Graph Representation Learning via Graph Neural Networks

Johannes Lutzeyer

LIX Seminar Series

October 20, 2022

Johannes Lutzeyer

johanneslutzeyer.com

Background:

2022 - Present	Assistant Professor in DaSciM
2020 - 2022	Postdoctoral Researcher in DaSciM
	Supervisor: Prof. Michalis Vazirgiannis
2015 - 2019	PhD in Statistics
	Supervisor: Prof. Andrew Walden

Imperial College London

Teaching:

• INF554 Machine and Deep Learning	Responsable: Michalis Vazirgiannis
• INF573 Image Analysis	Responsable: Mathieu Brédif
• Advanced Learning for Text and Graph Data	Responsable: Michalis Vazirgiannis
• CSE204: Machine Learning	Responsable: Jesse Read

Today I present work that was done in collaboration with:

Dr. George Dasoulas
Postdoctoral Researcher
Harvard University

Michalis Chatzianastasis
PhD Student LIX

Dr. Guillaume Salha Galvan
Research Scientist Deezer

Dr. Romain Hennequin
Lead Research Scientist
Deezer

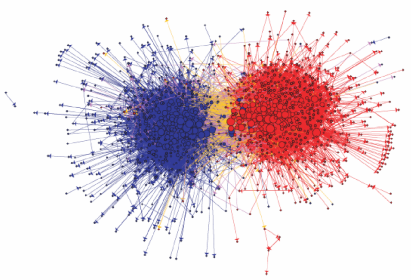
Prof. Michalis Vazirgiannis
Distinguished Professor LIX

Graph Representation Learning

Overall Goal: Learn "informative" representations of graph structured data

What is graph structured data?
It's the combination of

- a graph $G = (V, E)$;
- node-features $X = [x_1, \ldots, x_n]^T$.



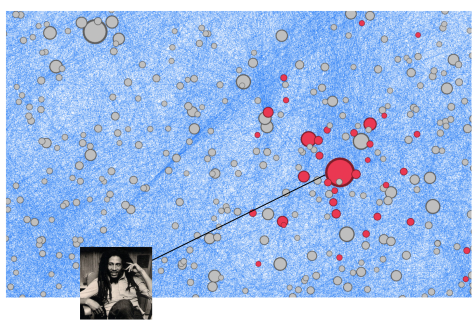
US political weblogs

Where does it arise?
It's ubiquitous!

Caffeine molecule

What can we learn from it?

- Node and Graph Classification
- Node and Graph Regression
- Graph Learning
- Link Prediction



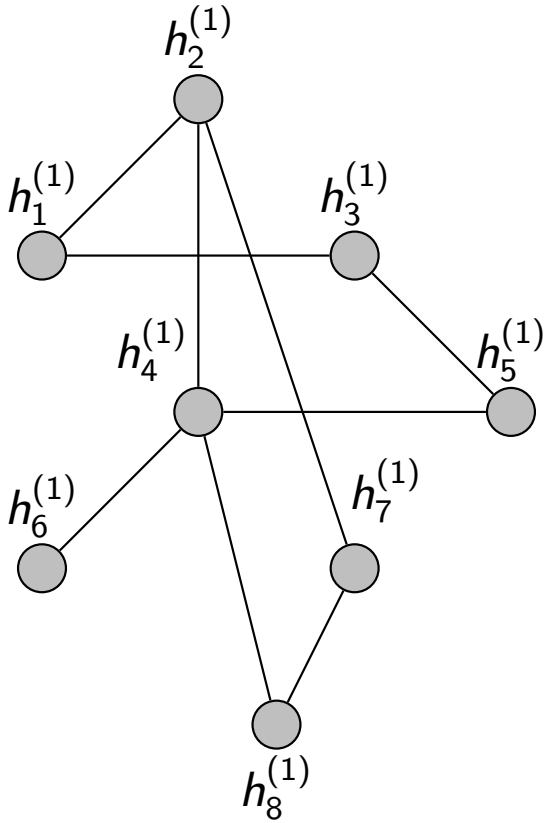
Deezer artists

Graph Neural Networks

Graph Neural Networks (GNNs) are neural networks that take graph-structured data as input.

In this talk we will only see a specific type of GNN, the Message Passing Neural Networks.

$$m_v^{(k)} = M^{(k)}\left(\left\{h_w^{(k-1)} : w \in \mathcal{N}(v)\right\}\right),$$
$$h_v^{(k)} = U^{(k)}\left(h_v^{(k-1)}, m_v^{(k)}\right).$$



E.g., the Graph Convolutional Network (GCN, Kipf and Welling, 2017)

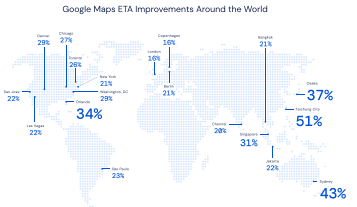
$$H^{(1)} = \mathrm{ReLU}\left(\tilde{A}XW^{(1)}\right).$$

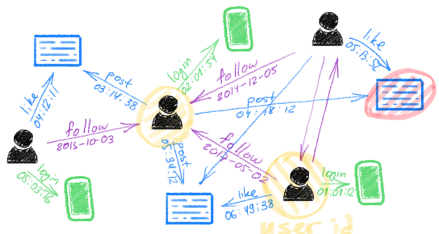
Other examples of popular GNN architectures are the GIN (Xu et al., 2019), GraphSage (Hamilton et al., 2017) and GAT (Veličković et al., 2018).

