Deep Graph Networks

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
- Applications & wrap-up
Introduction
Why Graphs?
Why Graphs?

Context is fundamental for the correct interpretation of information
Graph Structured Data

Structures are useful because allow to represent relationships in the data.
A Nomenclature Nightmare

Deep learning for graphs
- Graph neural networks
- Deep Graph Networks
- Graph CNN

Neural networks for graphs
- CNN for/on graphs

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

Structure classification/regression
Transductive tasks

Given a vectorial and/or structured input $x$, learn to generate a structured prediction $y \sim P(y|x)$.
An Historical (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
Scarselli et al, TNN 2009
Contractive - Graph Neural Networks (GNN)

- 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/
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

- Contextual Graph Processing
  - Contextual Graph Convolutions
  - Node embeddings

- Generative approaches
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 \( e_k \) 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
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 $f$ with a filter

$$ (f \ast_G g) = \mathcal{F}^{-1}(\mathcal{F}(f) \mathcal{F}(g)) = U\mathbf{W}(\lambda)U^T f $$

Convolutional filter $g$ in spectral domain

Graph equivalent of the learnable CNN filter matrix $\mathbf{W}$

Spectral convolution matrix $\mathbf{W}$ 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

Neighborhoods depend on node ordering: how can I get coherent node ordering across multiple graphs?
Leverage graph labelling techniques (e.g. Weisfeiler-Lehman) to determine a coherent ordering within the graph and between the graphs.
Contextual 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

\[
    h^1_v = \sigma(W^{AGG}_{l}\{h^{l-1}_i: i \in N(v)\}), \bar{W}_l h^{l-1}_v)
\]
The graph convolutional layer

\[ h_{v}^{\ell+1} = \phi^{\ell+1}(h_{v}^{\ell}, \Psi(\{\psi^{\ell+1}(h_{u}^{\ell}) | u \in \mathcal{N}_{v}\})) \]

Variants/extensions:
- Edge-aware convolution
- Attention over neighbors
- Laplacian-normalized

<table>
<thead>
<tr>
<th>Model</th>
<th>Neighborhood Aggregation ( h_{v}^{\ell+1} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>NNAG [88]</td>
<td>( \sigma \left( w_{x}^{\ell+1} x_{v} + \sum_{u \in \mathcal{N}<em>{v}} w</em>{e_{u}} x_{u} \right) )</td>
</tr>
<tr>
<td>GNN [104]</td>
<td>( \sum_{u \in \mathcal{N}<em>{v}} MLP</em>{\psi}^{\ell+1}(x_{u}, x_{v}, a_{uv}, h_{v}) )</td>
</tr>
<tr>
<td>GraphESN [44]</td>
<td>( \sigma \left( W_{x}^{\ell+1} x_{v} + W_{e_{u}}^{\ell+1}[h_{u}, \ldots, h_{u_{\mathcal{N}_{v}}}] \right) )</td>
</tr>
<tr>
<td>GCN [72]</td>
<td>( \sigma \left( W_{x}^{\ell+1} \sum_{u \in \mathcal{N}(v)} h_{u} \right) )</td>
</tr>
<tr>
<td>GAT [120]</td>
<td>( \sigma \left( \sum_{u \in \mathcal{N}<em>{v}} \alpha</em>{uv}^{\ell+1} + W_{x}^{\ell+1} h_{u} \right) )</td>
</tr>
<tr>
<td>ECC [111]</td>
<td>( \sigma \left( \frac{1}{\theta_{u}} \sum_{v \in \mathcal{N}<em>{u}} MLP</em>{\psi}^{\ell+1}(a_{uv}) h_{v} \right) )</td>
</tr>
<tr>
<td>R-GCN [105]</td>
<td>( \sigma \left( \sum_{u \in \mathcal{N}<em>{v}} \frac{1}{\theta</em>{u}} \left( W_{x}^{\ell+1} h_{u} + W_{e_{u}}^{\ell+1} h_{v} \right) \right) )</td>
</tr>
<tr>
<td>GraphSAGE [84]</td>
<td>( \sigma \left( W_{x}^{\ell+1} \left( \frac{1}{\theta_{u}} h_{u} + \sum_{v \in \mathcal{N}<em>{u}} h</em>{v} \right) \right) )</td>
</tr>
<tr>
<td>CGMM [3]</td>
<td>( \sum_{u \in \mathcal{N}<em>{v}} \frac{1}{\sum</em>{v \in \mathcal{N}<em>{u}} h</em>{v}} \left( W_{x}^{\ell+1} h_{u} + \sum_{v \in \mathcal{N}<em>{u}} h</em>{v} \right) )</td>
</tr>
<tr>
<td>GIN [131]</td>
<td>( MLP_{\psi}^{\ell+1}(1 + e^{\ell+1}) h_{v} + \sum_{u \in \mathcal{N}<em>{v}} h</em>{u} )</td>
</tr>
</tbody>
</table>
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 $\{h_n^{(0)} = x_n\}$
Aggregate($\cdot$) function
Update($\cdot$, $\cdot$) function

