Benchmarking Graph Neural Networks

Xavier Bresson

School of Computer Science and Engineering
Data Science and AI Research Centre
Nanyang Technological University (NTU), Singapore

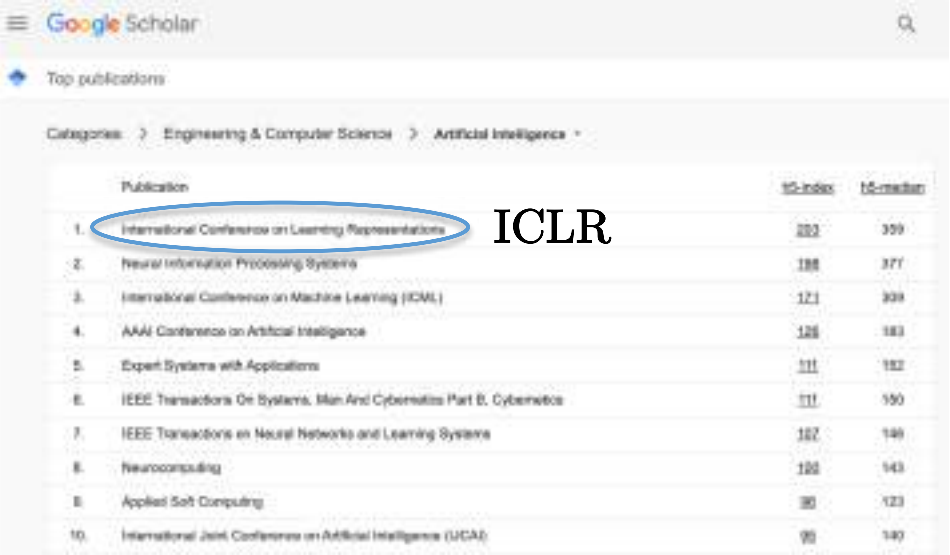
Summer School on Data Science (SSDS 2020)

Sven Loncaric, Tomislav Smuc, Vinko Zlatic, Tomislav Lipic

Sept 11th 2020

Graph Neural Networks (GNNs)

- GNNs have become the standard toolkit for analyzing and learning from data on graphs.
- Hottest machine learning topics in 2020.



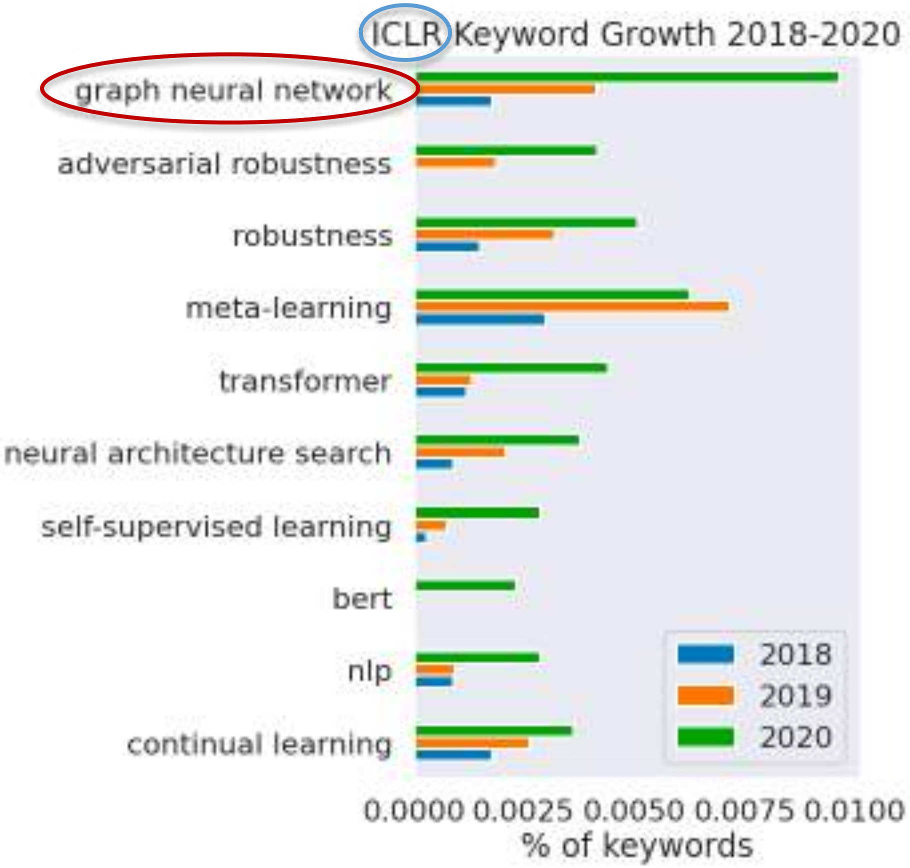
Google Scholar

Top publications

Categories > Engineering & Computer Science > Artificial Intelligence

	Publication	h5-index	h5-median
1.	International Conference on Learning Representations	203	359
2.	Neural Information Processing Systems	198	377
3.	International Conference on Machine Learning (ICML)	171	309
4.	AAAI Conference on Artificial Intelligence	126	183
5.	Expert Systems with Applications	111	182
6.	IEEE Transactions On Systems, Man And Cybernetics Part B, Cybernetics	111	150
7.	IEEE Transactions on Neural Networks and Learning Systems	107	149
8.	Neurocomputing	100	143
9.	Applied Soft Computing	98	123
10.	International Joint Conference on Artificial Intelligence (IJCAI)	95	140

ICLR



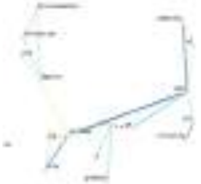
Graph Neural Networks (GNNs)

- Applicative domains :
  - Chemistry[1,2] (generate new drugs and materials)
  - Physics[3,4,5] (learn and accelerate physics)
  - Recommender systems[6,7] (leverage consumer-product choices)
  - Social sciences[8,9] (predict future collaborators, identify fake news)
  - Knowledge graphs[10,11] (reasoning with entity-relationship)
  - Neuroscience[12] (understanding brain mechanisms and neuro-degenerative diseases)
  - Computer Vision[13] (scene understanding for visual reasoning)
  - Natural Language Processing[14] (transformers)
  - Combinatorial Optimization[15,16] (better/faster approximated solutions to NP-hard problems)



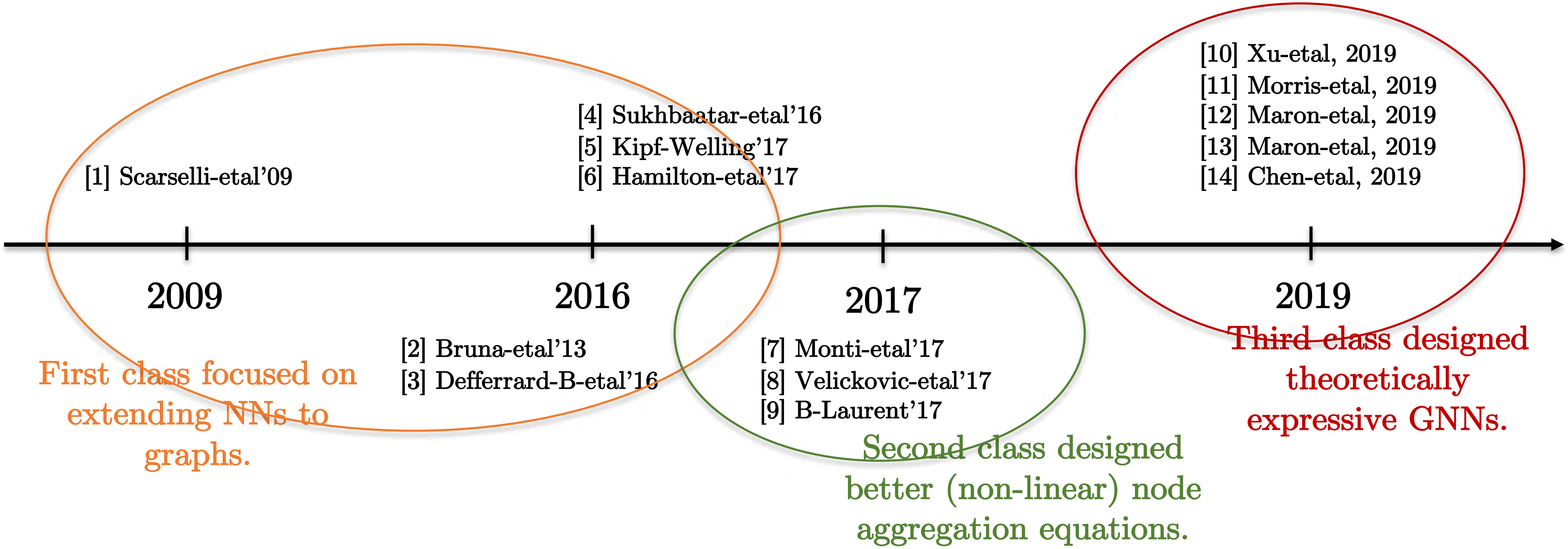






[1] Duvenaud, Maclaurin, Iparraguirre, Bombarell, Hirzel, Aspuru-Guzik, Adams, Convolutional networks on graphs for learning molecular fingerprints, 2015
[2] Gilmer, Schoenholz, Riley, Vinyals, Dahl, Neural message passing for quantum chemistry, 2017
[3] Battaglia, Pascanu, Lai, Rezende, Danilo, Interaction networks for learning about objects, relations and physics, 2016
[4] Cranmer, Xu, Battaglia, Ho, Learning symbolic physics with graph networks, 2019
[5] Sanchez-Gonzalez, Godwin, Pfaff, Ying, Leskovec, Battaglia, Learning to simulate complex physics with graph networks, 2020
[6] Monti, Bronstein, Bresson, Geometric matrix completion with recurrent multi-graph neural networks, 2017
[7] Ying, He, Chen, Eksombatchai, Hamilton, Leskovec, Graph convolutional neural networks for web-scale recommender systems, 2018
[8] Kipf, Welling, Semi-supervised classification with graph convolutional networks, 2017
[9] Monti, Frasca, Eynard, Mannion, Bronstein, Fake news detection on social media using geometric deep learning, 2019
[10] Schlichtkrull, Kipf, Bloem, VanDenBerg, Titov, Welling, Modeling relational data with graph convolutional networks, 2018
[11] Chami, Wolf, Juan Sala Ravi, Re, Low-dimensional hyperbolic knowledge graph embeddings, 2020
[12] Parisot, Ktena, Ferrante, Lee, Guerrero, Glocker, Rueckert, Disease prediction using graph networks: Application to Autism Spectrum Disorder and Alzheimer's disease, 2018
[13] Johnson, Gupta, Fei-Fei, Image generation from scene graphs, 2018
[14] Vaswani, Shazeer, Parmar, Uszkoreit, Jones, N Gomez, Kaiser, Polosukhin, Attention is all you need, 2017
[15] Vinyals, Fortunato, Jaitly, Pointer, 2015
[16] Bello, Pham, Le, Norouzi, Bengio, Neural combinatorial optimization with reinforcement learning, 2016

A Decade of GNNs



Developing powerful GNNs for real-world adoption of graph deep learning.

[1] Scarselli, Gori, Tsoi, Hagenbuchner, Monfardini, The Graph Neural Network Model, 2009
[2] Bruna, Zaremba, Szlam, LeCun, Spectral networks and locally connected networks on graphs, 2013
[3] Defferrard, Bresson, Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, 2016
[4] Sukhbaatar, Szlam, Fergus, Learning multiagent communication with backpropagation, 2016
[5] Kipf, Welling, Semi-supervised classification with graph convolutional networks, 2017
[6] Hamilton, Ying, Leskovec, Inductive representation learning on large graphs, 2017
[7] Monti, Boscaini, Masci, Rodola, Svoboda, Bronstein, Geometric deep learning on graphs using mixture model cnns, 2017
[8] Velickovic, Cucurull, Casanova, Romero, Lio, Bengio, Graph attention networks, 2017
[9] Bresson, Laurent, Residual gated graph convnets, 2017
[10] Xu, Hu, Leskovec, Jegelka, How powerful are graph neural networks?, 2019
[11] Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe, Weisfeiler and leman go neural: Higher-order graph networks, 2019
[12] Maron, Ben-Hamu, Shamir, Lipman, Invariant and equivariant graph networks, 2019
[13] Maron, Ben-Hamu, Serviansky, Lipman, Provably powerful graph networks, 2019
[14] Chen, Villar, Chen, Bruna, On the equivalence graph isomorphism testing and function approximation with gnns, 2019

Apologize for not citing more works (4000+ GNN papers).

Tracking Progress

- The field has grown extensively in the last three years ☺
- Unfortunately, evaluating the effectiveness of new architectures/ideas has become difficult for two major reasons :
  - We have been evaluated progress on small datasets such as Cora[1], Citeseer[2] and TU[3].
    - Cora is a single graph of 2.7K nodes, TU-IMDB has 1.5K graphs with 13 nodes on average, and TU-MUTAG has 188 molecules with 18 nodes.
    - Simple or graph-agnostic architectures provide statistically same performance as complex architectures[4,5].
  - We have not rigorously enforced standardized experimental settings for fair comparisons between models[6].
- It has become critical to solve these issues.

