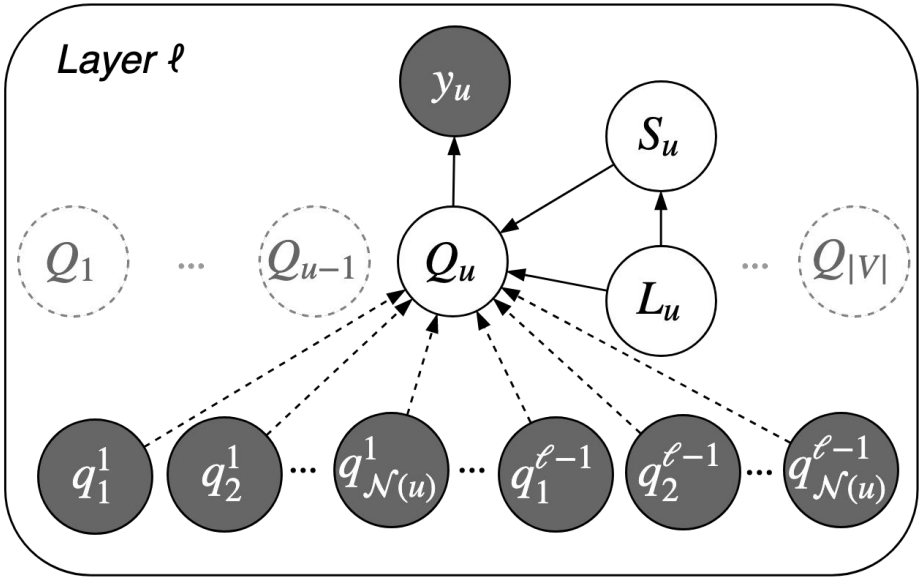
- Does not require training but for the output layer
- Let the Reservoir reach convergence
- Train a linear readout



What we do: *CGMM (Bacciu, Errica & Micheli, 2018)*

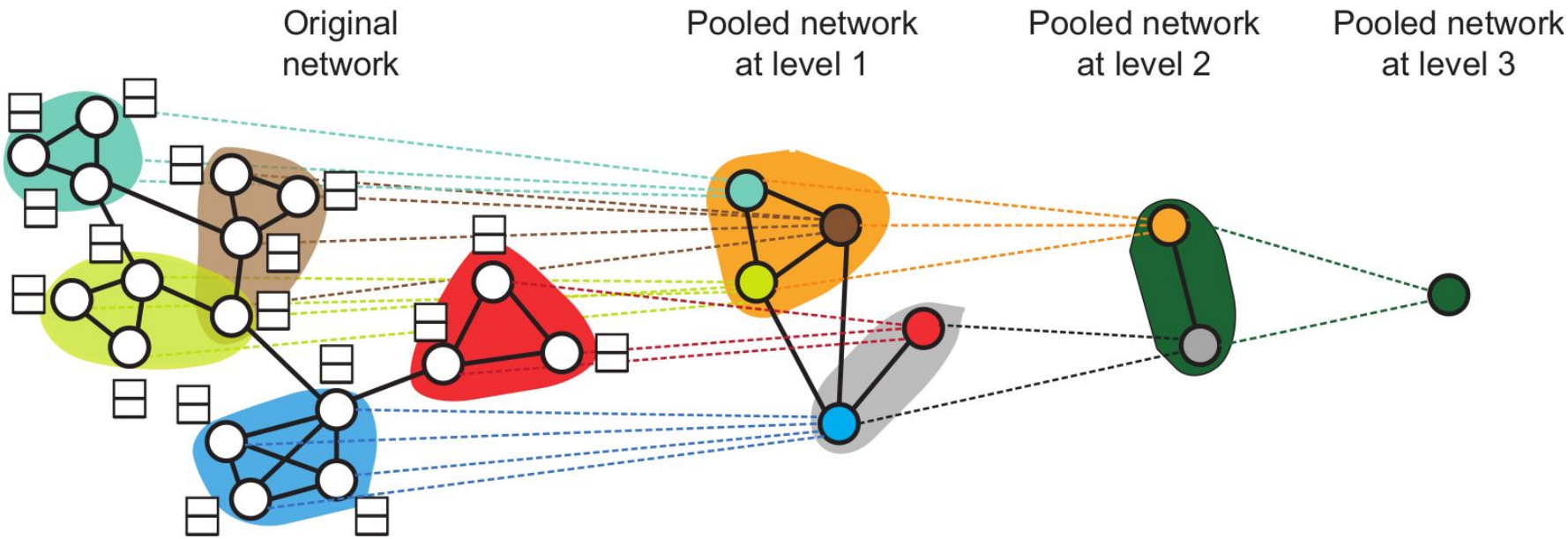
- A **deep** stack of probabilistic layers
- **Unsupervised** constructive approach
- **Switching Parent** approximation
  - Borrowed from Hidden Tree Markov Models
- It works well
  - State-of-the-art accuracy compared to GNNs
- CGMM exploits layering





DiffPool *(Ying et al., 2018)*

- Differentiable Pooling technique



On Aggregation Functions

From "*On the Limitations of Representing Functions on Sets", Wagstaff et al., 2019*

Theorem 2.8 (Countable case)**.** *Let* $f : 2^{\mathfrak{X}} \to \mathbb{R}$ *where* $\mathfrak{X}$ *is countable. Then* f *is permutation-invariant if and only if it is sum-decomposable via* $\mathbb{R}$*.*

Proof. Since $\mathfrak{X}$ is countable, each $x \in \mathfrak{X}$ can be mapped to a unique element in $\mathbb{N}$ by a function $c(x) : \mathfrak{X} \to \mathbb{N}$. Let $\Phi(X) = \sum_{x \in X} \phi(x)$. If we can choose ϕ so that Φ is injective, then we can set $\rho = f \circ \Phi^{-1}$, giving

$$f = \rho \circ \Phi$$
$$f(X) = \rho\big(\Sigma_{x \in X} \phi(x)\big)$$

i.e. f is sum-decomposable via $\mathbb{R}$.

Theorem 4.3 (Fixed set size)**.** *Let* $f : \mathbb{R}^M \to \mathbb{R}$ *be continuous. Then* f *is permutation-invariant if and only if it is continuously sum-decomposable via* $\mathbb{R}^M$*.*

Secondly, we can deal with variable set sizes $\leq M$:

Theorem 4.4 (Variable set size)**.** *Let* $f : \mathbb{R}^{\leq M} \to \mathbb{R}$ *be continuous. Then* f *is permutation-invariant if and only if it is continuously sum-decomposable via* $\mathbb{R}^M$*.*

Neighborhood aggregation!

M neighbors → M neurons!

Other works: GIN *(Xu et al., 2019)*

- Graph Isomorphism Network
  - As powerful as 1-dim WL test of graph isomorphism
  - But it can exploit continuous attributes
- Not able to distinguish **k-regular graphs**, k > 1
- Astonishing results **(under revision by me and Marco..)**
- Very nice theorems for aggregation on multi-sets
  - Similar to *Wagstaff et al., 2019*
- We can build other GNNs from here (Thesis?)

Open Questions *(nice & challenging)*

- Automatize hyper-parameter selections
- New pooling strategies
- CGMM extensions
- Design a **new GNN**
- Implement a Graph Neural Network (using Pytorch Geometrics)
  - Choose from a list of possible models

Thank you!

You can reach out to me via:

Email: federico.errica@phd.unipi.it (anytime)

Office: Room **328**, Department of Computer Science (late September)

Website: http://pages.di.unipi.it/errica/

Powered by CIML