Representation Learning using Graph Neural Networks

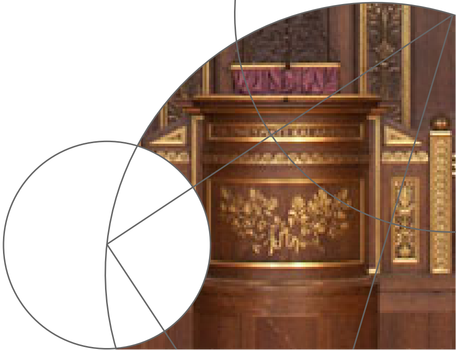
3rd Workshop on GraphNeT, Bornholm

Raghavendra Selvan

Assistant Professor
Dept. of Computer Science (ML Section)
Dept. of Neuroscience (Kiehn Lab)
Data Science Lab
University of Copenhagen

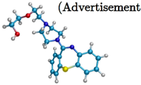
raghav@di.ku.dk
/raghavian
`https://raghavian.github.io`

May 2, 2023

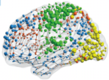


Graphs are everywhere!

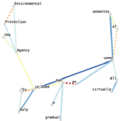
Social networks (Advertisement)



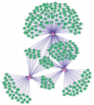
Drug/Material molecules (Chemistry)



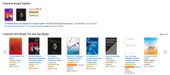
Brain connectivity (Neuroscience)



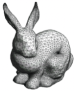
Words relationships (NLP)



Gene Regulatory Network

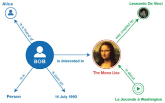


Recommender systems (Amazon, Netflix)

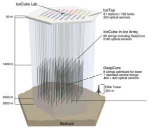


3D Meshes (Computer Graphics)

Transportation networks

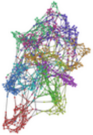


Knowledge graph (Causality)



Neutrino detection (High-energy Physics)

=

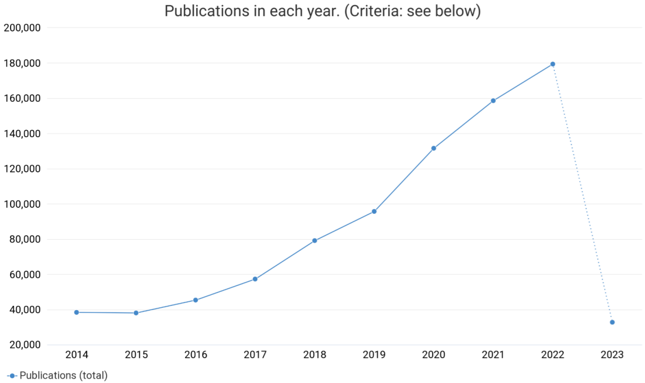


Graphs/ Networks

Heterogenous graph-structured data are non-Euclidean data

https://graphdeeplearning.github.io/project/spatial-convnets/

Graph/Geometric Deep Learning is also going everywhere!

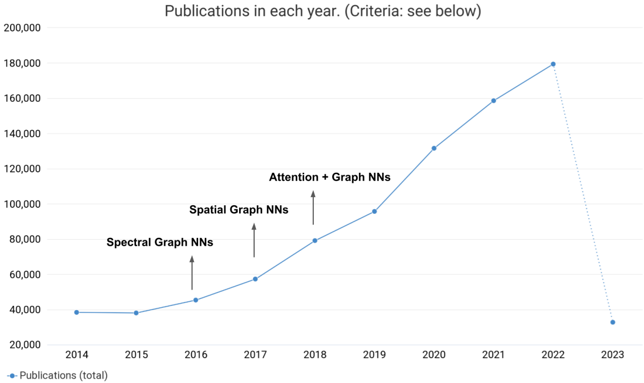


Source: https://app.dimensions.ai
Exported: March 15, 2023
Criteria: 'graph neural networks' in full data.

Graph-based ML has taken off in the past 3-4 years

(From dimensions.ai)

Graph/Geometric Deep Learning is also going everywhere!



Source: https://app.dimensions.ai
Exported: March 15, 2023
Criteria: 'graph neural networks' in full data.