[1] McCallum, Nigam, Rennie, Seymore, Automating the construction of internet portals with machine learning, 2000
[2] Getoor, Link-based classification, 2005
[3] Kersting, Kriege, Morris, Mutzel, Neumann, Benchmark data sets for graph kernels, 2020
[4] Hoang, Maehara, Revisiting graph neural networks, 2019
[5] Chen, Bian, Sun, Are powerful graph neural nets necessary? a dissection on graph classification, 2019
[6] Errica, Podda, Bacciu, Micheli, A fair comparison of graph neural networks for graph classification, 2019

Benchmarking GNNs

- Our motivation : Identify and quantify what types of architectures, first principles are universal, generalizable, and scalable to larger and more challenging datasets.
- Benchmarking has been beneficial for driving progress, identifying essential ideas, and solving domain-specific problems[1].
  - The 2012 ImageNet challenge[2] has provided a benchmark dataset that has triggered the deep learning revolution.
- Challenges :
  - Developing a rigorous experimental setting for fair comparisons, and being reproducible,
  - Using datasets that can statistically separate model performance,
  - Benchmarking distinct fundamental graph tasks.

[1] Weber, Saelens, Cannoodt, Soneson, Hapfelmeier, Gardner, Boulesteix, Saeys, Robinson, Essential guidelines for computational method benchmarking, 2019
[2] Deng, Dong, Socher, Li, Li, Fei-Fei, ImageNet: A Large-Scale Hierarchical Image Database, 2009

Outline

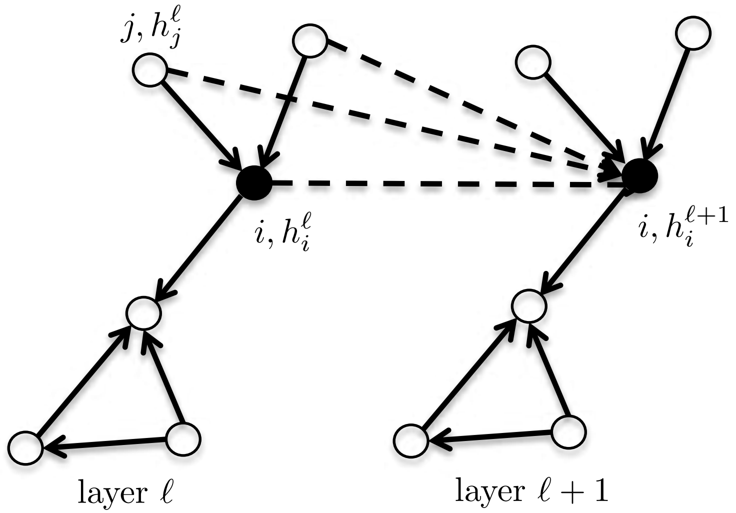
- Motivation
- Message-Passing GCNs
- Weisfeiler-Lehman GNNs
- Graph-Agnostic GNNs
- Datasets
- Infrastructure and Experimental Setting
- Benchmarking Results
- Laplacian Positional Encodings
- Link Prediction with Edge Representation
- Conclusion

Outline

- Motivation
- Message-Passing GCNs
- Weisfeiler-Lehman GNNs
- Graph-Agnostic GNNs
- Datasets
- Infrastructure and Experimental Setting
- Benchmarking Results
- Laplacian Positional Encodings
- Link Prediction with Edge Representation
- Conclusion

MP-GCNs

- Message-passing Graph Convolutional Networks
- What are the minimal inner structures to design a message-passing node aggregation function f_{GCN} ?
  - Invariant by node permutation (equivariant/invariant layers[1]).
  - Independent of graph size n and neighborhood size.
  - Locality (local reception field - only neighbors are considered).
  - Graph convolution operator (weight sharing across graph).
  - Linear complexity $\mathrm{O}(\boldsymbol{E})$, $\boldsymbol{E}$ being the number of edges (reduces to $\mathrm{O}(n)$ for sparse graphs).



$$h_i^{\ell+1} = f_{\mathrm{GCN}}\big(h_i^\ell, \{h_j^\ell : j \in \mathcal{N}_i\}\big)$$

[1] Maron, Ben-Hamu, Shamir, Lipman, Invariant and equivariant graph networks, 2019

Isotropic GCNs

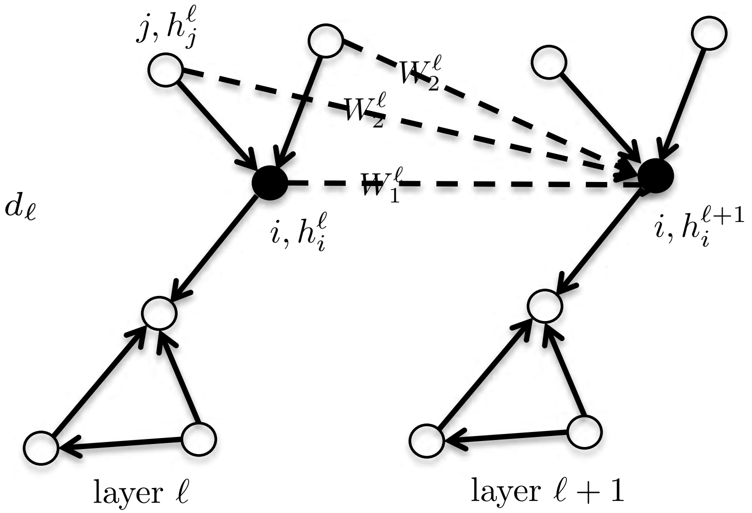
- We consider a class of isotropic GCNs when the node update equation treats every "edge direction" equally, i.e. each neighbor contributes equally to the update of the central node :

ReLU

Sum/mean /max

$$h_i^{\ell+1} = \sigma\Big(W_1^\ell \; h_i^\ell + \sum_{j \in \mathcal{N}_i} W_2^\ell h_j^\ell \Big),$$

$$h^{\ell+1} \in \mathbb{R}^{n \times d_{\ell+1}}, \;\; h^\ell \in \mathbb{R}^{n \times d_\ell}, \;\; W_{1,2}^\ell \in \mathbb{R}^{d_{\ell+1} \times d_\ell}$$



- Models :
  - Vanilla GCNs[1,2]
  - GraphSage[3]
  - ChebNets[4]

- Isotropic GCNs are limited to learn template weights for the central node h_i (weight W_1) and the neighbors h_j (weight W_2).

[1] Sukhbaatar, Szlam, Fergus, Learning multiagent communication with backpropagation, 2016
[2] Kipf, Welling, Semi-supervised classification with graph convolutional networks, 2017
[3] Hamilton, Ying, Leskovec, Inductive representation learning on large graphs, 2017
[4] Defferrard, Bresson, Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, 2016

Anisotropic GCNs

- Standard ConvNets[1] produce anisotropic filters because Euclidean grids have directional structure (up, down, left, right).
- GCNs such as vanilla GCNs, GraphSage, ChebNets compute isotropic filters as there is no notion of direction on arbitrary graphs.
- How to get anisotropy back ?
  - Natural edge features[2,3] if available (e.g. different bond connections between atoms in molecular graphs or distinct edge attributes in knowledge graphs).
  - Learn anisotropy with a mechanism invariant by index permutation and can treat neighbors differently :
    - MoNets[4] with edge degrees
    - GatedGCNs [5] with edge gates
    - GAT[6] with attention mechanism[7]

[1] LeCun Bottou Bengio, Haffner, Gradient-based learning applied to document recognition, 1998
[2] Gilmer, Schoenholz, Riley, Vinyals, Dahl, Neural message passing for quantum chemistry, 2017
[3] Bresson, Laurent, A Two-Step Graph Convolutional Decoder for Molecule Generation, 2019
[4] Monti, Boscaini, Masci, Rodolà, J. Svoboda, M. Bronstein, Geometric deep learning on graphs and manifolds using mixture model CNNs, 2016
[5] Bresson, Laurent, Residual gated graph convnets, 2017
[6] Velickovic, Cucurull, Casanova, Romero, Lio, Bengio, Graph Attention Networks, 2018
[7] Bahdanau, Cho, Bengio, Neural machine translation by jointly learning to align and translate, 2015

Anisotropic GCNs

- When the update equation treats every edge direction differently, we instantiate anisotropic GCNs with an aggregation equation of the form :

$$h_i^{\ell+1} = \sigma\Big(W_1^\ell \; h_i^\ell + \sum_{j \in \mathcal{N}_i} \eta_{ij}^\ell W_2^\ell h_j^\ell\Big), \;\; h^{\ell+1} \in \mathbb{R}^{n \times d_{\ell+1}}, \;\; h^\ell \in \mathbb{R}^{n \times d_\ell}, \;\; W_{1,2}^\ell \in \mathbb{R}^{d_{\ell+1} \times d_\ell},$$

where $\eta_{ij}^l = g_W(h_i^l, h_j^l)$ is a parameterized function which weights are learned during training.

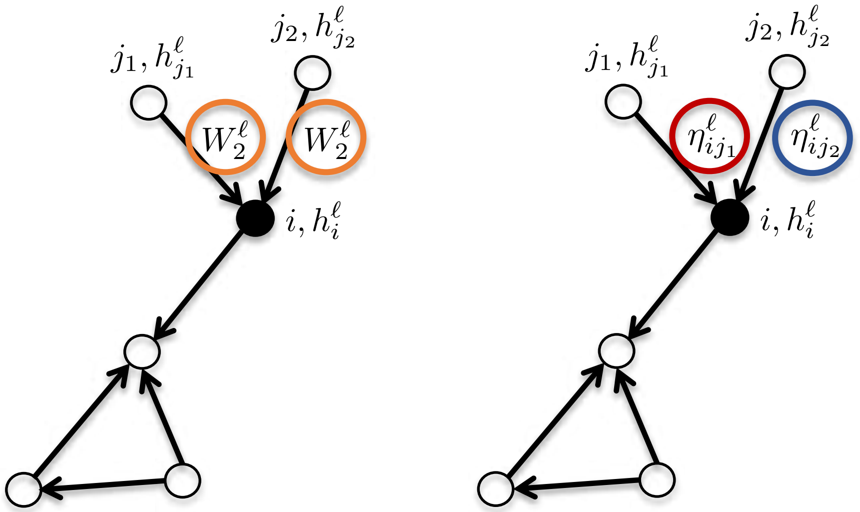
$$\begin{aligned} \eta_{ij}^\ell &= g_{W_{QK}}(h_i^\ell, h_j^\ell) \\ &= \mathrm{Softmax}_{\mathcal{N}_i}(e_{ij}^\ell) = \frac{\exp(e_{ij}^\ell)}{\sum_{j' \in \mathcal{N}_i} \exp(e_{ij'}^\ell)} \end{aligned}$$

$$\text{with } e_{ij}^\ell = (W_Q^\ell h_i^\ell)^T (W_K^\ell h_j^\ell), \;\; W_Q, W_K \in \mathbb{R}^{d_{\ell+1} \times d_\ell}$$

Attention mechanism in 1-hop neighborhood

Query

Key

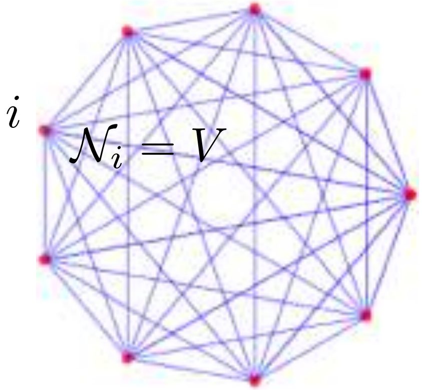


Isotropic weights

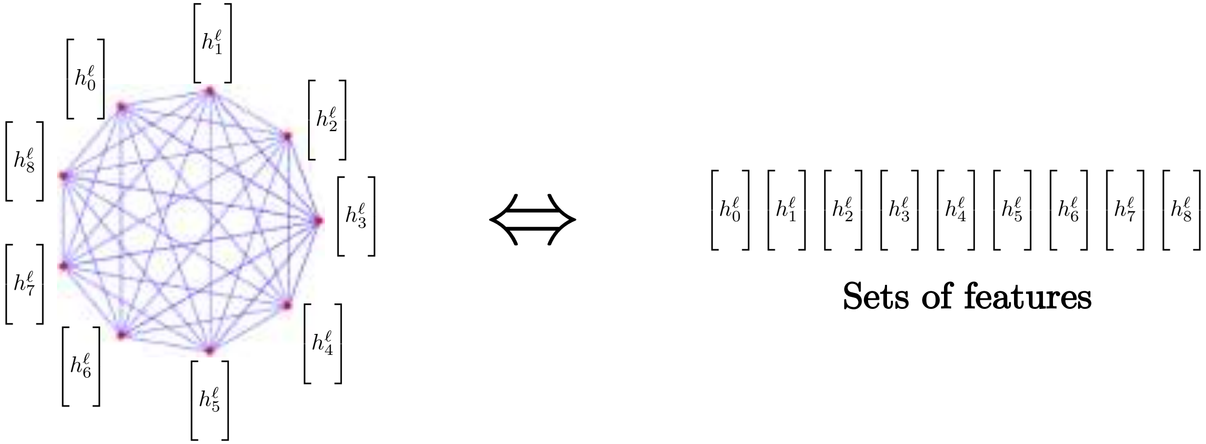
Anisotropic weights

Transformers

- Transformers is a special case of GCNs when the graph is fully connected.
  - The neighborhood $\mathcal{N}_i$ is the whole graph.
- What does it mean to have a fully connected graph ?
  - There is no particular graph structure that can be used. It becomes useless to talk about graphs (as each data point is connected to all other points).
  - It would be better to talk about sets.
  - Transformers are set neural networks, which analyze bags of features.