Academic and Industrial Success of GNNs

Empirical and Theoretical **Research**:

- expressivity analysis of GNNs (Xu et al., 2019; Morris et al., 2019; Geerts and Reutter, 2022);
- robustness to adversarial attacks and noise (Günnemann, 2022; Sun et al., 2020; Zhou et al., 2020).
- bottlenecks, e.g., oversmoothing and oversquashing (Alon and Yahav, 2020; Deac et al., 2022)





Successful **Applications** of GNNs:

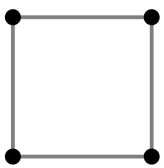
- Google Maps (Lange and Perez, 2020);
- Twitter (Bronstein, 2020);
- Amazon (Virinchi et al., 2022);
- Discovery of a *new antibiotic* (Stokes et al., 2020).

Graph Shift Operators

Definition

Graphs $G = (V, E)$ can be represented using:

- *adjacency matrix* $A \in \{0,1\}^{n \times n}$ where $A_{ij} = 1$ iff $(i,j) \in E$.
- *unnormalised graph Laplacian matrix* $L = D - A$, where $D = \mathrm{diag}(A\mathbf{1}_n)$.
- *symmetric normalised graph Laplacian matrix* $L_{sym} = D^{-1/2} L D^{-1/2}$ and *random-walk normalised Laplacian matrix* $L_{rw} = D^{-1} L$.



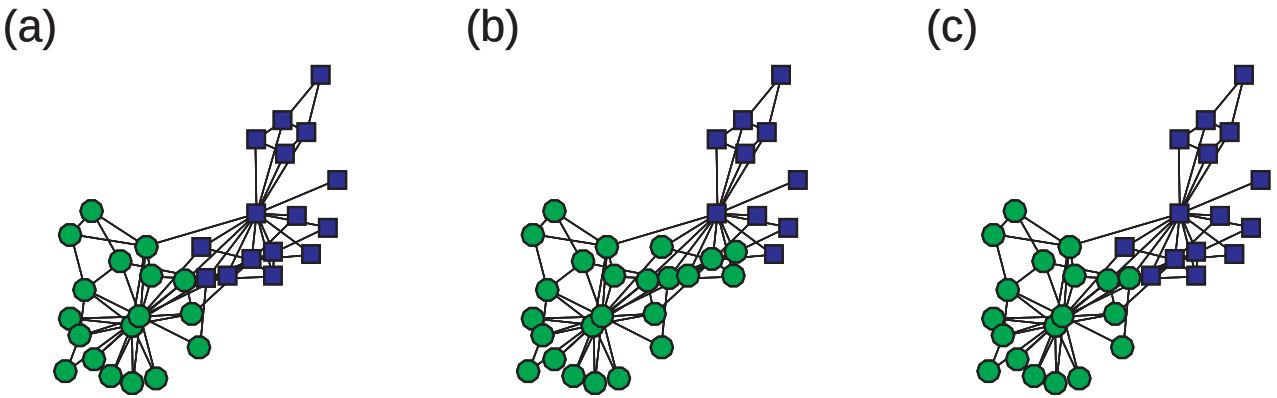
$$A = \begin{pmatrix} 0 & 1 & \mathbf{0} & 1 \\ 1 & 0 & 1 & \mathbf{0} \\ \mathbf{0} & 1 & 0 & 1 \\ 1 & \mathbf{0} & 1 & 0 \end{pmatrix} \quad L = \begin{pmatrix} 2 & -1 & \mathbf{0} & -1 \\ -1 & 2 & -1 & \mathbf{0} \\ \mathbf{0} & -1 & 2 & -1 \\ -1 & \mathbf{0} & -1 & 2 \end{pmatrix} \quad L_{sym} = \begin{pmatrix} 1 & -0.5 & \mathbf{0} & -0.5 \\ -0.5 & 1 & -0.5 & \mathbf{0} \\ \mathbf{0} & -0.5 & 1 & -0.5 \\ -0.5 & \mathbf{0} & -0.5 & 1 \end{pmatrix}$$

Definition (Sandryhaila and Moura, 2013)

A matrix $S \in \mathbb{R}^{n \times n}$ is called a *Graph Shift Operator* (GSO) if it satisfies: $S_{ij} = 0$ for $i \neq j$ and $(i,j) \notin E$.

GSOs in Graph Representation Learning

- Spectral clustering:



Spectral clustering of the karate network using A in (a), L in (b) and L_{rw} in (c) (Lutzeyer, 2020).

- Graph Neural Networks (GNNs), e.g., GCN (Kipf and Welling, 2017)

$$H^{(l+1)} = \sigma\left(D_1^{-\frac{1}{2}} A_1 D_1^{-\frac{1}{2}} H^{(l)} W^{(l)}\right). \tag{1}$$

The *sum-based aggregator* in the GIN (Xu et al., 2019) corresponds to the use of the adjacency matrix A.

In Message Passing Neural Networks, the choice of message passing function corresponds to a choice of GSO.

Overview of the Talk

Recall,

- GNN: neural networks that process graph-structured data
- GSO: matrices that represent graphs

In this talk,

1) learn optimal GSO in GNNs;
2) use of an LSTM instead of a GSO in a GNN;
3) introduce global cluster information to GSO in Graph Autoencoders.

1) Learning Parametrised Graph Shift Operators

Dasoulas, Lutzeyer & Vazirgiannis (2021, ICLR)

Motivation to Learn GSOs

- When introducing the different standard GSO choices Butler and Chung (2017) state: *"No one matrix is best because each matrix has its own limitations in that there is some property which the matrix cannot always determine"*.
- Graph signal processing literature: the GSO choice involves "different tradeoffs" and leads to different signal models (Deri and Moura, 2017; Ortega et al., 2018). Therefore, they recommend using *whichever GSO works best* in a particular analysis or learning task.