**Output:** Final node embeddings $\{h_n^{(L)}\}$

// Iterative message-passing
for $l \in \{0, \ldots, L - 1\}$ do
\[ z_n^{(l)} \leftarrow \text{Aggregate} \left( \left\{ h_m^{(l)} : m \in \mathcal{N}(n) \right\} \right) \]
\[ h_n^{(l+1)} \leftarrow \text{Update} \left( h_n^{(l)}, z_n^{(l)} \right) \]
end for
return $\{h_n^{(L)}\}$
Different kinds of message-passing updates

Edge

Node

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

- 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

\[ h_v^{(k)} = \text{MLP}^{(k)} \left( (1 + \epsilon^{(k)}) \cdot h_v^{(k-1)} + \sum_{u \in N(v)} h_u^{(k-1)} \right) \]

\[ h_G = \text{CONCAT}(\text{READOUT} \left( \{ h_v^{(k)} | v \in G \} \right) | k = 0, 1, \cdots, K) \]
Graph Attention

Learning to **weight contribution** of other nodes when aggregating to form the node embedding.
Using Node Embedding

Aggregate all node embeddings to compute graph level predictions

\[ h_G = \sum_{i \in G} h_{i}^L \]

Typically embedding from top layer

Train node level predictors

Works also for inductive learning
Deep Graph Networks - The Complete Picture
What About Pooling?

- Standard aggregation operates on 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

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

Bacciu et al, AAAI 2023
Training the Embedding

Backpropagate from the (graph or node level) error computed from the top layer embeddings to the early layers.
Generative Approaches
Graph Generation

Generate a prediction that is itself a graph

Adjacency-based

Structure-based
Graph Variational Autoencoder

GraphVAE generates adjacency matrix up to k vertices along with the relevant edge/node features (for molecular data)

Argmax a.k.a. sampling \(\Rightarrow\) non-differentiable

Simonovsky, Komodakis, ICLR-WS 2018
Language-Based Graph Generation

Generate a graph node-by-node and edge-by-edge through a sequential approach.

Sample output dist.

Start token

Initialize 2nd net with state of 1st
Generate Molecules by Fragmentation

- Molecule is scanned in SMILES order
- Find first breakable bond
- Break the molecule at that bond, set aside leftmost fragment
- Proceed recursively on rightmost fragment

- Order is deterministic and the molecule can be reconstructed
- Keep a vocabulary of all possible fragments found in a dataset
- Graphs are transformed into fragment sequences
Diffusion Models for Graphs

\[
\begin{align*}
X^1 &\sim Cat(X^0Q_X^1) \\
E^1 &\sim Cat(E^0Q_E^1)
\end{align*}
\]

\[
G^0 = (X^0, E^0)
\]

\[
\begin{align*}
(X^0, E^0) &\sim p_\theta(G^0|G^1, y) \\
(X^{T-1}, E^{T-1}) &\sim p_\theta(G^{T-1}|G^T, y)
\end{align*}
\]

Node/edge transition matrices
Forward noising
Backward denoising
Conditioning information (classifier free)

Ninniri, Podda, Bacciu AAAI-WS 2024
Applications
Predicting Properties of Chemical Compounds

Toxicity
Solubility
Quantum mechanical properties

Simulation methods
\( \sim 10^{-2} \text{ seconds} \)
\( \sim 10^3 \text{ seconds} \)

Micheli et al, JCICS 2001
Duvenaud, Maclaurin et al, NIPS 2015
Gilmer et al, ICML 2017
Generating Molecules