Fully connected graph



Sets of features

[1] A Vaswani, N Shazeer, N Parmar, J Uszkoreit, L Jones, A. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, 2017

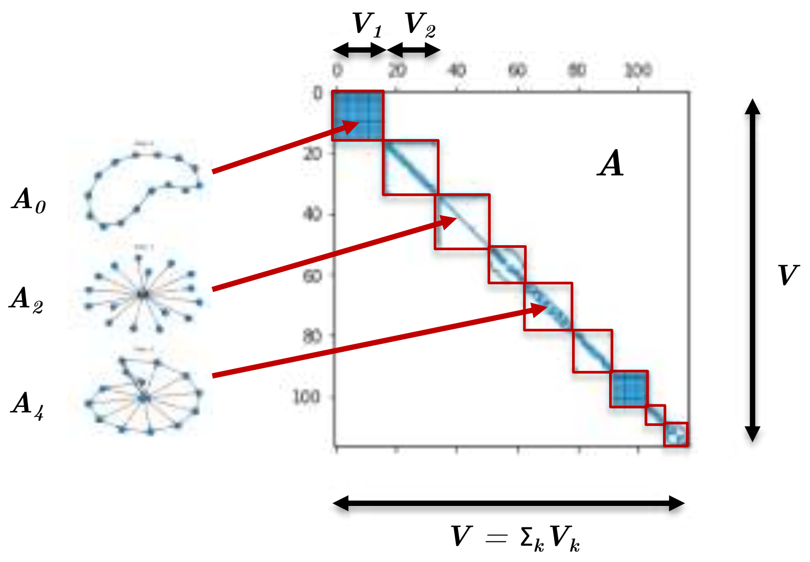
Batch Normalization and Residual Connections

- GCNs benefit from BN[1,2] and RC[1,3] :
  - Speed up learning process.
  - Improve performance with better generalization.

$$h_i^{\ell+1} = h_i^{\ell} \;+\; \sigma\Big(\mathrm{BN}\left(\hat{h}_i^{\ell+1}\right)\Big)$$

$$\hat{h}_i^{\ell+1} = f_{\mathrm{GCN}}\left(\, h_i^{\ell} \,,\; \{h_j^{\ell} : j \in \mathcal{N}_i\}\,\right)$$

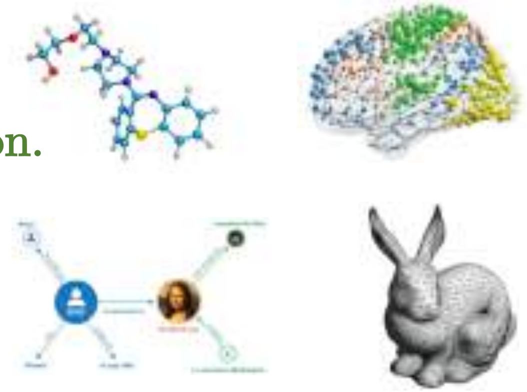
- How to batch graphs of different sizes ?
  - For images of size $\boldsymbol{B}$ x $\boldsymbol{V_x}$ x $\boldsymbol{V_y}$ x $\boldsymbol{d}$, BN is performed along the batch dimension, i.e. dim=0.
  - For graphs, a (big) sparse block diagonal matrix $\boldsymbol{A}$ of size $\boldsymbol{V}$ x $\boldsymbol{d}$ is first built from K matrices $\boldsymbol{A}_k$ and BN is carried out along the node direction, i.e. dim=0.



[1] Bresson, Laurent, Residual gated graph convnets, 2017
[2] Ioffe, Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015
[3] Li, Muller, Qian, Delgadillo, Abualshour, Thabet, Ghanem, Deepgcns: Making gcns go as deep as cnns, 2019

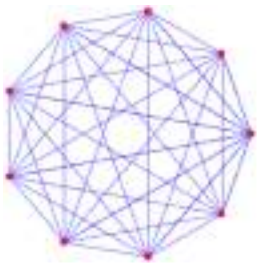
Sparsity and Local Computations

- GCNs leverage graph sparsity :
  - Sparsity is a good inductive bias for generalization.
  - Sparse vs full graphs.



Sparse graphs

vs.



Fully connected graph

- Local computations :
  - Node update equation is local, only depending on neighborhood $\mathcal{N}_i$ of node i, and independent of graph size n, making the space/time complexity $O(n)$ for sparse graphs.
  - GCNs are highly parallelizable on GPUs, and sparse matrix multiplications are efficiently implemented via GNN libraries s.a. DGL[1], PyG[2], Spektral[3].

$$h_i^{\ell+1} = f_{\mathrm{GCN}}\big(h_i^\ell, \{h_j^\ell : j \in \mathcal{N}_i\}\big)$$

DGL PyTorch geometric Spektral

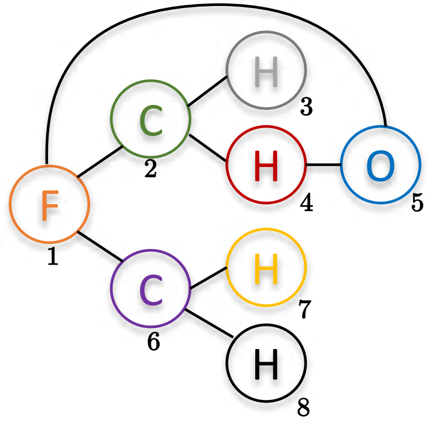
[1] Wang-etal, Deep graph library: Towards efficient and scalable deep learning on graphs, 2019
[2] Fey, Lenssen, Fast graph representation learning with pytorch geometric, 2019
[3] Grattarola, Alippi, Graph Neural Networks in TensorFlow and Keras with Spektral, 2020

Outline

- Motivation
- Message-Passing GCNs
- Weisfeiler-Lehman GNNs
- Graph-Agnostic GNNs
- Datasets
- Infrastructure and Experimental Setting
- Benchmarking Results
- Laplacian Positional Encodings
- Link Prediction with Edge Representation
- Conclusion

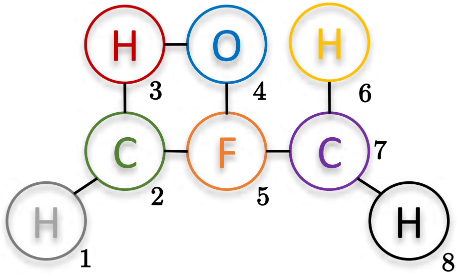
Weisfeiler-Lehman GNNs

- Motivation : Characterize expressivity power of GNNs with graph isomorphism.
- Graph isomorphism : Two graphs are isomorphic if there exists an index permutation between the nodes that preserves node adjacencies.
- Determining whether two graphs are isomorphic is NP-intermediate : It is not known if a polynomial time algorithm exists, or the problem is NP-hard.
- Weisfeiler-Lehman (WL) proposed an algorithm[1] to test if two graphs are not isomorphic. However, the WL test is not sufficient to guarantee that two graphs are isomorphic.



Graph 1

⇕ Isomorphic graphs



Graph 2

[1] B Weisfeiler, A Lehman, A reduction of a graph to a canonical form and an algebra arising during this reduction, 1968

WL Test[1]

- Idea : Design an injective "coloring" function f_{WL} :

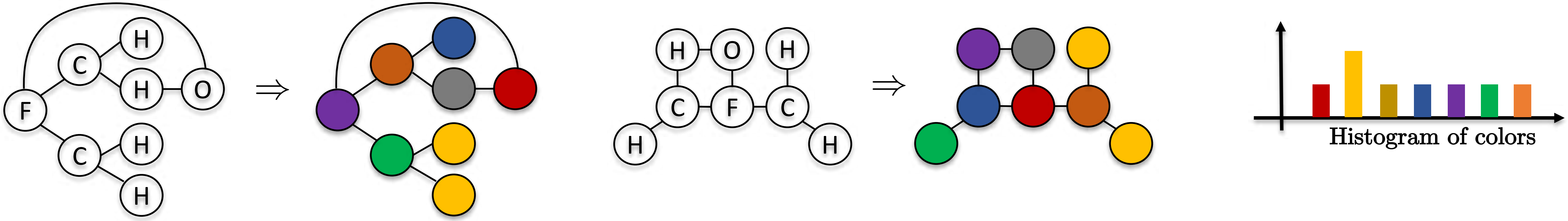
iteration

$$f_{\mathrm{WL}}\left(c_i^{\ell}, \{c_j^{\ell}\}_{j\in\mathcal{N}_i}\right) = c_i^{\ell+1}$$

Multiset function

Multiset (set of unordered and repetitive elements)

- WL algorithm iteratively applies function f_{WL} until no new colors are created.
- This produces a canonical representation of a graph as a histogram of colors.
- If two graphs have different color histograms (up to all color permutations) then the two graphs are guaranteed to be non-isomorphic.



[1] B Weisfeiler, A Lehman, A reduction of a graph to a canonical form and an algebra arising during this reduction, 1968

Graph Isomorphism Networks (GINs)[1]

- Goal : Design a GNN as expressive as the WL test to distinguish non-isomorphic graphs.
- Node aggregation of the form :

$$f_{\mathrm{NN}}\big(h_i^\ell, \{h_j^\ell\}_{j\in\mathcal{N}_i}\big) = (1+\varepsilon)g(h_i^\ell) + \sum_{j\in\mathcal{N}_i} g(h_j^\ell)$$

irrational injective sum

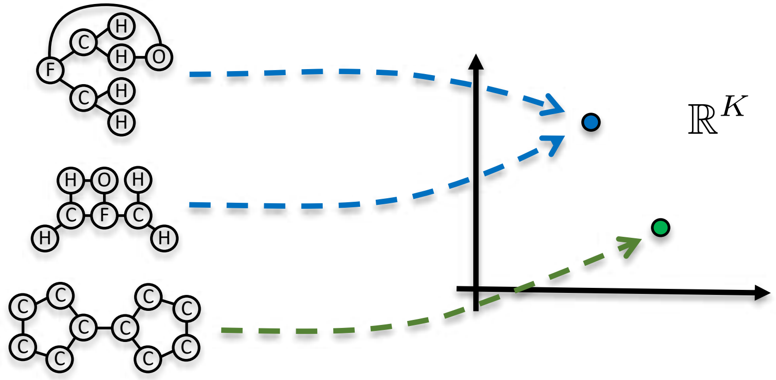
- It is difficult to design an analytical injective function ⇒ MLP is used to approximate g (existence guaranteed by the universal approximation theorem) :

$$h_i^{\ell+1} = f_{\mathrm{GIN}}\big(h_i^\ell, \{h_j^\ell\}_{j\in\mathcal{N}_i}\big) = \mathrm{MLP}^\ell\Big((1+\varepsilon)h_i^\ell + \sum_{j\in\mathcal{N}_i} h_j^\ell\Big)$$

- Graph readout function must also be injective :

$$h_{\mathcal{G}} = \mathrm{MLP}\Big(\sum_{i\in V} h_i^L\Big) \in \mathbb{R}^K$$

Sum function

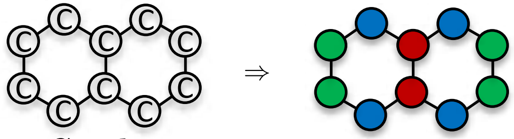


- Pioneer work with[2] on GNN expressivity.

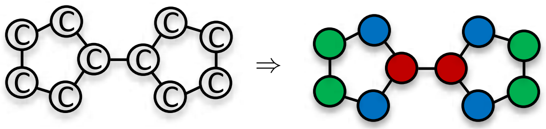
[1] Xu, Hu, Leskovec, Jegelka, How powerful are graph neural networks?, 2019
[2] Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe, Weisfeiler and leman go neural: Higher-order graph networks, 2019

Limitation

- WL test[1] is not a sufficient condition, it can fail to differentiate non-isomorphic graphs :
  - Example of two graphs with same color signatures but not isomorphic :



Graph 1



Graph 2

  - Another example w/ Circular Skip Length (CSL) graphs[2] (a circular graph with skip connections for each node to R-hop nodes) :

CSL(n=11, R=2)

CSL(n=11, R=3)

CSL(n=11, R=4)

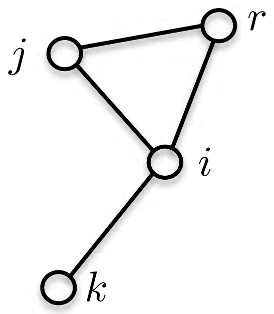
Nodes are isomorphic

  - Can we improve the expressivity of the original WL test ?