Research Questions

Q1: Is there a *single optimal* representation to encode graph structures?
Q2: Can we learn optimal representations in an *efficient* way?

Parametrised Graph Shift Operators

Definition

We define the *Parametrised Graph Shift Operator (PGSO)*, denoted by $\gamma(A, \mathcal{S})$, as

$$\gamma(A, \mathcal{S}) = m_1 D_a^{e_1} + m_2 D_a^{e_2} A_a D_a^{e_3} + m_3 I_n, \tag{2}$$

where $A_a = A + aI_n$, $D_a = \text{Diag}(A_a \mathbf{1}_n)$ and $\mathcal{S} = (m_1, m_2, m_3, e_1, e_2, e_3, a)$.

$\mathcal{S} = (m_1, m_2, m_3, e_1, e_2, e_3, a)$	Operator	Description
$(0, 1, 0, 0, 0, 0, 0)$	A	Adjacency matrix and Summation Aggregation Operator of GNNs
$(1, -1, 0, 1, 0, 0, 0)$	$D - A$	Unnormalised Laplacian matrix L
$(1, 1, 0, 1, 0, 0, 0)$	$D + A$	Signless Laplacian matrix Q (Cvetkovic et al., 1997)
$(0, -1, 1, 0, -1, 0, 0)$	$I_n - D^{-1}A$	Random-walk Normalised Laplacian L_{rw}
$(0, -1, 1, 0, -\frac{1}{2}, -\frac{1}{2}, 0)$	$I_n - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$	Symmetric Normalised Laplacian L_{sym}
$(0, 1, 0, 0, -\frac{1}{2}, -\frac{1}{2}, 1)$	$D_1^{-\frac{1}{2}} A_1 D_1^{-\frac{1}{2}}$	Normalised Adjacency matrix of GCNs (Kipf and Welling, 2017)
$(0, 1, 0, 0, -1, 0, 0)$	$D^{-1}A$	Mean Aggregation Operator of GNNs (Xu et al., 2019)

PGSO in Graph Neural Networks

Notation:

- $\phi(A) : [0,1]^{n \times n} \to \mathbb{R}^{n \times n}$ denotes a non-parametrised function of A.
- $\mathcal{M}(\phi(A), X)$ denotes a GNN model.

Utilization of PGSO in GNNs

1. **GNN-PGSO**: $\mathcal{M}(\phi(A), X) \to \mathcal{M}'(\gamma(A, \mathcal{S}), X)$.
2. **GNN-mPGSO (multi-PGSO)**: $\mathcal{M}(\phi(A), X) \to \mathcal{M}''(\gamma^{[K]}(A, \mathcal{S}^{[K]}), X)$, where $\gamma^{[K]}(A, \mathcal{S}^{[K]}) = [\gamma(A, \mathcal{S}^1), \ldots, \gamma(A, \mathcal{S}^K)]$.

- Put simply, we **replace** the GSO used in a GNN model by $\gamma(A, \mathcal{S})$.

Convolutions and Message-Passing

- Examples of utilisation of GNN-PGSO models

1. **GCN (Kipf & Welling, 2017)**: The propagation rule is

$$H^{(l+1)} = \sigma\left(D_1^{-\frac{1}{2}} A_1 D_1^{-\frac{1}{2}} H^{(l)} W^{(l)}\right),$$

where $W^{(l)}$ is a weight matrix and σ is a non-linear activation function. The GCN-PGSO and GCN-mPGSO models are defined, respectively, as

$$H^{(l+1)} = \sigma\left(\gamma(A, \mathcal{S}) H^{(l)} W^{(l)}\right) \text{ and } H^{(l+1)} = \sigma\left(\gamma(A, \mathcal{S}^l) H^{(l)} W^{(l)}\right).$$

2. **GIN (Xu et al., 2019)**: The propagation rule is

$$h_i^{(l+1)} = \sigma\left(h_i^{(l)} W^{(l)} + \sum_{j:v_j \in \mathcal{N}(v_i)} h_j^{(l)} W^{(l)}\right).$$

The GIN-PGSO model is defined as

$$h_i^{(l+1)} = \sigma\left(\left(m_1 \left(D_a\right)_i^{e_1} + m_3\right) h_i^{(l)} W^{(l)} + \sum_{j:v_j \in \mathcal{N}(v_i)} \epsilon_{ij} h_j^{(l)} W^{(l)}\right),$$

where ϵ_{ij} are edge weights defined as $\epsilon_{ij} = m_2 \left(D_a\right)_i^{e_2} \left(D_a\right)_j^{e_3}$.

Spectral Analysis

Theorem

$\gamma(A, \mathcal{S})$ has real eigenvalues and a set of real eigenvectors independent of the parameters chosen in $\mathcal{S}$.