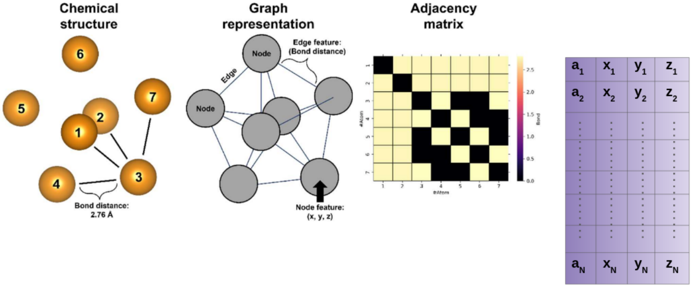
Graph-based ML has taken off in the past 3-4 years

(From dimensions.ai)

Representation Learning with GNNs: Overview

1. Motivation
   - Basics
2. Spectral & Spatial GNNs
   - ChebyNet
   - GNNs w/wo Self-Attention
3. Graph Representation Learning
   - Self-Supervised GNNs
   - Conditional Graph Generation
4. Summary

Graphs as data structures



Graph Notations

Consider a graph $\mathbf{G} = (\mathbf{X}, \mathbf{A}, \mathbf{E})$ with N nodes comprising F features per node: $\mathbf{X} \in \mathbb{R}^{N \times F}$ and adjacency matrix $\mathbf{A} \in \{0, 1\}^{N \times N}$ and edge attributes $\mathbf{E} \in \mathbb{R}^{E \times N \times N}$.

One slide introduction to Deep Learning

$$\begin{aligned} \text{Observed data} \quad & \{\mathbf{G}_1, \mathbf{G}_2, \ldots, \mathbf{G}_N\} \in \mathcal{G}, \mathbf{G}_i : \mathbf{X}_i, \mathbf{A}_i, \mathbf{E}_i \\ \text{Labels / Targets} \quad & \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\} \in \mathcal{Y} \\ \text{Decision functions/ Models} \quad & g_\theta(\cdot) : \mathcal{G} \rightarrow \mathcal{Y} \end{aligned}$$

- $g_\theta(\cdot)$ can be any DL model
- θ are the trainable parameters; $|\theta| >> N$
- Over-parameterised, non-linear function approximator
- Supervised training uses large labelled datasets
- Gradient descent based optimisation
- Multi-layered perceptrons, Convolutional/Graph neural networks, Transformers...

Representation Learning with GNNs: Overview

1. Motivation
   - Basics

2. Spectral & Spatial GNNs
   - ChebyNet
   - GNNs w/wo Self-Attention

3. Graph Representation Learning
   - Self-Supervised GNNs
   - Conditional Graph Generation

4. Summary

How to perform Convolution on Graphs?

Consider a signal operating on the nodes, $\mathbf{X} \in \mathbb{R}^N$ and a filter g_Θ parameterized by $\mathbf{\Theta} \in \mathbb{R}^N$, the graph convolution is given as:

$$g_\Theta \star \mathbf{X} = \tag{1}$$

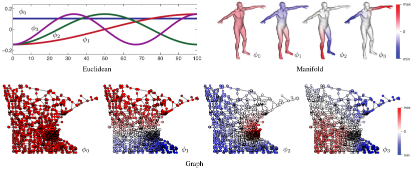
where, $\mathbf{\Phi} = (\phi_0, \phi_1 \ldots \phi_{N-1}) \in \mathbb{R}^{N\times N}$ are the orthogonal eigenfunctions forming the Fourier basis

Graph Laplacian

For a graph, $\mathbf{G}$ with adjacency matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$: the graph Laplcian is simply given as:

$$\mathbf{L} = \mathbf{D} - \mathbf{A} \tag{2}$$

$\mathbf{D} \in \mathbb{R}^{N\times N}$ is the degree matrix with $D_{ii} = \sum_j A_{ij}$



with eigenvalues in the diagonal matrix $\mathbf{\Lambda} = \text{diag}(\lambda_0, \lambda_1, \ldots, \lambda_{N-1})$

Figure from Geometric deep learning: Going beyond Euclidean data. Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, Pierre Vandergheynst 2016

Approximating Convolution on graphs with Chebyshev polynomials

Why the approximation?

Eigen decomposition of Graph Laplacian is expensive! Scales poorly with number of nodes

Recursive approximation with Chebyshev coefficients yields the popular spectral graph convolution method: **ChebyNet**[1]

$$g_\Theta \star \mathbf{X} = (\mathbf{\Phi} g_\Theta)(\mathbf{\Phi}^T \mathbf{X}) \approx \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\mathbf{L}})\mathbf{X} \tag{3}$$

with $\tilde{\mathbf{L}} = \frac{2}{\lambda_{\text{max}}}\mathbf{L} - \mathbf{I}_N$ and Chebyshev coefficients θ_k.