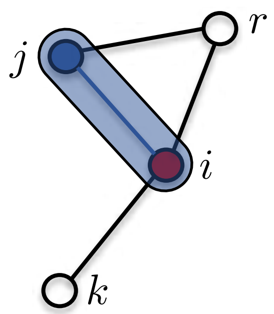
[1] B Weisfeiler, A Lehman, A reduction of a graph to a canonical form and an algebra arising during this reduction, 1968
[2] Murphy, Srinivasan, Rao, Ribeiro, Relational pooling for graph representations, 2019

k-WL Test

- Can we design better WL tests ?
  - Original WL test uses 2-tuple of nodes to produce colors. To produce more colors and improve expressivity of WL tests, k-tuple of nodes with $k \geq 3$ can be used (higher-order interactions between nodes).

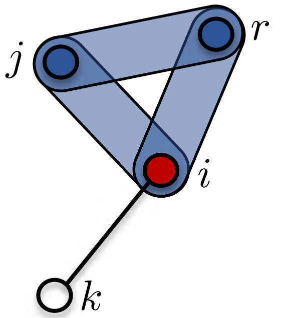


$\mathcal{G} = (V, E)$



Edges $= \{i, j\}, \{i, r\}, \{i, k\}$

2-tuple of nodes can distinguish non-isomorphic graphs with the 1-WL/2-WL tests.



Hyper-edges $= \{i, j, r\}, \{i, k, r\}, \{i, j, k\}$

3-tuple of nodes can distinguish non-isomorphic graphs with the 3-WL test.

Equivariant GNNs[1]

- How to design GNNs with the same expressivity power as the k-WL test ?
  - Let us define a k-order equivariant GNNs :

Permutation

$f_W(P \circ h) = P \circ f_W(h)$

Equivariant linear layers

MLP

$$y = m_{W^{L+1}} \circ g_{W^L} \circ \sigma \circ f_{W^{L-1}} \circ \sigma \circ f_{W^{L-2}} \circ ... \sigma \circ f_{W^0} \circ h_0$$

$$\text{with } k = \max_{\ell \in [0, L-1]} k_\ell$$

$g_W(P \circ h) = g_W(h) \in \mathbb{R}^K, K \geq 1$

Invariant linear layer

$$\text{and } f_{W^\ell} : \mathbb{R}^{n^{k_\ell} \times d_\ell} \to \mathbb{R}^{n^{k_{\ell+1}} \times d_{\ell+1}}$$

- Theorem[1] : There exist k-order E-GNNs that can distinguish non-isomorphic graphs with the k-WL test.
  - However, k-order E-GNNs requite $O(n^k)$ memory/speed complexities.
    - Note that we need at least k=*3* (hence $O(n^3)$) to be more powerful than GINs as GINs have the discriminative power of the *1*-WL/*2*-WL tests[2].

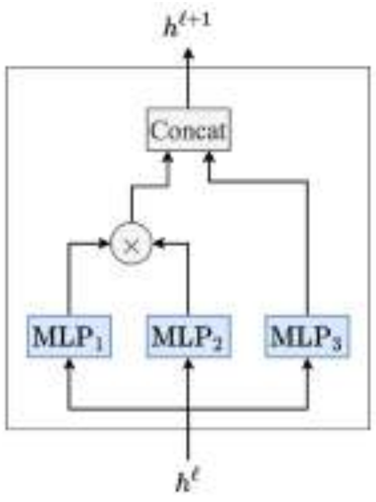
[1] Maron, Ben-Hamu, Shamir, Lipman, Invariant and equivariant graph networks, 2019
[2] Cai, Furer, Immerman, An optimal lower bound on the number of variables for graph identification, 1992

3-WL GNNs[1]

- How to design GNNs that are 3-WL expressive but do not require $O(n^3)$ memory/speed complexities?
- 3-WL GNNs : To achieve higher interactions between nodes, it is sufficient to multiply second-order tensors feature-wise.
  - Theorem[1] : There exist 3-WL GNNs as expressive as the 3-WL test.
  - Memory is quadratic $O(n^2)$ but matrix multiplication implies $O(n^3)$ speed.
    - Also, matrix multiplication densifies sparse matrix.
  - This is the simplest, most scalable and most expressive GNNs in terms of 3-WL test.
- 3-WL GNN update layer :

$$h^{\ell+1} = \text{Concat}\big(m_{W_1^\ell}(h^\ell) \cdot m_{W_2^\ell}(h^\ell),\ m_{W_3^\ell}(h^\ell)\big)$$

where $h^{\ell+1} \in \mathbb{R}^{n \times n \times d_{\ell+1}}, h^\ell \in \mathbb{R}^{n \times n \times d_\ell}, W_1, W_2, W_3 \in \mathbb{R}^{2 \times d_\ell \times d_{\ell+1}}$



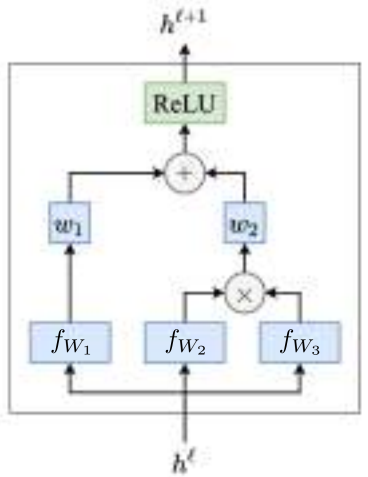
[1] Maron, Ben-Hamu, Serviansky, Lipman, Provably powerful graph networks, 2019

RingGNNs[1]

- Related method to *3*-WL GNNs to design more expressive GNNs than GINs.
  - Higher-order interaction between nodes is produced by multiplying equivariant linear layers[2].
  - RingGNN update layer :

$$h^{\ell+1} = \sigma\big(w_1^\ell \ f_{W_1^\ell}(h^\ell) + w_2^\ell \ f_{W_2^\ell}(h^\ell) \cdot f_{W_3^\ell}(h^\ell)\big),$$

where $h^{\ell+1} \in \mathbb{R}^{n \times n \times d_{\ell+1}}, h^\ell \in \mathbb{R}^{n \times n \times d_\ell}, w_{1,2}^\ell \in \mathbb{R}, W_1, W_2, W_3 \in \mathbb{R}^{d_\ell \times d_{\ell+1} \times 17},$



where f_W are the equivariant linear layers defined as :

$$\big(f_W(h)\big)_{\cdot,\cdot,q'} = \sum_{k=1}^{15+2} \sum_{q=1}^{d_\ell} W_{k,q,q'} \ f_k(h_{\cdot,\cdot,q}) \ \in \mathbb{R}^{n \times n \times d_{\ell+1}}$$

and f_k are all possible 15 equivariant linear functions $f_k : \mathrm{R}^{n \times n} \to \mathrm{R}^{n \times n}$ for a given tensor $h \in \mathrm{R}^{n \times n}$, and 2 bias functions.

[1] Chen, Villar, Chen, Bruna, On the equivalence between graph isomorphism testing and function approximation with gnns, 2019
[2] Maron, Ben-Hamu, Shamir, Lipman, Invariant and equivariant graph networks, 2019

Expressivity Hierarchy w.r.t. WL Test

MP-GCNs
Vanilla GCNs [1]
GraphSage [2]
GAT [3]
GatedGCNs [4]
$O(n)$

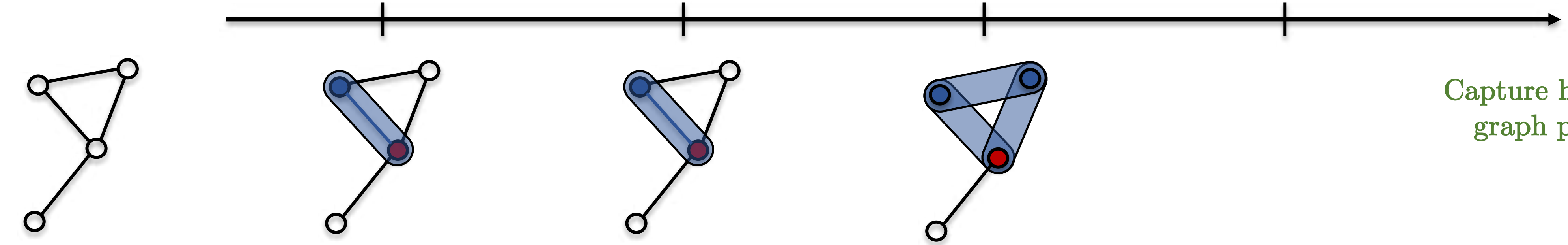
GIN [5]
$O(n)$

3-WL GNNs [6]
RingGNNs [7]
$O(n^2)/O(n^3)$
Morris-etal [8]
$O(n^3)/O(n^4)$

k-WL GNNs [9]
$O(n^k)$

≤ powerful than 1-WL ≤ 1-WL/ 2-WL < 3-WL < k-WL

Expressive power measured by WL tests



Capture higher-order graph properties

$\mathcal{G} = (V, E)$

[1] Kipf, Welling, Semi-supervised classification with graph convolutional networks, 2017
[2] Hamilton, Ying, Leskovec, Inductive representation learning on large graphs, 2017
[3] Velickovic, Cucurull, Casanova, Romero, Lio, Bengio, Graph attention networks, 2017
[4] Bresson, Laurent, Residual gated graph convnets, 2017
[5] Xu, Hu, Leskovec, Jegelka, How powerful are graph neural networks?, 2019
[6] Maron, Ben-Hamu, Serviansky, Lipman, Provably powerful graph networks, 2019
[7] Chen, Villar, Chen, Bruna, On the equivalence between graph isomorphism testing and function approximation with gnns, 2019
[8] Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe, Weisfeiler and leman go neural: Higher-order graph neural networks, 2019
[9] Maron, Ben-Hamu, Shamir, Lipman, Invariant and equivariant graph networks, 2019

Outline

- Motivation
- Message-Passing GCNs
- Weisfeiler-Lehman GNNs
- Graph-Agnostic GNNs
- Datasets
- Infrastructure and Experimental Setting
- Benchmarking Results
- Laplacian Positional Encodings
- Link Prediction with Edge Representation
- Conclusion

MLP Baseline

- Graph-agnostic NNs : As a sanity check, we compare GNNs to a simple MLP network which updates each node independent of one-other :

$$h_i^{\ell+1} = \sigma\Big(W^\ell \ h_i^\ell\Big), \ h^{\ell+1} \in \mathbb{R}^{n\times d_{\ell+1}}, \ h^\ell \in \mathbb{R}^{n\times d_\ell}, \ W^\ell \in \mathbb{R}^{d_{\ell+1}\times d_\ell}$$

and passes these features to the task-based readout layer.

Outline

- Motivation
- Message-Passing GCNs
- Weisfeiler-Lehman GNNs
- Graph-Agnostic GNNs
- Datasets
- Infrastructure and Experimental Setting
- Benchmarking Results
- Laplacian Positional Encodings
- Link Prediction with Edge Representation
- Conclusion

Datasets

- Main issue with prevalent datasets : Small sizes
- Ideal dataset : Representative, realistic, and medium/large-scale size.
- Challenges :
  - What theoretical tool to define the quality of dataset or validate its statistical representativeness for a given task ?
  - Several arbitrary choices when preparing graphs, such as node and edge features. For example, e-commerce product features.
  - Appropriate size may depend on the task complexity as well as the dimensionality and statistics of underlying data.
- The recent Open Graph Benchmark[1] (OGB) project is a much needed initiative to tackle these challenges.
  - OGB offers a collection of medium/large-scale real-world graph datasets and evaluation protocols, with an emphasis on out-of-distribution generalization performance through meaningful data splits.

[1] Hu, Fey, Zitnik, Dong, Ren, Liu, Catasta, Leskovec, Open graph benchmark: Datasets for machine learning on graphs, 2020

Proposed Datasets

- Appropriate datasets :
  - Datasets that are able to statistically separate the performance of GNNs.