Podda, Bacciu, Micheli, AISTATS 2020

Fragment-based deep molecule generation
Knowledge graphs

A natural way of representing known entities and relationships in a domain

Node/link embeddings are numerical encodings of entities and relationships
Side Effects of Drug Combinations

Analyzing a multimodal graph of interactions

- Drug-drug
- Drug-protein
- Protein-protein

Zitnik, Agrawal, Leskovec, Bioinformatics 2018
Recommendation Systems

...and other kinds of social network analyses
Relational Stock Learning
Point Clouds – Semantic Segmentation

Build point cloud graphs and train semantic class predictors based on vertex embeddings.

Landrieu, Simonovsky, CVPR 2018
Analysis of ICT systems/Blockchains
Advanced Topics
Unsupervised Graph Embeddings

- Learn unsupervised node and graph embeddings
- Requiring less supervised labelling
- Reusing embeddings across multiple tasks

- Mix supervised and unsupervised modules

Probabilistic  Unsupervised  Deep  Supervised

Contextual Graph Markov Model (CGMM)

Fixed-size graph representation

Bacciu, Errica, Micheli, ICML 2018
Contextual Graph Markov Model (CGMM)
Incremental Construction

1. Map the graph to the model (base case)

2. Perform inference and freeze states

3. Add a new layer and use frozen states as observed variables in the graphical model

Go back to step 2
Computing embedding

✓ Finding the most likely **state assignment**

\[
\max_i P(y_u | Q_u = i) P(Q_u = i | q_{N(u)})
\]

✓ The inferred latent states are used as observable variables in subsequent layers

✓ A **fixed-size vector of states frequencies** as graph encoding

\[
\text{Frequency vector: } \begin{bmatrix} 2 & 1 & 0 & 2 \end{bmatrix}
\]

# Hidden states = 4

Inferred states (as colors)
CGMM Layer Training

A maximum likelihood approach to learning

\[ \mathcal{L} = \prod_{g \in G} \prod_{u \in g} \sum_{i} P^l(y_u | Q_u = i) P^l(Q_u = i | q^\text{Lprec}_{\mathcal{N}(u)}(g)) \]

Split by layer and by arc

Trained by Expectation-Maximization

Assumption: i.i.d. graphs

- Emission distrib.
- Switching Parents distrib.
- Transition distrib.
CGMM – Depth Matters...

...possibly more than width

Bacciu, Errica, Micheli, JMLR 2020
The Infinite CGMM

✓ Hierarchical Dirichlet process to sample (potentially) infinitely many hidden states

✓ Automatically learn the size of node embedding space from data

✓ Choice of observations’ groups determined by neighbors’ states

✓ Batch version for larger datasets

Castellana, Errica, Bacciu, Micheli , ICML 2022
ICGMM – Finer grained control on hidden space

CHosen States Per Layer

Cumulative Graph Embedding Size
Graph embedding by learning-free neurons

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

\[
\mathbf{h}(\nu) = \tanh(\mathbf{V} \mathbf{x}(\nu) + \sum_{\nu' \in N(\nu)} \mathbf{W} \mathbf{h}(\nu'))
\]

Need this to be contractive to ensure convergence of embedding
Deep Reservoirs for Graphs

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

Deep reservoir embedding
A Dynamical Systems View on Deep Graph Networks

- Node message passing can also be seen as a discretization of a continuous dynamical process

\[
\begin{aligned}
\frac{\partial x_u(t)}{\partial t} &= f_G(x_u(t)) & t \in [0, T], \\
x_u(0) &= x^0_u \in \mathbb{R}^d
\end{aligned}
\]
A Dynamical Systems View on Deep Graph Networks

- Node message passing can also be seen as a discretization of a continuous dynamical process.
- The graph neural network has as many layers as the length of the unfolded ODE.
- Neural (Graph) ODE
Non-Dissipative Propagation

- An intermediate step is fundamental before working with dynamic graphs to obtain a **stable and non-dissipative message passing**

- The primary challenge in the graph representation learning is **capturing and encoding structural information** in the learning model

- Exploiting **local interactions between nodes might not be enough** to learn representative embeddings
  - A specific range of node interactions is required to effectively solve the problem
  - The DGN requires a specific number (possibly large) of layers
  - Over-squashing problem
Non-Dissipative Message Passing