[1] Defferrard, Michaël, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. Advances in neural information processing systems 29 (2016): 3844-3852.

Overview of ChebyNet: CNN for Graphs (but in spectral domain)

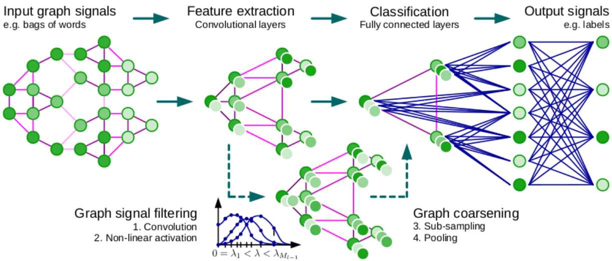


Figure 1: Architecture of a CNN on graphs and the four ingredients of a (graph) convolutional layer.

Defferrard et al. 2016

Graph Representation Learning with GNNs: Overview

1. Motivation
   - Basics
2. Spectral & Spatial GNNs
   - ChebyNet
   - GNNs w/wo Self-Attention
3. Graph Representation Learning
   - Self-Supervised GNNs
   - Conditional Graph Generation
4. Summary

Further approximation of ChebyNet yields the class of Spatial GNNs

Starting with Chebyshev approximation:

$$g_\Theta \star \mathbf{X} \approx \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\mathbf{L}})\mathbf{X} \tag{4}$$

With $K = 2, \lambda_{max} = 2$, the graph convolution operation becomes:

$$g_\Theta \star \mathbf{X} \approx \theta_0 \mathbf{X} + \theta_1 \tilde{\mathbf{L}}\mathbf{X} \tag{5}$$

$$g_\Theta \star \mathbf{X} \approx \theta(\tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{X}) \tag{6}$$

with $\theta_0 = -\theta_1$, $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$. More generally, for input $\mathbf{X} \in \mathbb{R}^{N \times F}$ and weight matrix $\mathbf{\Theta} \in \mathbb{R}^{F \times L}$:

$$\mathbf{H} = \tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{X}\mathbf{\Theta} \tag{7}$$

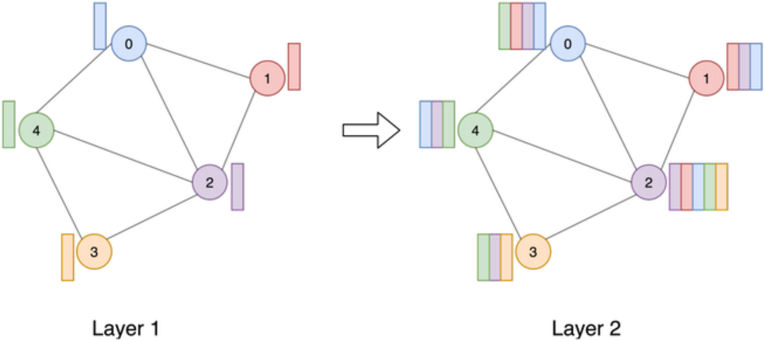
Stacking multiple of these layers with non-linearities yields the class of node GNNs[2]!

[2] Kipf, Thomas N., and Max Welling. "Semi-supervised classification with graph convolutional networks." arXiv preprint arXiv:1609.02907 (2016).

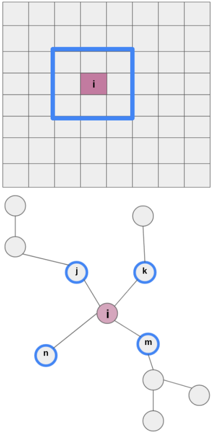
Note: Chebyshev polynomial $T_k(y)$ of order k is given by the recurrence: $T_k(y) = 2yT_{k-1}(y) - T_{k-2}(y)$ with $T_0 = 1, T_1 = y$

GNNs update states of nodes based on neighbourhood

Graphically, $\mathbf{H} = \tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}\mathbf{X}\mathbf{\Theta}$ means:



GNNs when seen from a node's point of view



- For a node $i \in \mathcal{V}$ with neighbours $\mathcal{N}_i$ the GNN operation in layer-ℓ is given as:

$$\mathbf{h}_i^{(\ell)} = \sigma\left(\frac{\Theta_\ell}{|\mathcal{N}_i|}\sum_{j\in\mathcal{N}_i}\mathbf{h}_i^{\ell-1}\right) \tag{8}$$

  with $\mathbf{h}_i^{(0)} = \mathbf{x}_i$ and Θ_ℓ are trainable parameters.
- Aggregation of transformed neighbouring node features
- M-layered GNN provides information from M-hops away!
- A form of learnable message passing!