- Summary of the 7 medium-scale datasets :

Domain & Construction	Dataset	#Graphs	#Nodes	Total #Nodes	Task
Chemistry: Real-world molecular graphs	ZINC	12K	9-37	277,864	Graph Regression
Mathematical Modelling: Artificial graphs generated from Stochastic Block Models	PATTERN CLUSTER	14K 12K	44-188 41-190	1,664,491 1,406,436	Node Classification
Computer Vision: Graphs constructed with SLIC super-pixels of images	MNIST CIFAR10	70K 60K	40-75 85-150	4,939,668 7,058,005	Graph Classification
Combinatorial Optimization: Uniformly generated artificial Euclidean graphs	TSP	12K	50-500	3,309,140	Edge Classification
Social Networks: Real-world citation graph	COLLAB	1	235,868	235,868	Edge Classification
Circular Skip Links: Isomorphic graphs with same degree	CSL	150	41	6,150	Graph Classification

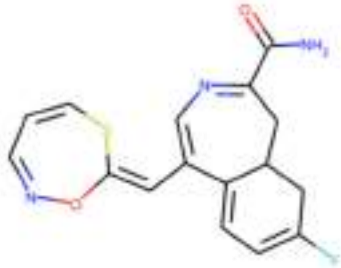
4 datasets are artificially generated, 2 datasets are semi-artificial, and 2 are real-world.

Sizes in terms of total number of nodes vary between 0.27M to 7M.

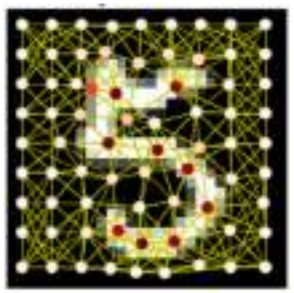
4 most fundamental graph tasks : graph regression, graph classification, node classification and link prediction.

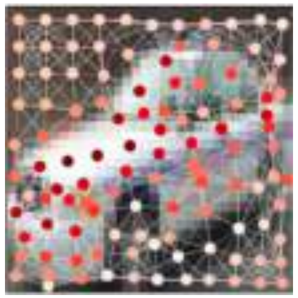
Proposed Datasets

- ZINC[1] : A popular real-world molecular dataset of 250K graphs, out of which we randomly select 12K for efficiency. We consider the task of graph property regression for constrained solubility, an important chemical property for designing generative GNNs for molecules[2].
  - Statistics : 10,000 train/1,000 validation/1,000 test graphs of sizes 9-37 nodes/heavy atoms.



- MNIST[3]/CIFAR10[4] : Classical image classification datasets converted into graphs using super-pixels[5] and assigning node features as the super-pixel coordinates and mean intensity. These datasets are sanity-checks, as we expect most GNNs to perform close to 100% for MNIST and well enough for CIFAR10.
  - Statistics : MNIST has 55,000 train/5,000 validation/10,000 test graphs of sizes 40-75 nodes and CIFAR10 has 45,000 train/5,000 validation/10,000 test graphs of sizes 85-150 nodes.

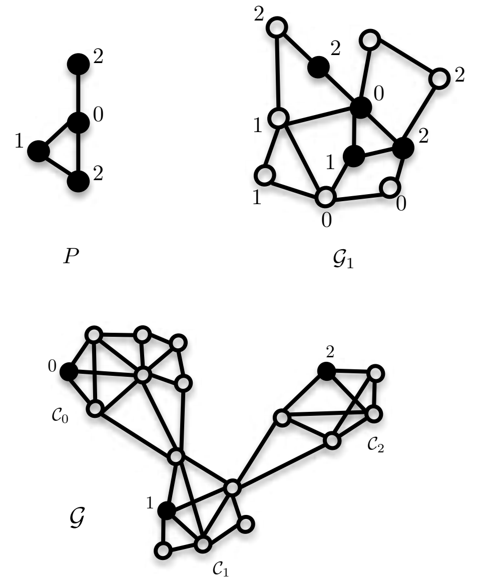




[1] Irwin, Sterling, Mysinger, Bolstad, Coleman, Zinc: a free tool to discover chemistry for biology, 2012
[2] Jia, Lin, Ying, You, Leskovec, Aiken, Redundancy-free computation graphs for graph neural networks, 2019
[3] LeCun, Bottou, Bengio, Haffner, Gradient-based learning applied to document recognition, 1998
[4] Alex Krizhevsky etal, Learning multiple layers of features from tiny images, 2009
[5] Achanta, Shaji, Smith, Lucchi, Fua, Süsstrunk, Slic superpixels compared to state-of-the-art superpixel methods, 2012

Proposed Datasets

- PATTERN/CLUSTER : Node classification tasks generated with Stochastic Block Models[1], which are widely used to model communities in social networks by modulating the intra- and extra-communities connections. PATTERN tests the fundamental task of recognizing specific predetermined subgraphs[2].
  - Statistics : PATTERN has 10,000 train/2,000 validation/2,000 test graphs of sizes 50-180 nodes. CLUSTER has 10,000 train/1,000 validation/1,000 test graphs of sizes 40-190 nodes.



- CSL[3] : Synthetic to test the expressivity of GNNs. Graphs are isomorphic if they have the same degree and the task is to classify non-isomorphic graphs.
  - Statistics : 5-fold cross-validation split of 150 graphs of sizes 41 nodes with train-val-test ratio 3:1:1.

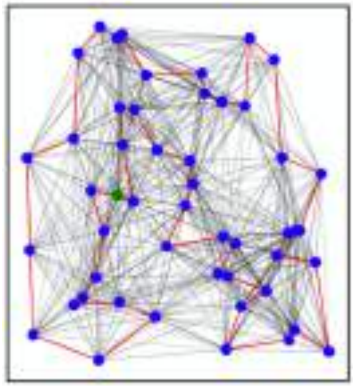

CSL(n=11, R=2)


CSL(n=11, R=4)

[1] Abbe, Community detection and stochastic block models: recent developments, 2017
[2] Scarselli, Gori, Tsoi, Hagenbuchner, Monfardini, The Graph Neural Network Model, 2009
[3] Murphy, Srinivasan, Rao, Ribeiro, Relational pooling for graph representations, 2019

Proposed Datasets

- Travelling Salesman Problem (TSP) : Link prediction on 2D Euclidean graphs to identify edges belonging to the optimal TSP solution given by Concorde[1]. TSP is the most studied NP-hard combinatorial problem with a growing body of literature on leveraging GNNs to learn better solvers[2,3].
  - Statistics : 10,000 train TSPs / 1,000 validation TSPs / 1,000 test TSPs, where the number of nodes is randomly selected in [50, 500].



- OGB-COLLAB[4] : Link prediction dataset proposed by OGB corresponding to a collaboration network between scientists. The task is to predict future author collaboration relationships given past collaboration links.
  - Statistics : A single large temporal graph of size 235K nodes with given train/validation/test edge splits.

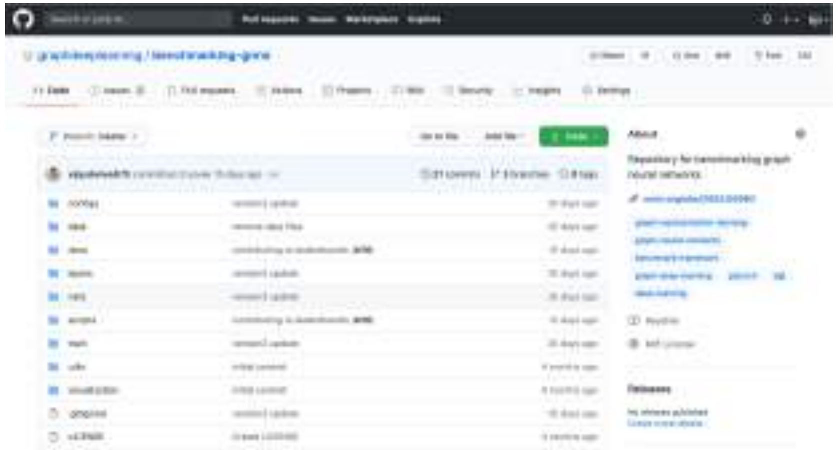
[1] Applegate, Bixby, Chvatal, Cook, Concorde tsp solver, 2006
[2] Vinyals, Fortunato, Jaitly, Pointer, 2015
[3] Bello, Pham, Le, Norouzi, Bengio, Neural combinatorial optimization with reinforcement learning, 2016
[4] Hu, Fey, Zitnik, Dong, Ren, Liu, Catasta, Leskovec, Open graph benchmark: Datasets for machine learning on graphs, 2020

Outline

- Motivation
- Message-Passing GCNs
- Weisfeiler-Lehman GNNs
- Graph-Agnostic GNNs
- Datasets
- Infrastructure and Experimental Setting
- Benchmarking Results
- Laplacian Positional Encodings
- Link Prediction with Edge Representation
- Conclusion

Benchmark Infrastructure

- GitHub Repo :
  https://github.com/graphdeeplearning/benchmarking-gnns
- Objectives :
  - Ease-of-use and modular, enabling new users to experiment and study the building blocks of GNNs.
  - Experimentally rigorous and fair for all models being benchmarked.
  - Being comprehensive for tracking progress of new GNNs and novel dataset/task.
- Components :
  - Data pipelines
  - GNN layers and models
  - Training and evaluation functions
  - Network and hyperparameter configurations
  - Scripts for reproducibility



Experimental Setting

- Data splits : Given for ZINC, MNIST, CIFAR10, CSL, OGB-COLLAB, random for PATTERN, CLUSTER, TSP.
- Training :
  - Adam optimizer w/ learning rate decay strategy.

    Initial learning rate is 1e-3/1e-4, reduced by half if validation loss does not improve after 5/10 epochs.

    Training is stopped if learning rate ≤ 1e-6 or computational time ≥ 12 hours.
  - Statistics with 4 results using 4 different seeds are reported.
- Parameter budgets :
  - Our goal is not to find the optimal hyperparameters (computationally expensive) but to compare models within the same parameter budget and a maximal computational time.
  - Two parameter budgets :
    - 100k (w/ L=4) and 500k (w/ L=16) parameters for each GNNs for all tasks.

Outline

- Motivation
- Message-Passing GCNs
- Weisfeiler-Lehman GNNs
- Graph-Agnostic GNNs
- Datasets
- Infrastructure and Experimental Setting
- **Benchmarking Results**
- Laplacian Positional Encodings
- Link Prediction with Edge Representation
- Conclusion

Benchmarking Results

- Result #1 : MP-GCNs outperformed WL-GNNs on all datasets.
  - Potential reasons :
    - $O(n^2)/O(n^3)$ memory/speed complexities of WL-GNNs do not scale to medium-scale datasets.
      - Best result for ZINC, which is the dataset with the smallest sizes, $n \in [9,37]$.
    - Early stage of developments of WL-GNNs.

- Result #2 : MP-GCNs benefit from batch normalization and residual connection :
  - Boost performance
  - Accelerate training
  - WL-GNNs do not benefit from RC, BN or LN (performances degrade).
    - Batch implementation is carried out by gradient accumulation.

NODE CLASSIFICATION

Model	L	PATTERN #Param	PATTERN Test Acc.±s.d.	PATTERN Train Acc.±s.d.	PATTERN #Epoch	PATTERN Epoch/Total	CLUSTER #Param	CLUSTER Test Acc.±s.d.	CLUSTER Train Acc.±s.d.	CLUSTER #Epoch	CLUSTER Epoch/Total
MLP	4	105263	50.519±0.000	50.487±0.014	42.25	8.95s/0.11hr	106015	20.973±0.004	20.938±0.002	42.25	5.83s/0.07hr
GCN	4	100923	63.880±0.074	65.126±0.135	105.00	118.85s/3.51hr	101655	53.445±2.029	54.041±2.197	70.00	65.72s/1.30hr
	16	500823	71.892±0.334	78.409±1.592	81.50	492.19s/11.31hr	501687	68.498±0.976	71.729±2.212	79.75	270.28s/6.08hr
GraphSage	4	101739	50.516±0.001	50.473±0.014	43.75	93.41s/1.17hr	102187	50.454±0.145	54.374±0.203	64.00	53.56s/0.97hr
	16	502842	50.492±0.001	50.487±0.005	46.50	391.19s/5.19hr	503350	63.844±0.110	86.710±0.167	57.75	225.61s/3.70hr
MoNet	4	103775	85.482±0.037	85.569±0.044	89.75	35.71s/0.90hr	104227	58.064±0.131	58.454±0.183	76.25	24.29s/0.52hr
	16	511487	85.582±0.038	85.720±0.068	81.75	68.49s/1.58hr	511999	66.407±0.540	67.727±0.649	77.75	47.82s/1.05hr
GAT	4	109936	75.824±1.823	77.883±1.632	96.00	20.92s/0.57hr	110700	57.732±0.323	58.331±0.342	67.25	14.17s/0.27hr
	16	526990	78.271±0.186	90.212±0.476	53.50	50.33s/0.77hr	527874	70.587±0.447	76.074±1.362	73.50	35.94s/0.75hr
GatedGCN	4	104003	84.480±0.122	84.474±0.155	78.75	139.01s/3.09hr	104355	60.404±0.419	61.618±0.536	94.50	79.97s/2.13hr
	16	502223	85.568±0.088	86.007±0.123	65.25	644.71s/11.91hr	502615	73.840±0.326	87.880±0.908	60.00	400.07s/6.81hr
GatedGCN-PE	16	502457	**86.508±0.085**	86.801±0.133	65.75	647.94s/12.08hr	504253	**76.082±0.196**	88.919±0.720	57.75	399.66s/6.58hr
GIN	4	100884	85.590±0.011	85.852±0.030	93.00	15.24s/0.40hr	103544	58.384±0.236	59.480±0.337	74.75	10.71s/0.23hr
	16	508574	85.387±0.136	85.664±0.116	86.75	25.14s/0.62hr	517570	64.716±1.553	65.973±1.816	80.75	20.67s/0.47hr
RingGNN	2	105206	86.245±0.013	86.118±0.034	75.00	573.37s/12.17hr	104746	42.418±20.063	42.520±20.212	74.50	428.24s/8.79hr
	2	504766	86.244±0.025	86.105±0.021	72.00	595.97s/12.15hr	524202	22.340±0.000	22.304±0.000	43.25	501.84s/6.22hr
	8	505749	Diverged	Diverged	Diverged	Diverged	514380	Diverged	Diverged	Diverged	Diverged
3WLGNN	3	103572	85.661±0.353	85.608±0.337	95.00	304.79s/7.88hr	105552	57.130±6.539	57.404±6.597	116.00	219.51s/6.52hr
	3	502872	85.341±0.207	85.270±0.198	81.75	424.23s/9.56hr	507252	55.489±7.863	55.736±8.024	66.00	319.98s/5.79hr
	8	581716	Diverged	Diverged	Diverged	Diverged	586788	Diverged	Diverged	Diverged	Diverged