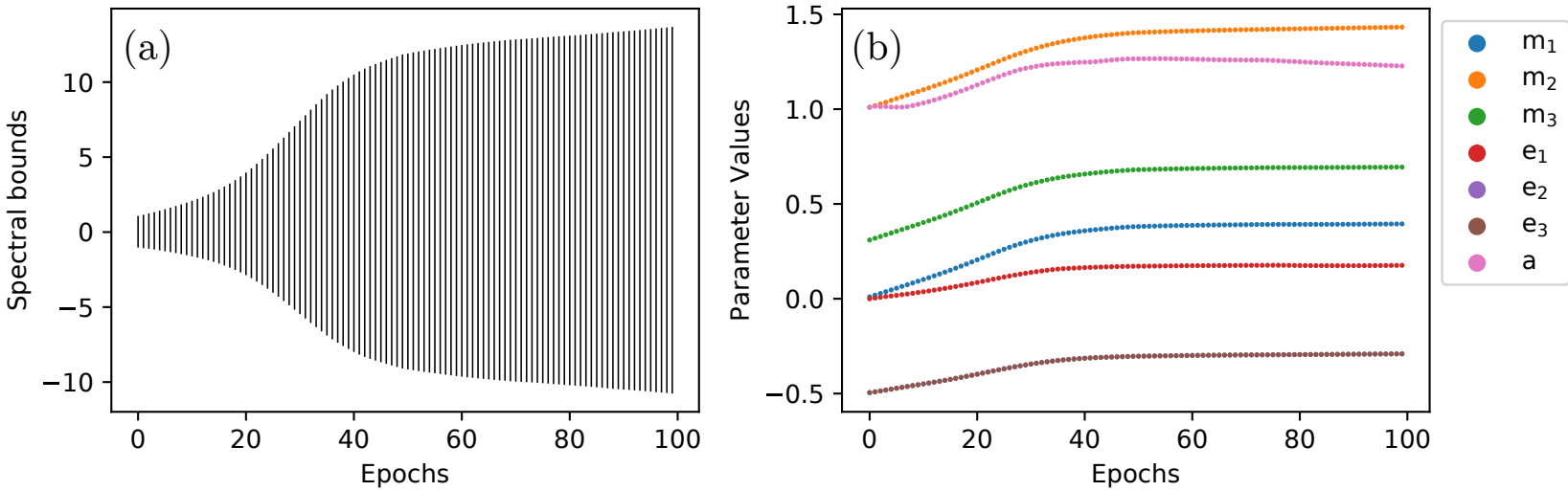
Theorem

Let $C_i = m_1(d_i + a)^{e_1} + m_2(d_i + a)^{e_2+e_3} a + m_3$ and $R_i = |m_2|(d_i + a)^{e_2+e_3} d_i$, where d_i denotes the degree of node v_i. Furthermore, we denote eigenvalues of $\gamma(A, \mathcal{S})$ by $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$. Then, for all $j \in \{1, \ldots, n\}$,

$$\lambda_j \in \left[\min_{i \in \{1,\ldots,n\}} (C_i - R_i), \max_{i \in \{1,\ldots,n\}} (C_i + R_i) \right]. \tag{3}$$

- For the parametrisation of $\gamma(A, \mathcal{S})$ corresponding to the adjacency matrix, we obtain the spectral support $[-d_{\max}, d_{\max}]$, as required.
- For the message passing operator in the GCN, we obtain the following bounds on the spectral support $[-(d_{\max} - 1)/(d_{\max} + 1), 1]$, the lower bound of this interval tends to -1 as $d_{\max} \to \infty$.

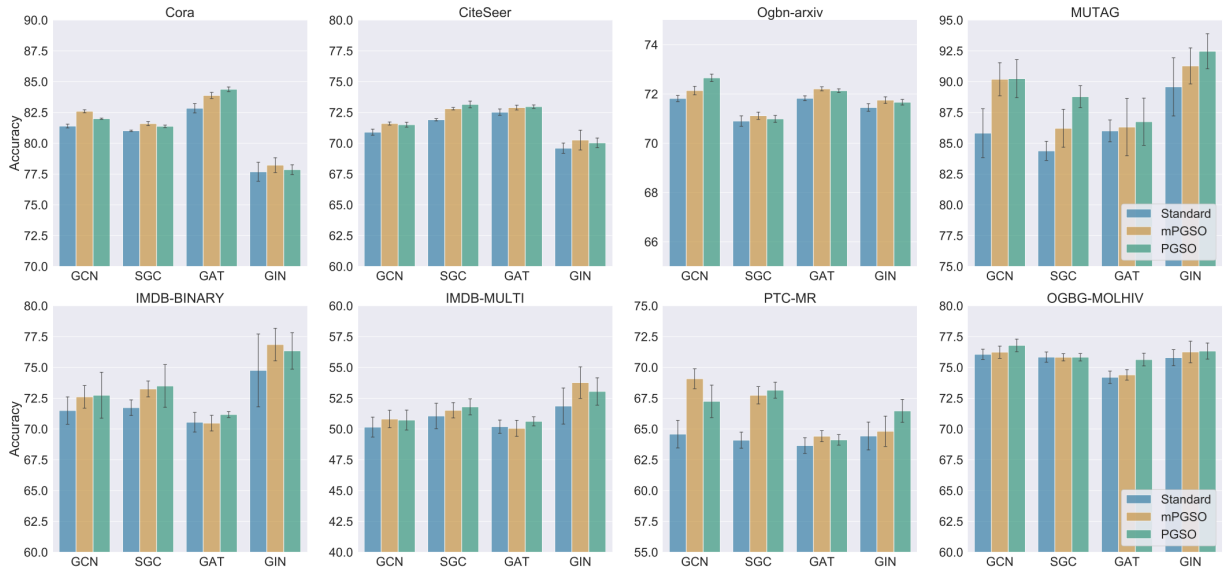
Spectral Analysis: Empirical Observation



- Surprisingly, the spectral support of the PGSO remains centered at 0 throughout training.
- We observe the parameters of the PGSO to be smoothly varying throughout training.