Graphical notation for Graph Convolution Network

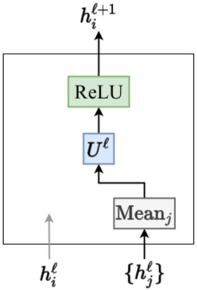


Figure 6: GCN Layer

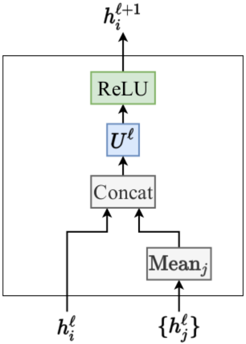
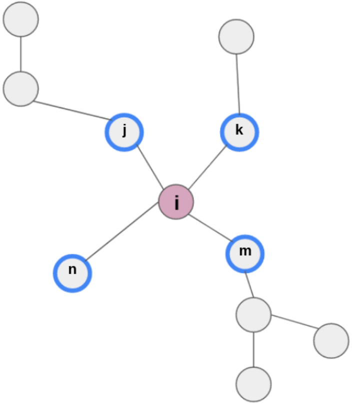


Figure 7: GraphSage Layer

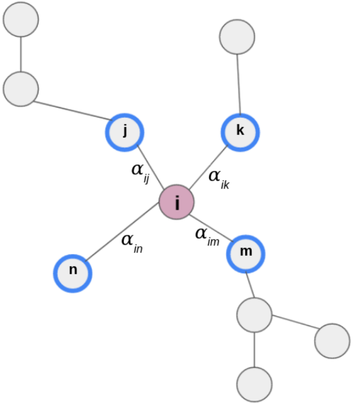
Dwivedi et al. 2022 Benchmarking Graph neural networks

Graph Attention Network pays differential attention to neighbours



Petar Veličković et al. 2018, Graph Attention Networks

Graph Attention Network pays differential attention to neighbours



Petar Veličković et al. 2018, Graph Attention Networks

Graph Attention Networks

$$\mathbf{h}_i^{(\ell)} = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \Theta_\ell \mathbf{h}_i^{\ell-1}\right) \tag{9}$$

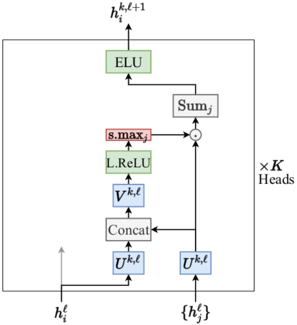


Figure 8: GAT Layer

Representation Learning with GNNs: Overview

1. Motivation
   - Basics
2. Spectral & Spatial GNNs
   - ChebyNet
   - GNNs w/wo Self-Attention
3. Graph Representation Learning
   - Self-Supervised GNNs
   - Conditional Graph Generation
4. Summary

Representation Learning is Compact Feature Learning

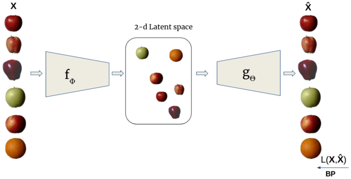
Formalizing representation learning:

> *“Learning representations of the data that make it easier to extract useful information when building classifiers or other predictors.” [1]*

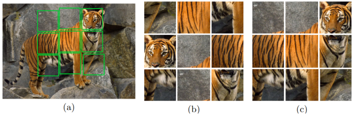
$$\mathbf{h} = g_{\phi}(\mathbf{z}) \tag{10}$$

$$\mathbf{z} = f_{\theta}(\mathbf{x}) \in \mathbb{R}^{F} \tag{11}$$

h are usually psuedo-labels, **z** can then be used in any downstream task to predict, **y**.