GRAPH CLASSIFICATION

Model	L	MNIST #Param	MNIST Test Acc.±s.d.	MNIST Train Acc.±s.d.	MNIST #Epoch	MNIST Epoch/Total	CIFAR10 #Param	CIFAR10 Test Acc.±s.d.	CIFAR10 Train Acc.±s.d.	CIFAR10 #Epoch	CIFAR10 Epoch/Total
MLP	4	104044	95.340±0.138	97.432±0.470	232.25	22.74s/1.48hr	104380	56.340±0.181	65.113±1.685	185.25	29.48s/1.53hr
GCN	4	101365	90.705±0.218	97.196±0.223	127.50	83.41s/2.99hr	101657	55.710±0.381	69.523±1.948	142.50	109.70s/4.39hr
GraphSage	4	104337	**97.312±0.097**	100.000±0.000	98.25	113.12s/3.13hr	104517	65.767±0.308	99.719±0.062	93.50	124.61s/3.29hr
MoNet	4	104049	90.805±0.032	96.609±0.440	146.25	93.19s/3.82hr	104229	54.655±0.518	65.911±2.515	141.50	97.13s/3.85hr
GAT	4	110400	95.535±0.205	99.994±0.008	104.75	42.26s/1.25hr	110704	64.223±0.455	89.114±0.499	103.75	55.27s/1.62hr
GatedGCN	4	104217	**97.340±0.143**	100.000±0.000	96.25	128.79s/3.50hr	104357	**67.312±0.311**	94.553±1.018	97.00	154.15s/4.22hr
GIN	4	105434	96.485±0.252	100.000±0.000	128.00	39.22s/1.41hr	105654	55.255±1.527	79.412±9.700	141.50	52.12s/2.07hr
RingGNN	2	105398	11.350±0.000	11.235±0.000	14.00	2945.69s/12.77hr	105165	19.300±16.108	19.556±16.397	13.50	3112.96s/13.00hr
	2	505182	91.860±0.449	92.169±0.505	16.25	2575.99s/12.63hr	504949	39.165±17.114	40.209±17.790	13.75	2998.24s/12.60hr
	8	506357	Diverged	Diverged	Diverged	Diverged	510439	Diverged	Diverged	Diverged	Diverged
3WLGNN	3	108024	95.075±0.961	95.830±1.338	27.75	1523.20s/12.40hr	108516	59.175±1.593	63.751±2.697	28.50	1506.29s/12.60hr
	3	501690	95.002±0.419	95.692±0.677	26.25	1608.73s/12.42hr	502770	58.043±2.512	61.574±3.575	20.00	2091.22s/12.55hr
	8	500816	Diverged	Diverged	Diverged	Diverged	501584	Diverged	Diverged	Diverged	Diverged

LINK PREDICTION

Model	L	TSP #Param	TSP Test F1±s.d.	TSP Train F1±s.d.	TSP #Epoch	TSP Epoch/Total	COLLAB #Param ($L=3$)	COLLAB Test Hits±s.d.	COLLAB Train Hits±s.d.	COLLAB #Epoch	COLLAB Epoch/Total
MLP	4	96956	0.544±0.001	0.544±0.001	164.25	50.15s/2.31hr	39441	20.350±2.168	29.807±3.360	147.50	2.09s/0.09hr
GCN	4	95702	0.630±0.001	0.631±0.001	261.00	152.89s/11.15hr	40479	50.422±1.131	92.112±0.991	122.50	351.05s/12.04hr
GraphSage	4	99263	0.665±0.003	0.669±0.003	266.00	157.26s/11.68hr	39856	51.618±0.690	99.949±0.052	152.75	277.93s/11.87hr
MoNet	4	99007	0.641±0.002	0.643±0.002	282.00	84.46s/6.65hr	39751	36.144±2.191	61.156±3.973	167.50	26.69s/1.26hr
GAT	4	96182	0.671±0.002	0.673±0.002	328.25	68.23s/6.25hr	42637	51.501±0.962	97.851±1.114	157.00	18.12s/0.80hr
GatedGCN	4	97858	0.791±0.003	0.793±0.003	159.00	218.20s/9.72hr	40965	52.635±1.168	96.103±1.876	95.00	453.47s/12.09hr
GatedGCN-PE				-			41889	**52.849±1.345**	96.165±0.453	94.75	452.75s/12.08hr
GatedGCN-E	4	97858	0.808±0.003	0.811±0.003	197.00	218.51s/12.04hr	40965	49.212±1.560	88.747±1.058	95.00	451.21s/12.03hr
GatedGCN-E	16	500770	**0.838±0.002**	0.850±0.001	53.00	807.23s/12.17hr			-		
GIN	4	99002	0.656±0.003	0.660±0.003	273.50	72.73s/5.56hr	39544	41.730±2.284	70.555±4.444	140.25	8.66s/0.34hr
RingGNN	2	106862	0.643±0.024	0.644±0.024	2.00	17850.52s/17.19hr	-	OOM			
	2	507938	0.704±0.003	0.705±0.003	3.00	12835.53s/16.08hr	-	OOM			
	8	506564	Diverged	Diverged	Diverged	Diverged	-	OOM	RingGNN and 3WLGNN rely on dense tensors which leads to **OOM** on both GPU and CPU memory.		
3WLGNN	3	106366	0.694±0.073	0.695±0.073	2.00	17468.81s/16.59hr	-	OOM			
	3	506681	0.288±0.311	0.290±0.312	2.00	17190.17s/16.51hr	-	OOM			
	8	508832	OOM	OOM	OOM	OOM	-	OOM			
k-NN Heuristic		k =2	Test F1: 0.693								
Matrix Fact.	0			-			60546561	44.206±0.452	100.000±0.000	254.33	2.66s/0.21hr

GRAPH REGRESSION – ZINC

Model	L	#Param	Test MAE±s.d.	Train MAE±s.d.	#Epoch	Epoch/Total
MLP	4	108975	0.706±0.006	0.644±0.005	116.75	1.01s/0.03hr
GCN	4	103077	0.459±0.006	0.343±0.011	196.25	2.89s/0.16hr
	16	505079	0.367±0.011	0.128±0.019	197.00	12.78s/0.71hr
GraphSage	4	94977	0.468±0.003	0.251±0.004	147.25	3.74s/0.15hr
	16	505341	0.398±0.002	0.081±0.009	145.50	16.61s/0.68hr
MoNet	4	106002	0.397±0.010	0.318±0.016	188.25	1.97s/0.10hr
	16	504013	0.292±0.006	0.093±0.014	171.75	10.82s/0.52hr
GAT	4	102385	0.475±0.007	0.317±0.006	137.50	2.93s/0.11hr
	16	531345	0.384±0.007	0.067±0.004	144.00	12.98s/0.53hr
GatedGCN	4	105735	0.435±0.011	0.287±0.014	173.50	5.76s/0.28hr
GatedGCN-E	4	105875	0.375±0.003	0.236±0.007	194.75	5.37s/0.29hr
	16	504309	0.282±0.015	0.074±0.016	166.75	20.50s/0.96hr
GatedGCN-E-PE	16	505011	**0.214±0.013**	0.067±0.019	185.00	10.70s/0.56hr
GIN	4	103079	0.387±0.015	0.319±0.015	153.25	2.29s/0.10hr
	16	509549	0.526±0.051	0.444±0.039	147.00	10.22s/0.42hr
RingGNN	2	97978	0.512±0.023	0.383±0.020	90.25	327.65s/8.32hr
RingGNN-E	2	104403	0.363±0.026	0.243±0.025	95.00	366.29s/9.76hr
	2	527283	0.353±0.019	0.236±0.019	79.75	293.94s/6.63hr
	8	510305	Diverged	Diverged	Diverged	Diverged
3WLGNN	3	102150	0.407±0.028	0.272±0.037	111.25	286.23s/8.88hr
3WLGNN-E	3	103098	0.256±0.054	0.140±0.044	117.25	334.69s/10.90hr
	3	507603	0.303±0.068	0.173±0.041	120.25	329.49s/11.08hr
	8	582824	0.303±0.057	0.246±0.043	52.50	811.27s/12.15hr

Evaluation Metrics: (higher is better, except for ZINC)
- CLUSTER, PATTERN use weighted accuracy w.r.t. the class sizes.
- MNIST, CIFAR10 use multi-label classification accuracy.
- TSP uses binary F1 score for the positive edges.
- COLLAB uses Hits@50 via the evaluator provided by OGB [?].
- ZINC uses mean absolute regression error.

Notation:
- Models with the suffix **-E** use input edge features to initialize edge representations (ZINC: bond type, TSP: Euclidean distance, COLLAB: collaboration frequency and year).
- Models with the suffix **-PE** use Laplacian Eigenvectors as node positional encodings, with dimension 8 for ZINC, 2 for PATTERN and 20 for others.
- Results denoted by **Diverged** indicate unstable and divergent runs across all 4 seeds and initial learning rate values $\{10^{-3}, 10^{-4}, 10^{-5}\}$.
- Results denoted by **OOM** indicate runs which throw out of memory errors on our hardware configuration.

Extended training:
- For TSP, RingGNN/3WLGNN with 100K parameters achieved 0.733±0.020 and 0.649±0.051 respectively after 48hr of training.

Benchmarking Results

- Result #3 : Anisotropic mechanism improve (isotropic) GCNs.
  - Sparse attention[1,2] and dense attention[3] are not injective (unlike f_{GIN}) but experiments showed that they are good at generalization. See also[4].
  - Intuitively, softmax attention is flexible to represent max/mean/weighted mean w.r.t. contextual information.
  - Recent work[5] studied effectiveness of global attention for generalization.