Results on Real-world Datasets

- In **Simulation Studies** we independently verify theoretical results:
  - PGSO parameters replicate the GSO regularisation derived in Qin and Rohe (2013).
- **Real-World Datasets**:
  - 3 node classification and 5 graph classification datasets.
  - 4 GNN architectures: GCN, SGC, GAT and GIN.
  - 3 GSO variants: **Standard**, **mPGSO** and **PGSO**.



- For all datasets and architectures, the incorporation of the PGSO and the mPGSO **enhances** the model performance.
- The impact of PGSO is **higher** in graph classification tasks.
- Our code is publicly available: https://github.com/gdasoulas/PGSO.

2) Graph Ordering Attention Networks

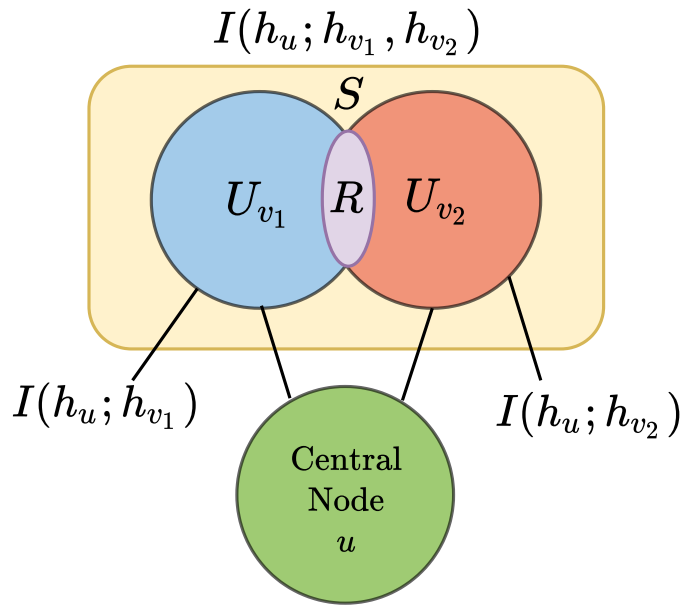
Chatzianastasis, Lutzeyer, Dasoulas & Vazirgiannis (2023, AAAI)

An Information Theory Perspective for Graphs

Following Williams and Beer (2010), the mutual information of the hidden state of a node u and the hidden states in its closed neighbourhood $\boldsymbol{H}_{\overline{\mathcal{N}}(u)}$ can be decomposed into three components:

$$I(h_u; \boldsymbol{H}_{\overline{\mathcal{N}}(u)}) = \sum_{v \in \overline{\mathcal{N}}(u)} U_v + R + S,$$

- unique information U_v,
- redundant information R,
- synergistic information S.



E.g.: The Cora Dataset (Sen et al., 2008)

- U_v – unique key words,
- R – repeatedly present key words,
- S – combinations of key words.

Problem

Most GNN layers **only capture unique information**, since they consider nodes individually or in pairs.

Graph Ordering Attention Networks

To capture these three types of information, we introduce a dependence of the contribution c_{uv} of the neighbour node v to the central node u on all neighbours of u.

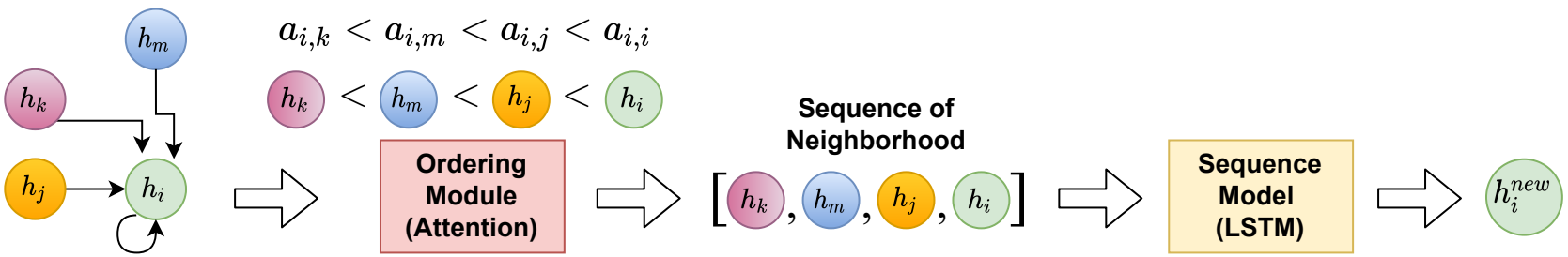


Figure: An illustration of the aggregation and update of the representation of node v_i using a GOAT layer.

1) A self-attention mechanism is used to obtain a ranking between the nodes of the neighbourhood.
2) Then, the ordered neighbourhood is given as input into a sequence model (LSTM) to produce the updated representation of node v_i.

GOAT – Theoretical Results

Permutation-Equivariance of GOAT

The GOAT layer performs a *permutation-equivariant transformation* of the hidden states.

Sketch Proof: We perform a permutation-invariant operation on each local neighborhood resulting in a permutation-equivariant architecture.

Injectivity of GOAT

The GOAT layer is able to approximate any measurable *injective* function arbitrarily well in probability.

Sketch Proof: The reordering in the Ordering Part is straightforwardly injective and for the Sequence Modeling Part we make use of Theorem 3 from Hammer (2000, p. 6), which establishes that recurrent neural networks can approximate any measurable function (including injective functions) arbitrarily well in probability.