Undercomplete Autoencoder


(a)
(b)
(c)

Also, interesting for graphs:

- Access meaningful vector representations of graphs
- Many applications: Graph similarity, graph matching, graph generation

[1]Representation Learning: A Review and New Perspectives. Yoshua Bengio, Aaron Courville, Pascal Vincent. 2014
Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles. Mehdi Noroozi, Paolo Favaro. 2017

Self-Supervised GNNs

- Pseudo-labels derived graph data
- Labels can be derived from: $\mathbf{X} \in \mathbb{R}^{N \times F}, \mathbf{A} \in [0,1]^{N \times N}, \mathbf{E} \in \mathbb{R}^{E \times N \times N}$
  - Masking node features: $\mathbf{h} = \mathbf{M} \cdot \mathbf{X}, \mathbf{M} \in \{0,1\}^{N \times F}$
  - Noisy edges: $\mathbf{h} = 0.5 \cdot \mathbf{A} + \mathbf{N}, \mathbf{N} \in [-0.5, 0.5]^{N \times N}$
- Reconstruction with undercomplete autoencoders

Self-Supervised GNNs

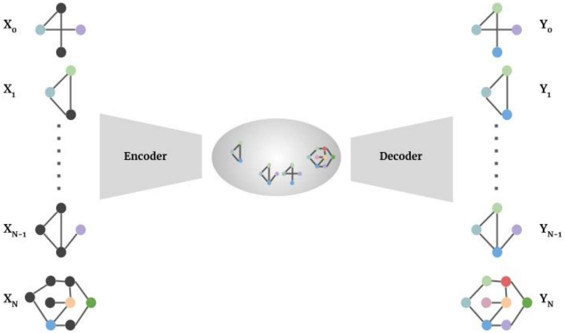
- Pseudo-labels derived graph data
- Labels can be derived from: $\mathbf{X} \in \mathbb{R}^{N \times F}, \mathbf{A} \in [0,1]^{N \times N}, \mathbf{E} \in \mathbb{R}^{E \times N \times N}$
  - Masking node features: $\mathbf{h} = \mathbf{M} \cdot \mathbf{X}, \mathbf{M} \in \{0,1\}^{N \times F}$
  - Noisy edges: $\mathbf{h} = 0.5 \cdot \mathbf{A} + \mathbf{N}, \mathbf{N} \in [-0.5, 0.5]^{N \times N}$
- Reconstruction with undercomplete autoencoders

Self-Supervised GNNs

- Pseudo-labels derived graph data
- Labels can be derived from: $\mathbf{X} \in \mathbb{R}^{N\times F}, \mathbf{A} \in [0,1]^{N\times N}, \mathbf{E} \in \mathbb{R}^{E\times N\times N}$
  - Masking node features: $\mathbf{h} = \mathbf{M} \cdot \mathbf{X}, \mathbf{M} \in \{0,1\}^{N\times F}$
  - Noisy edges: $\mathbf{h} = 0.5 \cdot \mathbf{A} + \mathbf{N}, \mathbf{N} \in [-0.5, 0.5]^{N\times N}$
- Reconstruction with undercomplete autoencoders

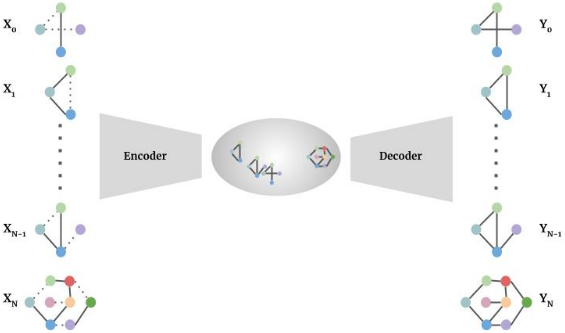
Self-Supervised GNNs

- Pseudo-labels derived graph data
- Labels can be derived from: $\mathbf{X} \in \mathbb{R}^{N \times F}, \mathbf{A} \in [0,1]^{N \times N}, \mathbf{E} \in \mathbb{R}^{E \times N \times N}$
  - Masking node features: $\mathbf{h} = \mathbf{M} \cdot \mathbf{X}, \mathbf{M} \in \{0,1\}^{N \times F}$
  - Noisy edges: $\mathbf{h} = 0.5 \cdot \mathbf{A} + \mathbf{N}, \mathbf{N} \in [-0.5, 0.5]^{N \times N}$
- Reconstruction with undercomplete autoencoders