NODE CLASSIFICATION

		PATTERN					CLUSTER				
Model	L	#Param	Test Acc.±s.d.	Train Acc.±s.d.	#Epoch	Epoch/Total	#Param	Test Acc.±s.d.	Train Acc.±s.d.	#Epoch	Epoch/Total
MLP	4	105263	50.519±0.000	50.487±0.014	42.25	8.95s/0.11hr	106015	20.973±0.004	20.938±0.002	42.25	5.83s/0.07hr
GCN	4	100923	63.880±0.074	65.126±0.135	105.00	118.85s/3.51hr	101655	53.445±2.029	54.041±2.197	70.00	65.72s/1.30hr
	16	500823	71.892±0.334	78.409±1.592	81.50	492.19s/11.31hr	501687	68.498±0.976	71.729±2.212	79.75	270.28s/6.08hr
GraphSage	4	101739	50.516±0.001	50.473±0.014	43.75	93.41s/1.17hr	102187	50.454±0.145	54.374±0.203	64.00	53.56s/0.97hr
	16	502842	50.492±0.001	50.487±0.005	46.50	391.19s/5.19hr	503350	63.844±0.110	86.710±0.167	57.75	225.61s/3.70hr
MoNet	4	103775	85.482±0.037	85.569±0.044	89.75	35.71s/0.90hr	104227	58.064±0.131	58.454±0.183	76.25	24.29s/0.52hr
	16	511487	85.582±0.038	85.720±0.068	81.75	68.49s/1.58hr	511999	66.407±0.540	67.727±0.649	77.75	47.82s/1.05hr
GAT	4	109936	75.824±1.823	77.883±1.632	96.00	20.92s/0.57hr	110700	57.732±0.323	58.331±0.342	67.25	14.17s/0.27hr
	16	526990	78.271±0.186	90.212±0.476	53.50	50.33s/0.77hr	527874	70.587±0.447	76.074±1.362	73.50	35.94s/0.75hr
GatedGCN	4	104003	84.480±0.122	84.474±0.155	78.75	139.01s/3.09hr	104355	60.404±0.419	61.618±0.536	94.50	79.97s/2.13hr
	16	502223	85.568±0.088	86.007±0.123	65.25	644.71s/11.91hr	502615	73.840±0.326	87.880±0.908	60.00	400.07s/6.81hr
GatedGCN-PE	16	502457	**86.508±0.085**	86.801±0.133	65.75	647.94s/12.08hr	504253	**76.082±0.196**	88.919±0.720	57.75	399.66s/6.58hr
GIN	4	100884	85.590±0.011	85.852±0.030	93.00	15.24s/0.40hr	103544	58.384±0.236	59.480±0.337	74.75	10.71s/0.23hr
	16	508574	85.387±0.136	85.664±0.116	86.75	25.14s/0.62hr	517570	64.716±1.553	65.973±1.816	80.75	20.67s/0.47hr
RingGNN	2	105206	86.245±0.013	86.118±0.034	75.00	573.37s/12.17hr	104746	42.418±20.063	42.520±20.212	74.50	428.24s/8.79hr
	2	504766	86.244±0.025	86.105±0.021	72.00	595.97s/12.15hr	524202	22.340±0.000	22.304±0.000	43.25	501.84s/6.22hr
	8	505749	Diverged	Diverged	Diverged	Diverged	514380	Diverged	Diverged	Diverged	Diverged
3WLGNN	3	103572	85.661±0.353	85.608±0.337	95.00	304.79s/7.88hr	105552	57.130±6.539	57.404±6.597	116.00	219.51s/6.52hr
	3	502872	85.341±0.207	85.270±0.198	81.75	424.23s/9.56hr	507252	55.489±7.863	55.736±8.024	66.00	319.98s/5.79hr
	8	581716	Diverged	Diverged	Diverged	Diverged	586788	Diverged	Diverged	Diverged	Diverged

GRAPH CLASSIFICATION

		MNIST					CIFAR10				
Model	L	#Param	Test Acc.±s.d.	Train Acc.±s.d.	#Epoch	Epoch/Total	#Param	Test Acc.±s.d.	Train Acc.±s.d.	#Epoch	Epoch/Total
MLP	4	104044	95.340±0.138	97.432±0.470	232.25	22.74s/1.48hr	104380	56.340±0.181	65.113±1.685	185.25	29.48s/1.53hr
GCN	4	101365	90.705±0.218	97.196±0.223	127.50	83.41s/2.99hr	101657	55.710±0.381	69.523±1.948	142.50	109.70s/4.39hr
GraphSage	4	104337	**97.312±0.097**	100.000±0.000	98.25	113.12s/3.13hr	104517	65.767±0.308	99.719±0.062	93.50	124.61s/3.29hr
MoNet	4	104049	90.805±0.032	96.609±0.440	146.25	93.19s/3.82hr	104229	54.655±0.518	65.911±2.515	141.50	97.13s/3.85hr
GAT	4	110400	95.535±0.205	99.994±0.008	104.75	42.26s/1.25hr	110704	64.223±0.455	89.114±0.499	103.75	55.27s/1.62hr
GatedGCN	4	104217	**97.340±0.143**	100.000±0.000	96.25	128.79s/3.50hr	104357	**67.312±0.311**	94.553±1.018	97.00	154.15s/4.22hr
GIN	4	105434	96.485±0.252	100.000±0.000	128.00	39.22s/1.41hr	105654	55.255±1.527	79.412±9.700	141.50	52.12s/2.07hr
RingGNN	2	105398	11.350±0.000	11.235±0.000	14.00	2945.69s/12.77hr	105165	19.300±16.108	19.556±16.397	13.50	3112.96s/13.00hr
	2	505182	91.860±0.449	92.169±0.505	16.25	2575.99s/12.63hr	504949	39.165±17.114	40.209±17.790	13.75	2998.24s/12.60hr
	8	506357	Diverged	Diverged	Diverged	Diverged	510439	Diverged	Diverged	Diverged	Diverged
3WLGNN	3	108024	95.075±0.961	95.830±1.338	27.75	1523.20s/12.40hr	108516	59.175±1.593	63.751±2.697	28.50	1506.29s/12.60hr
	3	501690	95.002±0.419	95.692±0.677	26.25	1608.73s/12.42hr	502770	58.043±2.512	61.574±3.575	20.00	2091.22s/12.55hr
	8	500816	Diverged	Diverged	Diverged	Diverged	501584	Diverged	Diverged	Diverged	Diverged

LINK PREDICTION

		TSP					COLLAB				
Model	L	#Param	Test F1±s.d.	Train F1±s.d.	#Epoch	Epoch/Total	#Param ($L=3$)	Test Hits±s.d.	Train Hits±s.d.	#Epoch	Epoch/Total
MLP	4	96956	0.544±0.001	0.544±0.001	164.25	50.15s/2.31hr	39441	20.350±2.168	29.807±3.360	147.50	2.09s/0.09hr
GCN	4	95702	0.630±0.001	0.631±0.001	261.00	152.89s/11.15hr	40479	50.422±1.131	92.112±0.991	122.50	351.05s/12.04hr
GraphSage	4	99263	0.665±0.003	0.669±0.003	266.00	157.26s/11.68hr	39856	51.618±0.690	99.949±0.052	152.75	277.93s/11.87hr
MoNet	4	99007	0.641±0.002	0.643±0.002	282.00	84.46s/6.65hr	39751	36.144±2.191	61.156±3.973	167.50	26.69s/1.26hr
GAT	4	96182	0.671±0.002	0.673±0.002	328.25	68.23s/6.25hr	42637	51.501±0.962	97.851±1.114	157.00	18.12s/0.80hr
GatedGCN	4	97858	0.791±0.003	0.793±0.003	159.00	218.20s/9.72hr	40965	52.635±1.168	96.103±1.876	95.00	453.47s/12.09hr
GatedGCN-PE				-			41889	**52.849±1.345**	96.165±0.453	94.75	452.75s/12.08hr
GatedGCN-E	4	97858	0.808±0.003	0.811±0.003	197.00	218.51s/12.04hr	40965	49.212±1.560	88.747±1.058	95.00	451.21s/12.03hr
GatedGCN-E	16	500770	**0.838±0.002**	0.850±0.001	53.00	807.23s/12.17hr			-		
GIN	4	99002	0.656±0.003	0.660±0.003	273.50	72.73s/5.56hr	39544	41.730±2.284	70.555±4.444	140.25	8.66s/0.34hr
RingGNN	2	106862	0.643±0.024	0.644±0.024	2.00	17850.52s/17.19hr	-	OOM			
	2	507938	0.704±0.003	0.705±0.003	3.00	12835.53s/16.08hr	-	OOM			
	8	506564	Diverged	Diverged	Diverged	Diverged	-	OOM			
3WLGNN	3	106366	0.694±0.073	0.695±0.073	2.00	17468.81s/16.59hr	-	OOM			
	3	506681	0.288±0.311	0.290±0.312	2.00	17190.17s/16.51hr	-	OOM			
	8	508832	OOM	OOM	OOM	OOM	-	OOM			
k-NN Heuristic		k =2	Test F1: 0.693								
Matrix Fact.	0			-			60546561	44.206±0.452	100.000±0.000	254.33	2.66s/0.21hr

RingGNN and 3WLGNN rely on dense tensors which leads to **OOM** on both GPU and CPU memory.

GRAPH REGRESSION – ZINC

Model	L	#Param	Test MAE±s.d.	Train MAE±s.d.	#Epoch	Epoch/Total
MLP	4	108975	0.706±0.006	0.644±0.005	116.75	1.01s/0.03hr
GCN	4	103077	0.459±0.006	0.343±0.011	196.25	2.89s/0.16hr
	16	505079	0.367±0.011	0.128±0.019	197.00	12.78s/0.71hr
GraphSage	4	94977	0.468±0.003	0.251±0.004	147.25	3.74s/0.15hr
	16	505341	0.398±0.002	0.081±0.009	145.50	16.61s/0.68hr
MoNet	4	106002	0.397±0.010	0.318±0.016	188.25	1.97s/0.10hr
	16	504013	0.292±0.006	0.093±0.014	171.75	10.82s/0.52hr
GAT	4	102385	0.475±0.007	0.317±0.006	137.50	2.93s/0.11hr
	16	531345	0.384±0.007	0.067±0.004	144.00	12.98s/0.53hr
GatedGCN	4	105735	0.435±0.011	0.287±0.014	173.50	5.76s/0.28hr
GatedGCN-E	4	105875	0.375±0.003	0.236±0.007	194.75	5.37s/0.29hr
	16	504309	0.282±0.015	0.074±0.016	166.75	20.50s/0.96hr
GatedGCN-E-PE	16	505011	**0.214±0.013**	0.067±0.019	185.00	10.70s/0.56hr
GIN	4	103079	0.387±0.015	0.319±0.015	153.25	2.29s/0.10hr
	16	509549	0.526±0.051	0.444±0.039	147.00	10.22s/0.42hr
RingGNN	2	97978	0.512±0.023	0.383±0.020	90.25	327.65s/8.32hr
RingGNN-E	2	104403	0.363±0.026	0.243±0.025	95.00	366.29s/9.76hr
	2	527283	0.353±0.019	0.236±0.019	79.75	293.94s/6.63hr
	8	510305	Diverged	Diverged	Diverged	Diverged
3WLGNN	3	102150	0.407±0.028	0.272±0.037	111.25	286.23s/8.88hr
3WLGNN-E	3	103098	0.256±0.054	0.140±0.044	117.25	334.69s/10.90hr
	3	507603	0.303±0.068	0.173±0.041	120.25	329.49s/11.08hr
	8	582824	0.303±0.057	0.246±0.043	52.50	811.27s/12.15hr

Evaluation Metrics: (higher is better, except for ZINC)
- CLUSTER, PATTERN use weighted accuracy w.r.t. the class sizes.
- MNIST, CIFAR10 use multi-label classification accuracy.
- TSP uses binary F1 score for the positive edges.
- COLLAB uses Hits@50 via the evaluator provided by OGB [?].
- ZINC uses mean absolute regression error.

Notation:
- Models with the suffix **-E** use input edge features to initialize edge representations (ZINC: bond type, TSP: Euclidean distance, COLLAB: collaboration frequency and year).
- Models with the suffix **-PE** use Laplacian Eigenvectors as node positional encodings, with dimension 8 for ZINC, 2 for PATTERN and 20 for others.
- Results denoted by **Diverged** indicate unstable and divergent runs across all 4 seeds and initial learning rate values $\{10^{-3}, 10^{-4}, 10^{-5}\}$.
- Results denoted by **OOM** indicate runs which throw out of memory errors on our hardware configuration.

Extended training:
- For TSP, RingGNN/3WLGNN with 100K parameters achieved 0.733±0.020 and 0.649±0.051 respectively after 48hr of training.

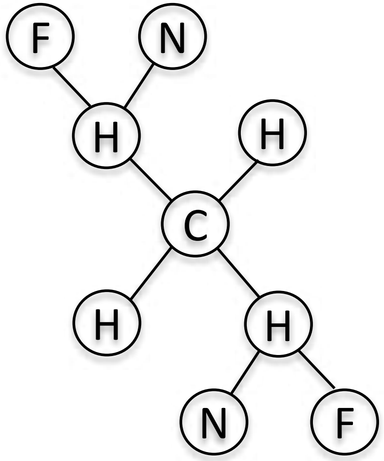
[1] Bahdanau, Cho, Bengio, Neural machine translation by jointly learning to align and translate, 2015
[2] Velickovic, Cucurull, Casanova, Romero, Lio, Bengio, Graph attention networks, 2017
[3] Bresson, Laurent, Residual gated graph convnets, 2017
[4] Abu-El-Haija, Perozzi, Al-Rfou, Alemi, Watch your step: Learning node embeddings via graph attention, 2018
[5] Puny, Ben-Hamu, Lipman, From Graph Low-Rank Global Attention to 2-FWL Approximation, 2020

Outline

- Motivation
- Message-Passing GCNs
- Weisfeiler-Lehman GNNs
- Graph-Agnostic GNNs
- Datasets
- Infrastructure and Experimental Setting
- Benchmarking Results
- Laplacian Positional Encodings
- Link Prediction with Edge Representation
- Conclusion

Structural (GCNs) vs Positional Encodings

- GCNs like GINs[1] are not able to differentiate isomorphic nodes, i.e. nodes with the same neighborhood structure :
  - All F atoms have the same representation.
  - All N atoms have the same representation.
  - Two H atoms have the same representation.
  - Two H atoms have the same representation.