Results on Real-World Datasets

- Good performance of GOAT on **Simulation Studies** for which synergistic information is crucial.
- **Real-World Datasets:**
  - 6 real-world node classification datasets
  - 3 variants using different RNNs as the sequence model of GOAT

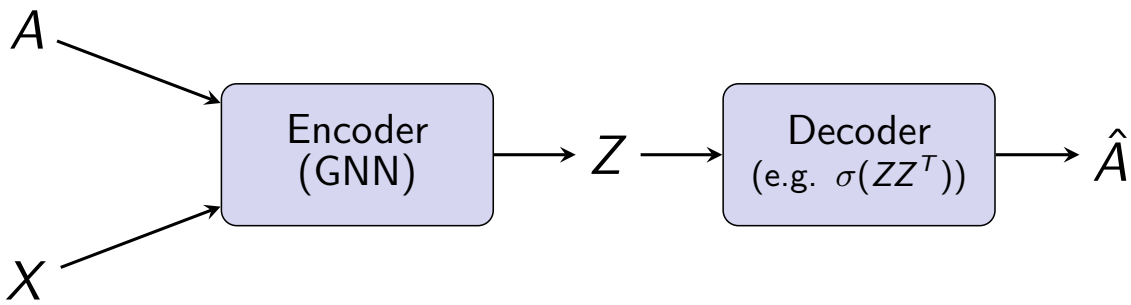
Method	Cora	Citeseer	Disease	LastFM Asia	Computers	Photo
MLP	43.8	52.9	79.10 ±0.97	72.27 ±1.00	79.53 ±0.66	87.89 ±1.04
GCN	81.4	67.5	88.98 ±2.21	83.58 ±0.93	90.72 ±0.50	93.99 ±0.42
GraphSAGE (mean)	77.2	65.3	88.79 ±1.95	83.07 ±1.19	91.47 ±0.37	94.32 ±0.46
GraphSAGE (lstm)	74.1	59.9	90.50 ±2.15	**86.85** ±1.07	91.26 ±0.51	94.32 ±0.64
GIN	75.5	62.1	90.20 ±2.23	82.94 ±1.25	84.68 ±2.33	90.07 ±1.19
GAT	83.0	69.3	89.13 ±2.22	77.57 ±1.82	85.41 ±2.95	90.30 ±1.76
PNA	76.4	58.9	86.84 ±1.89	83.24 ±1.10	90.80 ±0.51	94.35 ±0.68
GOAT(lstm)	83.2	68.9	**92.11** ±1.88	83.29 ±0.91	91.34 ±0.50	**94.38** ±0.66
GOAT(gru)	83.5	**70.0**	91.97 ±1.90	83.35 ±0.91	**91.54** ±0.48	94.22 ±0.58
GOAT(rnn)	**84.2**	67.9	91.67 ±1.69	83.21 ±0.98	89.10 ±0.51	92.45 ±0.60

- Our code is publicly available: https://github.com/MichailChatzianastasis/GOAT

3) Modularity-Aware Graph Autoencoders for Joint Community Detection and Link Prediction

Salha-Galvan, Lutzeyer, Dasoulas, Hennequin & Vazirgiannis (2022, Neural Networks Journal)

Graph Autoencoders (GAEs)



Graph Autoencoders (GAEs):

- **learn** low-dimensional representations Z in an unsupervised manner.
- typically **consist of** the composition of a GNN with an inner product decoder reconstructing the graph structure.
- currently find **industrial use in recommendation systems**.

Motivation of our Project

Research Problem

Graph Autoencoders are very good at link prediction and often underwhelming in community detection. Learning node embeddings Z that enable good performance in both tasks is desirable for real-world applications.

Our Modularity-Aware Graph Autoencoders address this problem by

- using a different GSO in the encoder's message passing scheme.
- modifying the loss function to consider a softened version of the modularity.
- considering both the clustering's modularity and the classification AUC in the hyperparameter selection.

To stay on topic, we will discuss only the first of these contributions.

Introducing Global Cluster Information to the Message Passing Step

Steps of our method modifying the encoder (GNN):

1) We run the Louvain algorithm to cluster the graph.
   - Automatically determines the number of communities.
   - It's fast $O(n \log n)$.
   - It maximises the modularity, which complements our other contributions.
2) We define a graph with adjacency A_c composed of fully connected components corresponding to the communities obtained in step 1).
3) We replace A in the GNN encoder by

$$A + \lambda A_c,$$

   where $\lambda \geq 0$ is a scalar hyperparameter determining the importance of the cluster information.
4) We randomly sample s neighbours for each node in its fully connected component to define a subgraph represented by A_s.
5) We replace A in the GNN encoder by

$$A + \lambda A_s,$$

   where $\lambda \geq 0$ is a scalar hyperparameter determining the importance of the cluster information.

Experiments: Cora Graph without Node Features

A weakness of existing models addressing this problem:
Their performance heavily decreases if no node features are available.