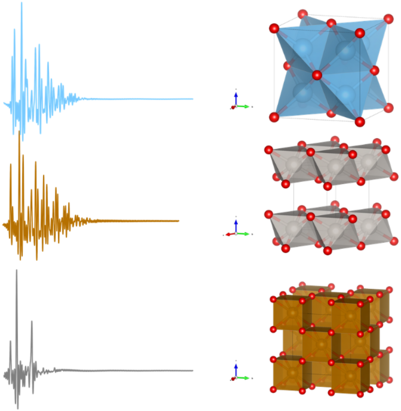


Self-Supervised GNNs

- Pseudo-labels derived graph data
- Labels can be derived from: $\mathbf{X} \in \mathbb{R}^{N\times F}, \mathbf{A} \in [0,1]^{N\times N}, \mathbf{E} \in \mathbb{R}^{E\times N\times N}$
  - Masking node features: $\mathbf{h} = \mathbf{M} \cdot \mathbf{X}, \mathbf{M} \in \{0,1\}^{N\times F}$
  - Noisy edges: $\mathbf{h} = 0.5{\cdot}\mathbf{A}{+}\mathbf{N}, \mathbf{N} \in [-0.5, 0.5]^{N\times N}$
- Reconstruction with undercomplete autoencoders

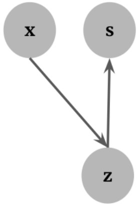


GNNs for Characterising Atomic Structure of Mono-Metallic Nanoparticles



- Solve structures starting from X-ray scattering measurements
- Reconstruct structures of nanoparticles, $\mathbf{s} \in \mathcal{S}$, from their corresponding property (PDFs), $\mathbf{x} \in \mathcal{X}$.
- Learning task: $f(\cdot) : \mathcal{X} \rightarrow \mathcal{S}$.
- From a density esimtation point of view:$p(\mathbf{s}|\mathbf{x})$.
- Many-to-one mapping

Latent Generative Model



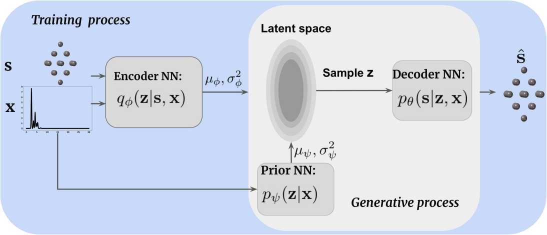
conditional-VAE

- Treat structures as a graphs with atoms as nodes, distances as edge attributes
- Using property-structure pairs formulate a conditional generative model
- Conditional Variational Autoencoder (CVAE)
- CVAE extends VAE framework to include conditioning input
- CVAE objective minimizes KLD between $p(\mathbf{z}|\mathbf{s},\mathbf{x})$ and its variational approximation $q_\phi(\mathbf{z}|\mathbf{s},\mathbf{x})$ resulting in an objective of the form:

$$\mathcal{L}_{\text{CVAE}} = \mathcal{L}_{sup} + \mathcal{L}_{reg} \tag{12}$$

$$= -\mathbb{E}_{q_\phi}\Big[\log p_\theta(\mathbf{s}|\mathbf{z},\mathbf{x})\Big] + \text{KL}\Big[q_\phi(\mathbf{z}|\mathbf{s},\mathbf{x})||p_\psi(\mathbf{z}|\mathbf{x})\Big] \tag{13}$$

High level overview of the conditional generative model



Training process

$\mathbf{s}$

$\mathbf{x}$

Encoder NN: $q_\phi(\mathbf{z}|\mathbf{s},\mathbf{x})$

μ_ϕ, σ^2_ϕ

Latent space

Sample **z**

Decoder NN: $p_\theta(\mathbf{s}|\mathbf{z},\mathbf{x})$

$\hat{\mathbf{s}}$

μ_ψ, σ^2_ψ

Prior NN: $p_\psi(\mathbf{z}|\mathbf{x})$

Generative process

- Conditioning input at Encoder/Prior networks using MLPs
- Encoder only during training
- Inference using Prior network alone
- Use GNNs in the encoder $q_\phi(\cdot)$:

$$\mathbf{H}^{(m)} = \sigma(\mathbf{H}^{(m-1)}, \mathbf{A}; \Theta_{m-1}) \quad (14)$$

with $\mathbf{H}^{(0)} = \mathbf{X} \in \mathbb{R}^{N \times F}$ and $\mathbf{H}^{(M)} = \mathbf{Z} \in \mathbb{R}^{N \times L}$