- Many-layer networks are needed to capture long range node interactions into representative embeddings
- Leverage the ODE formulation of DGNs to optimize forward and backward message propagation

Gravina, Bacciu, Gallicchio, ICLR 2023

DAVIDE BACCIU - ISPR COURSE
Non-Dissipation by Anti-Symmetric Parameterization

Forward Euler discretization of Graph ODE

\[ x_u^{\ell} = x_u^{\ell-1} + \epsilon \sigma \left( (W - W^T) - \gamma I \right) x_u^{\ell-1} + \Phi \left( X^{\ell-1}, N_u \right) + b \]

- Step size
- Monotonically non-decreasing activation function, e.g., tanh, relu
- Anti-symmetric weight matrix allowing stable and non-dissipative behavior of the ODE (eigenvalues of the Jacobian are all imaginary)
- Neighborhood aggregator (any standard DGNs)
- Diffusion term that preserves the stability of the discretized system
Learning with Dynamic Graphs

Graphs evolve with time in feature, connectivity and topology

- Spatio-temporal networks
- Graph streams

R. Trivedi et al. ICRL 2019
Gravina & Bacci, TNNLS 2024 (Survey)
Dynamic Graphs Vs Static DGNs

- DGNs cannot be directly applied to all real-life graphs
  - Most real-life graphs are dynamic
  - Majority of DGN approaches assume that the input graph is static
- Ignoring temporal information can make the problem impossible to solve
- **Objective**: develop methods that are able to exploit both spatial and temporal information
Common Tasks with Dynamic Graphs

❖ Future link/node prediction
  ● Predict at time $t + k$

❖ Path classification
  ● E.g. predict path congestion

❖ Event time prediction
  ● When an event will occur?

❖ Imputation
Irregular Graph Streams

Graphs occur in the stream as irregular events
Can be treated naturally by a neural ODE that propagates node signals between even occurrences

\[
x_{u}^{t+1} = x_{u}^{t} + \epsilon \phi_U \left( x_{u}^{t}, z(t), \rho \left( \{ \phi_M(x_{u}^{t}, x_{v}^{t}, e_{uv}) \}_{v \in N_u(t)} \right) \right)
\]

Can again be solved by forward Euler

\[
\begin{aligned}
\frac{dx_u}{dt} &= \phi_U \left( x_u, z, \rho \left( \{ \phi_M(x_u, x_v, e_{vu}) \}_{v \in N_u} \right) \right) \\
x_u(0) &= x_{t_{i-1},u}
\end{aligned}
\]

Observed snapshot of \( u \) at time \( t_{i-1} \)
Neural Algorithmic Reasoning - Combining algorithms and neural networks

- Reusable across tasks
- Executing on noisy conditions
- Sensitive to shift-of-distribution
- No interpretable operations
- Requires lots of data

- Sensitive to task variation
- Input must match pre-conditions
+ Inherent generalisation
+ Interpretable
+ Theoretical guarantees

Can we get the best of both worlds?
Learning Algorithmic Reasoning on Graphs

Recursive-type DGN

Dual Algorithmic Reasoning
Use knowledge on the structure of the optimization problem

Numeroso, Bacciu, Velickovic, ICLR 2023
Example: Ford-Fulkerson, Max-Flow & Min-Cut
Example: Ford-Fulkerson, Max-Flow & Min-Cut

Learning min-cut is essential! (Max-Flow/Min-Cut theorem)
Scaling up way out of distribution

Train on 16 nodes, predict on 10 millions

Numeroso, Bacciu, Velickovic, ICLR 2023
Wrap-Up
Software

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

[PyTorch geometric] [Deep Graph Library] [PyDGN]

Our Python library for Deep Graph Networks

[github.com/diningphil/PyDGN]
Data (Benchmarks)

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

- 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 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
- New wave of works focusing on
  - Dynamic graphs
  - DGNs as dynamical systems and their physical interpretation
  - Learning and aligning with (graph) algorithms
  - Oversmoothing, oversquashing and problems of the sort
- …in other words, plenty of opportunities for thesis work!
Next Lecture

Tomorrow 15/05 h.16.00

❖ Beyond accuracy: auditing LLMs based on exams designed for humans

❖ Guest Lecture by Wagner Meira