Models (Dimension $d = 16$)	**Joint Link Prediction and Community Detection on graph with 15% of edges being masked**			
	AMI (in %)	**ARI (in %)**	**AUC (in %)**	**AP (in %)**
Modularity-Aware GAE/VGAE Models				
Linear Modularity-Aware VGAE	*42.86 ± 1.65*	*34.53 ± 1.97*	*85.96 ± 1.24*	*87.21 ± 1.39*
Linear Modularity-Aware GAE	**43.48 ± 1.12**	**35.51 ± 1.20**	**87.18 ± 1.05**	*88.53 ± 1.33*
GCN-based Modularity-Aware VGAE	41.03 ± 1.55	*33.43 ± 2.17*	84.87 ± 1.14	85.16 ± 1.23
GCN-based Modularity-Aware GAE	41.13 ± 1.35	*35.01 ± 1.58*	*86.90 ± 1.16*	*87.55 ± 1.26*
Standard GAE/VGAE Models				
Linear VGAE	32.22 ± 1.76	21.82 ± 1.80	85.69 ± 1.17	**89.12 ± 0.82**
Linear GAE	28.41 ± 1.68	19.45 ± 1.75	84.46 ± 1.64	*88.42 ± 1.07*
GCN-based VGAE	28.62 ± 2.76	19.70 ± 3.71	85.47 ± 1.18	*88.90 ± 1.11*
GCN-based GAE	31.30 ± 2.07	19.89 ± 3.07	85.31 ± 1.35	*88.67 ± 1.24*
Other Baselines				
Louvain	39.09 ± 0.73	20.19 ± 1.73	–	–
VGAECD	33.54 ± 1.46	24.32 ± 2.25	83.12 ± 1.11	84.68 ± 0.98
VGAECD-OPT	34.41 ± 1.62	24.66 ± 1.98	82.89 ± 1.20	83.70 ± 1.16
ARGVA	28.96 ± 2.64	19.74 ± 3.02	85.85 ± 0.87	*88.94 ± 0.72*
ARGA	31.61 ± 2.05	20.18 ± 2.92	85.95 ± 0.85	*89.07 ± 0.70*
DVGAE	30.46 ± 4.12	21.06 ± 5.06	85.58 ± 1.31	*88.77 ± 1.29*
DeepWalk	30.26 ± 2.32	20.24 ± 3.91	80.67 ± 1.50	80.48 ± 1.28
node2vec	36.25 ± 1.38	29.43 ± 2.21	82.43 ± 1.23	81.60 ± 0.91

In the absence of node features, our model outperforms a large number of baselines achieving good performance in both tasks.

Experiments: Real-World Datasets with Node Features

Datasets	Models (Dimension $d = 16$)	Joint Link Prediction and Community Detection on graph with 15% of edges being masked			
		AMI (in %)	**ARI (in %)**	**AUC (in %)**	**AP (in %)**
Cora	Linear Modularity-Aware VGAE	**49.70 ± 2.04**	**43.64 ± 3.51**	**93.10 ± 0.88**	**94.06 ± 0.75**
	Linear Standard VGAE	46.90 ± 1.43	38.24 ± 3.56	*93.04 ± 0.80*	*94.04 ± 0.75*
	Louvain	39.09 ± 0.73	20.19 ± 1.73	–	–
	Best other baseline:				
	VGAECD-OPT	47.83 ± 1.64	39.45 ± 3.53	*92.25 ± 1.07*	92.60 ± 0.91
Citeseer	Linear Modularity-Aware VGAE	*22.21 ± 1.24*	**12.59 ± 1.25**	86.54 ± 1.20	88.07 ± 1.22
	Linear Standard VGAE	17.38 ± 1.43	6.10 ± 1.51	**89.08 ± 1.19**	**91.19 ± 0.98**
	Louvain	**22.71 ± 0.47**	7.70 ± 0.67	–	–
	Best other baseline:				
	DVGAE	16.02 ± 3.32	*10.03 ± 4.48*	86.85 ± 1.48	88.43 ± 1.23
Deezer-Album	GCN-Based Modularity-Aware VGAE	**19.10 ± 0.21**	**12.00 ± 0.17**	**85.40 ± 0.14**	*86.38 ± 0.15*
	GCN-Based Standard VGAE	13.98 ± 0.35	8.81 ± 0.32	*85.37 ± 0.12*	**86.41 ± 0.11**
	Louvain	17.68 ± 0.20	11.02 ± 0.13	–	–
	Best other baseline:				
	node2vec	18.34 ± 0.29	11.27 ± 0.28	83.51 ± 0.17	84.12 ± 0.15

The good performance of our model extends to industrial scale datasets such as a private Deezer graph containing 2.5 million music albums and 25 million edges.

Other Topics We Have Been or Are Working On

- Analysing the Robustness of GNNs to Structural Noise
  (Seddik et al., 2022, AISTATS)
- Sparsifying Weight Matrices in GNNs
  (Lutzeyer et al., 2022, ICLR Workshop)
- Graph Representation Learning to Detect Product Influence
  (collaborator: LVMH & SEPHORA)
- Quantifying Over-Smoothing in GNNs
- Improving GNNs at Scale: Approximate PageRank and CoreRank
  (under review: NeurIPS Workshop)
- Antibiotic Resistance Prediction Using GNNs
  (under review: NeurIPS Workshop)

Conclusions

- Graph Representation Learning is a highly active area of research at the moment gaining both academic and industrial interest.
- Graph Neural Networks are a versatile and powerful tool, that you may want to consider using.

Specifically, with regards to the presented projects

- Learning optimal graph representation – via a Parametrised GSO – in GNNs improves their performance on real world datasets.
- The Partial Information Decomposition offers a novel view of graph learning and the GOAT architecture effectively addresses the identified challenges.
- Adding global cluster information to the GSO in GNNs improves the performance of Graph Autoencoders in industrial application.

Thank you for your attention!

@JLutzeyer

References

U. Alon & E. Yahav, "On the Bottleneck of Graph Neural Networks and its Practical Implications," In: *International Conference on Learning Representations (ICLR)*, 2020.

M. Bronstein, "Graph ML at Twitter," *Twitter Engineering Blog Post*, `https://blog.twitter.com/engineering/en_us/topics/insights/2020/graph-ml-at-twitter`, 2020.