- Trained entirely with simulated data

Anker.. Selvan et al. 2021

Joint Representation Space of property + structures in 2D latent space

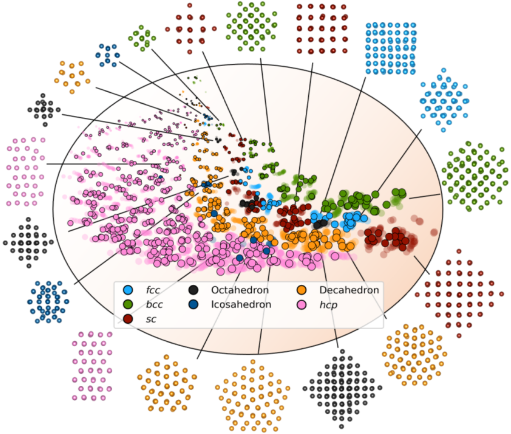


Fig. 2 | The two-dimensional latent space with structure reconstructions. The points in the latent space

Anker .. Selvan et al. 2023, DeepStruc

Meaningful interpolation in latent space

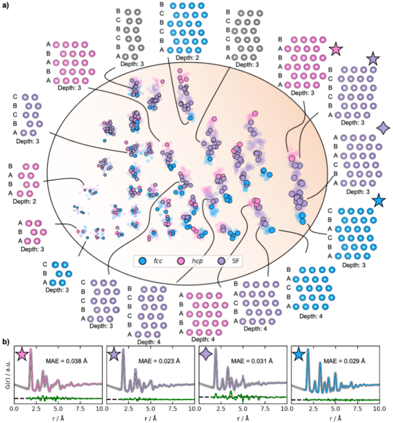


Fig. 6 | Latent space and reconstructions of stacking faulted nanoparticles. a) The latent space and reconstructed structures shown with their stacking sequence. The structures are shown in two dimensions, and

Anker .. Selvan et al. 2023, DeepStruc

Results on structure prediction of nanoparticles based on properties

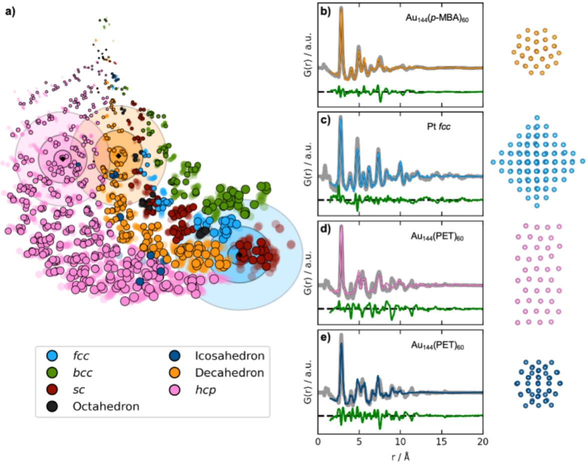
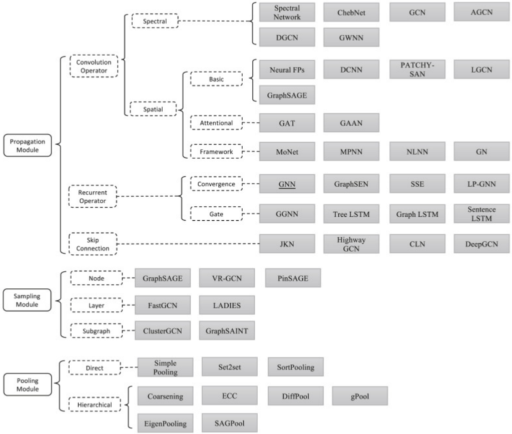


Fig. 5 | Fitting experimental PDFs with structures obtained by DeepStruc. a) The DeepStruc latent space

Representation Learning with GNNs: Overview

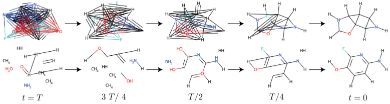
1. Motivation
   - Basics
2. Spectral & Spatial GNNs
   - ChebyNet
   - GNNs w/wo Self-Attention
3. Graph Representation Learning
   - Self-Supervised GNNs
   - Conditional Graph Generation
4. Summary