S. Butler & F. Chung, "Spectral graph theory," In: L. Hogben (ed) *Handbook of linear algebra (2nd edition)*, Boca Raton, FL: CRC Press, pp. 47/1—47/14, 2017.

M. Chatzianastasis, J. F. Lutzeyer, G. Dasoulas & M. Vazirgiannis, "Graph Ordering Attention Networks," *Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI),* 2023.

D. Cvetkovic, R. Rowlinson & S. Simic, *Eigenspaces of graphs*, Cambridge, UK: Cambridge University Press, 1997.

L. Dall'Amico, R. Couillet & N. Tremblay, "Optimal Laplacian regularization for sparse spectral community detection," *ICASSP*, 2020.

A. Deac, M. Lackenby & P. Veličković, "Expander Graph Propagation," *arXiv:2210.02997*, 2022.

G. Dasoulas, J. F. Lutzeyer & M. Vazirgiannis, "Learning Parametrised Graph Shift Operators," In: *International Conference on Learning Representations (ICLR)*, 2021.

J. A. Deri & J. M. F. Moura, "Spectral projector-based graph Fourier transforms," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, pp. 785–795, 2017.

V.P. Dwivedi, C.K. Joshi, T. Laurent, Y. Bengio & X. Bresson, "Benchmarking Graph Neural Networks," *arXiv:2003.00982*, 2020.

F. Geerts & J. L. Reutter, "Expressiveness and Approximation Properties of Graph Neural Networks," *International Conference on Learning Representations (ICLR)*, 2022.

J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals & G. E. Dahl, "Neural message passing for Quantum chemistry," *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

S. Günnemann, "Graph Neural Networks: Adversarial Robustness," *Graph Neural Networks: Foundations, Frontiers, and Applications*, pp. 149–176, 2022.

B. Hammer, "On the approximation capability of recurrent neural networks," *Neurocomputing*, pp. 107–123, 2000.

W. L. Hamilton, R. Ying & J. Leskovec, "Inductive Representatino Learning on Large Graphs," *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS),* pp. 1025 – 1035, 2017.

Thomas N. Kipf & M. Welling, "Variational Graph Auto-Encoders" *NeurIPS Workshop on Bayesian Deep Learning*, 2016.

Thomas N. Kipf & M. Welling, "Semi-supervised classification with graph convolutional networks," *International Conference on Learning Representations (ICLR)*, 2017.

O. Lange & L. Perez, "Traffic prediction with advanced Graph Neural Networks," *DeepMind Research Blog Post*, `https://deepmind.com/blog/article/traffic-prediction-with-advanced-graph-neural-networks`, 2020.

J. Lutzeyer, *Network Representation Matrices and their Eigenproperties: A Comparative Study,* PhD thesis: Imperial College London, 2020.

J. Lutzeyer, C. Wu & M. Vazirgiannis, "Graph Neural Network Simplification: Sparsifying the Update Step," *ICLR Workshop on Geometrical and Topological Representation Learning*, 2022.

C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J.E Lenssen, G. Rattan & M. Grohe, "Weisfeiler and Lehman Go Neural: Higher-order Graph Neural Networks," *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 4602–4609, 2019.

A. Ortega, P. Frossard, J. Kovacevic, J. M. F. Moura & P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, pp. 808–828, 2018.

T. Qin & K. Rohe, "Regularized Spectral Clustering under the Degree-Corrected Stochastic Blockmodel," *Advances in neural information processing systems (NIPS)*, pp. 3120–3128, 2013.

G. Salha-Galvan, J. F. Lutzeyer, G. Dasoulas, R. Hennequin & M. Vazirgiannis, "Modularity-Aware Graph Autoencoders for Joint Community Detection and Link Prediction," *arxiv:2202.00961*, 2022.

A. Sandryhaila & J. M. F. Moura "Discrete signal processing on graphs," *IEEE Transactions on Signal Processing*, vol. 61, pp. 1644–1656, 2013.

M. E. A. Seddik, C. Wu, J. F. Lutzeyer & M. Vazirgiannis, "Node Feature Kernels Increase Graph Convolutional Network Robustness," *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2022.

P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher & T. Eliassi-Rad, *AI Magazine*, p. 93, 2008.

J. M. Stokes, K. Yang, K. Swanson, W. Jin, A. Cubillos-Ruiz, N. M. Donghia, C. R. MacNair, S. French, L. A. Carfrae, Z. Bloom-Ackermann, V. M. Tran, A. Chiappino-Pepe, A. H. Badran, I. W. Andrews, E. J. Chory, G. M. Church, E. D. Brown, T. S. Jaakkola, R. Barzilay & J. J. Collins, "A Deep Learning Approach to Antibiotic Discovery," *Cell*, pp. 688–702, 2020.

L. Sun, Y. Dou, C. Yang, J. Wang, P. S. Yu & B. Li, "Adversarial attack and defense on graph data: A survey," *arXiv:1812.10528*, 2020.

P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò & Y. Bengio, "Graph Attention Networks," *6th International Conference on Learning Representations (ICLR),* 2018.

S. Virinchi, A. Saladi & A. Mondal, "Recommending Related Products Using Graph Neural Networks in Directed Graphs," In: *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, 2022.

P. L. Williams & R. D. Beer, "Nonnegative decomposition of multivariate information," *arXiv:1004.2515*, 2010.

K. Xu, W. Hu, J. Leskovec & S. Jegelka."How powerful are graph neural networks?", *International Conference on Learning Representations (ICLR)*, 2019.

Y. Zhou, H. Zheng & X. Huang, "Graph Neural Networks: Taxonomy, Advances and Trends," *arXiv:2012.08752*, 2020.