Other variations of GNN



- Propagation Module
  - Convolution Operator
    - Spectral: Spectral Network, ChebNet, GCN, AGCN, DGCN, GWNN
    - Spatial
      - Basic: Neural FPs, DCNN, PATCHY-SAN, LGCN, GraphSAGE
      - Attentional: GAT, GAAN
      - Framework: MoNet, MPNN, NLNN, GN
  - Recurrent Operator
    - Convergence: GNN, GraphSEN, SSE, LP-GNN
    - Gate: GGNN, Tree LSTM, Graph LSTM, Sentence LSTM
  - Skip Connection: JKN, Highway GCN, CLN, DeepGCN
- Sampling Module
  - Node: GraphSAGE, VR-GCN, PinSAGE
  - Layer: FastGCN, LADIES
  - Subgraph: ClusterGCN, GraphSAINT
- Pooling Module
  - Direct: Simple Pooling, Set2set, SortPooling
  - Hierarchical: Coarsening, ECC, DiffPool, gPool, EigenPooling, SAGPool

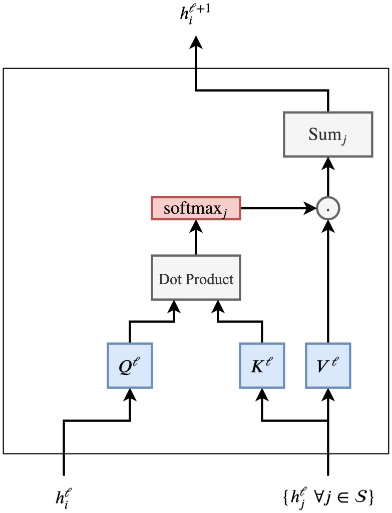
- Generalized message passing schemes
- Inductive (graph inputs) vs Transductive (single large graph)
- GNNs for Knowledge graphs, meshes
- Causal inference with GNNs
- Scaling GNNs to $\mathcal{O}(M)$ nodes
- Diffusion models



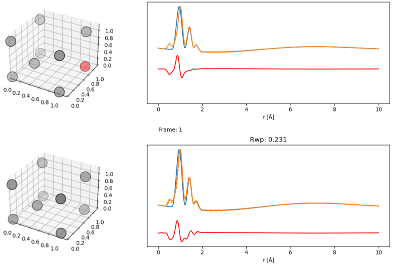
Zhou et al. 2020 Graph neural networks: A review of methods and applications
Vignac et al. 2023 DiGress: Discrete Denoising diffusion for graph generation

Summary

- Convolutions on graphs can be approximated in spectral domain
- ChebyNet uses a polynomial of spectral filter approximated with Chebyshev polynomials
- First order approximation to ChebyNet yields spatial graph convolutions
- Weighting nodes in a neighbourhood differently yields attention-type models
- Transformers can be viewed as fully-connected GATs
- Plenty of variations of GNNs by now
- Compact feature learning as Representation Learning



Thanks!



Carbontracker: Tracking and Predicting the Carbon Footprint of Training Deep Learning Models

Lasse F. Wolff Anthony* [1] Benjamin Kanding* [1] Raghavendra Selvan [1]

```
pip install carbontracker
```

Questions?

raghav@di.ku.dk
/raghavian

Funding

European Union's HorizonEurope research and innovation programme undergrant agreements No. 101070284 and No. 101070408

References

- Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. The graph neural network model. IEEE Trans. Neural Networks, 20(1):61–80, 2009.
- Defferrard, M., Bresson, X., and Vandergheynst, P. Con- volutional neural networks on graphs with fast localized spectral filtering. In Advances in Neural Information Processing Systems (NIPS), 2016.
- Kipf, T. N. and Welling, M. Semi-supervised classifica- tion with graph convolutional networks. In International Conference on Learning Representations (ICLR), 2017.
- Monti, F., Boscaini, D., and Masci, J. Geometric deep learning on graphs and manifolds using mixture model CNNs. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
- Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In International Conference on Learning Representations (ICLR), 2018.
- Selvan, R., Kipf, T., Welling, M., Juarez, A. G. U., Pedersen, J. H., Petersen, J., de Bruijne, M. (2020). Graph refinement based airway extraction using mean-field networks and graph neural networks. Medical Image Analysis, 64, 101751.
- Dwivedi, V. P., Joshi, C. K., Laurent, T., Bengio, Y., Bresson, X. (2020). Benchmarking graph neural networks. arXiv preprint arXiv:2003.00982.