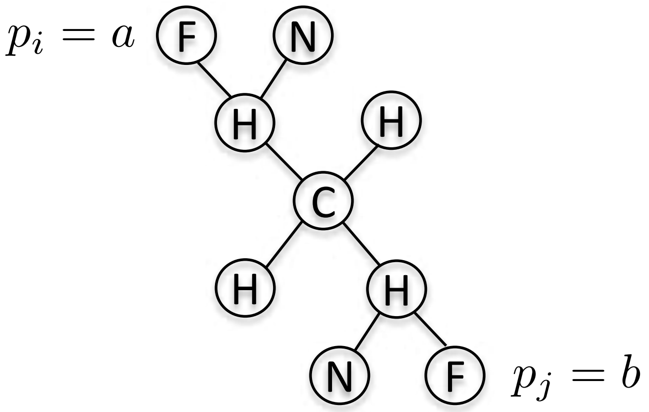


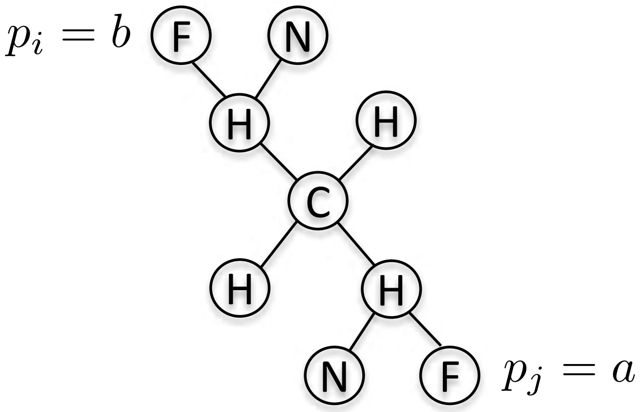
- This is a limitation of the expressivity of GCNs.
  - Can we break this structural symmetry ?
  - This can be done either by
    - Higher-order WL-GNNs, but require $O(n^2)/O(n^3)$.
    - Positional encodings of nodes with $O(n)$.

[1] Xu, Hu, Leskovec, Jegelka, How powerful are graph neural networks?, 2019

Graph Positional Encodings

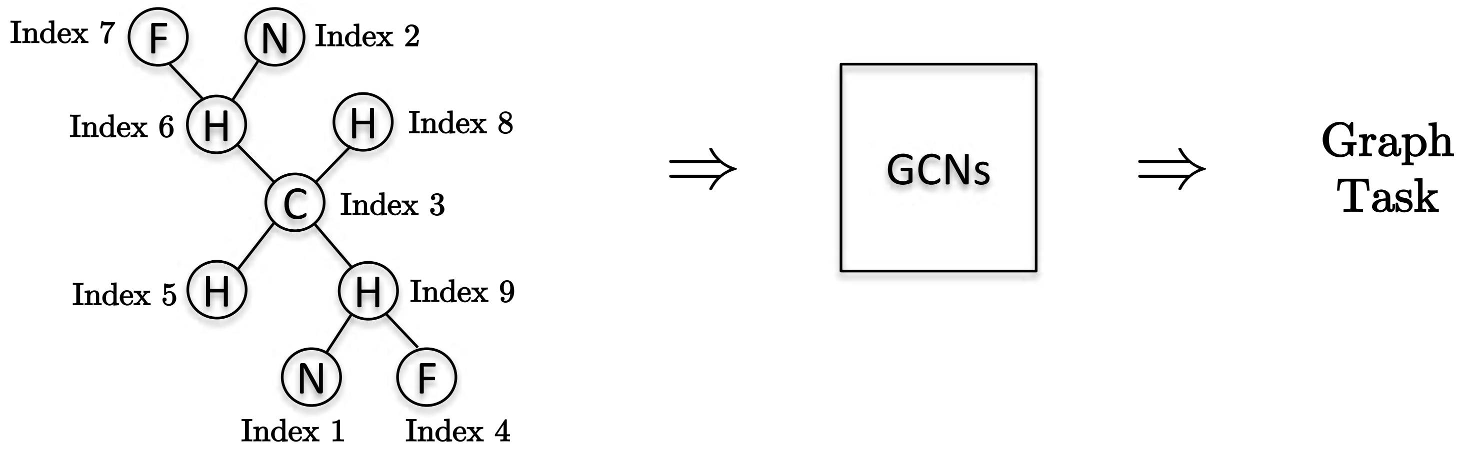
- Properties of PEs :
  - Unique representation for each node.
  - Distance-sensitive : Nodes far apart on the graph should have different positional features whereas nodes nearby have similar positional features.
- Graph symmetries prevent assigning canonical representation of PEs.
  - if nodes i and j are structurally symmetric and we have PEs $p_i{=}a,\ p_j{=}b$, then it is also possible to arbitrary choose $p_i{=}b,\ p_j{=}a$.
  - PEs are always arbitrary up to the number of graph symmetries.





Index Positional Encodings

- Simplest PEs are (arbitrary) ordering to the nodes, among $n!$ possible orderings.
- Theorem[1]: GCNs s.a. GINs[2] are more expressive than the *1*-WL test when considering (one-hot encoding of) node indices as PE features.
- During training, orderings are uniformly sampled from the $n!$ possible choices in order for the network to learn to be independent to these arbitrary choices.



[1] Murphy, Srinivasan, Rao, Ribeiro, Relational pooling for graph representations, 2019
[2] Xu, Hu, Leskovec, Jegelka, How powerful are graph neural networks?, 2019

Laplacian Positional Encodings

- Laplacian PEs :
  - Eigen-decomposition : $\Delta = \mathrm{I} - D^{-1/2} A D^{-1/2} = U^T \Lambda U$
  - Hybrid positional and structural encodings, invariant by index permutation.
  - Unique and distance-sensitive : two nodes far away on a graph have large PEs distance, and inversely (two one-encoding vectors of different indices are equally distant).
- LapPEs have also natural symmetries with the arbitrary sign of eigenvectors.
  - The number of possible sign flips is 2^k, k being the number of eigenvectors.
  - In practice, we choose $k \ll n$, and therefore 2^k is much smaller than $n!$ (the number of possible ordering of the nodes).
  - During the training, sign of eigenvectors will be uniformly sampled at random between the 2^k possibilities.
- Lap PEs are graph generalizations of Transformer's PEs[2].

[1] Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, Polosukhin, Attention is all you need, 2017

Numerical Results

- **Results :** PEs with GatedGCN[1]. Performance reported on the test sets of CSL, ZINC, PATTERN, CLUSTER and COLLAB (higher is better, except for ZINC).

CSL(n=11, R=2)

CSL(n=11, R=3)

CSL(n=11, R=4)

	PE type	L	#Param	Test Acc.±s.d.	Train Acc.±s.d.	#Epochs	Epoch/Total
CSL	No PE	4	104007	10.000±0.000	10.000±0.000	54.00	0.58s/0.05hr
	EigVecs-20	4	105407	68.633±7.143	99.811±0.232	107.16	0.59s/0.09hr
	Rand sign(EigVecs)	4	105407	**99.767±0.394**	99.689±0.550	188.76	0.59s/0.16hr
	Abs(EigVecs)	4	105407	99.433±1.133	100.000±0.000	143.64	0.60s/0.12hr
	Fixed node ordering	4	106807	10.533±4.469	76.056±14.136	60.56	0.59s/0.05hr
	Rand node ordering	4	106807	11.133±2.571	10.944±2.106	91.60	0.60s/0.08hr
PATTERN	No PE	16	502223	85.605±0.105	85.999±0.145	62.00	646.03s/11.36hr
	EigVecs-2	16	505421	86.029±0.085	86.955±0.227	65.00	645.36s/11.94hr
	Rand sign(EigVecs)	16	502457	**86.508±0.085**	86.801±0.133	65.75	647.94s/12.08hr
	Abs(EigVecs)	16	505421	86.393±0.037	87.011±0.172	62.00	645.90s/11.41hr
	Fixed node ordering	16	516887	80.133±0.202	98.416±0.141	45.00	643.23s/8.27hr
	Rand node ordering	16	516887	85.767±0.044	85.998±0.063	64.50	645.09s/11.79hr
CLUSTER	No PE	16	502615	73.684±0.348	88.356±1.577	61.50	399.44s/6.97hr
	EigVecs-20	16	504253	75.520±0.395	89.332±1.297	49.75	400.50s/5.70hr
	Rand sign(EigVecs)	16	504253	**76.082±0.196**	88.919±0.720	57.75	399.66s/6.58hr
	Abs(EigVecs)	16	504253	73.796±0.234	91.125±1.248	58.75	398.97s/6.68hr
	Fixed node ordering	16	517435	69.232±0.265	92.298±0.712	51.00	400.40s/5.82hr
	Rand node ordering	16	517435	74.656±0.314	82.940±1.718	61.00	397.75s/6.88hr
COLLAB	No PE	3	40965	52.635±1.168	96.103±1.876	95.00	453.47s/12.09hr
	EigVecs-20	3	41889	52.326±0.678	96.700±1.296	95.00	452.40s/12.10hr
	Rand sign(EigVecs)	3	41889	**52.849±1.345**	96.165±0.453	94.75	452.75s/12.08hr
	Abs(EigVecs)	3	41889	51.419±1.109	95.984±1.157	95.00	451.36s/12.07hr
	PE type	L	**#Param**	**Test MAE±s.d.**	**Train MAE±s.d.**	**#Epochs**	**Epoch/Total**
ZINC	No PE	16	504153	0.354±0.012	0.095±0.012	165.25	10.52s/0.49hr
	EigVecs-8	16	505011	0.319±0.010	0.038±0.007	143.25	10.62s/0.43hr
	Rand sign(EigVecs)	16	505011	**0.214±0.013**	0.067±0.019	185.00	10.70s/0.56hr
	Abs(EigVecs)	16	505011	**0.214±0.009**	0.035±0.011	167.50	10.61s/0.50hr
	Fixed node ordering	16	507195	0.431±0.007	0.044±0.009	118.25	10.62s/0.35hr
	Rand node ordering	16	507195	0.321±0.015	0.177±0.015	184.75	10.55s/0.55hr

[1] Bresson, Laurent, Residual gated graph convnets, 2017

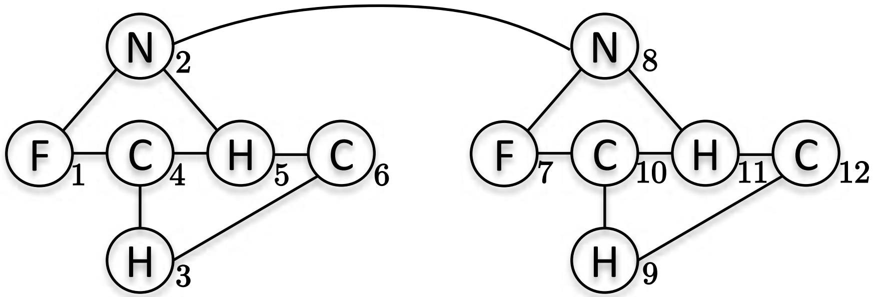
Outline

Link Prediction

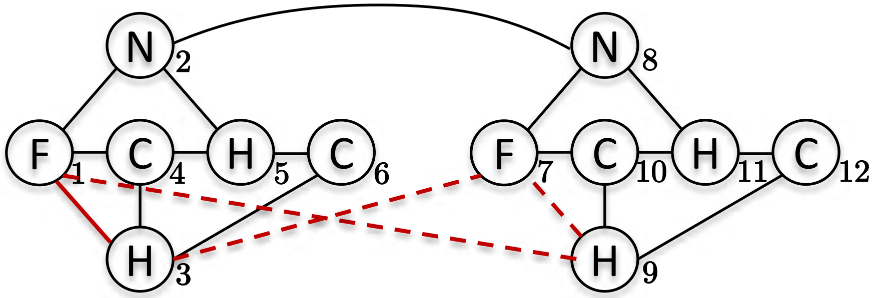
- GCNs may fail the link prediction task.
  - Apply any GCN to this molecule composed of two identical compounds :

We have

$$h_1 = h_7 \in \mathbb{R}^d$$
$$h_3 = h_9 \in \mathbb{R}^d$$

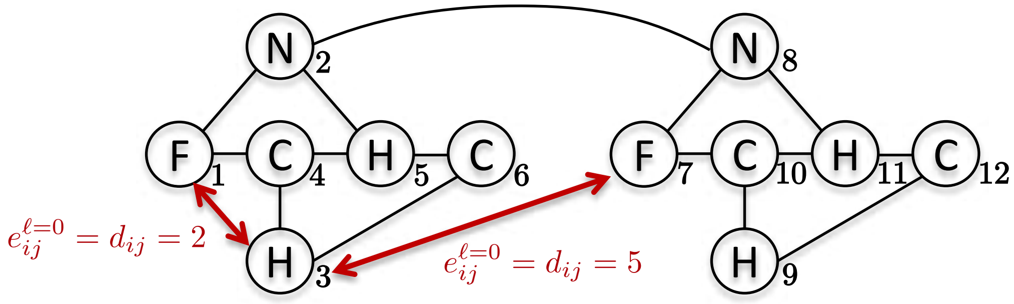


  - Perform link prediction (by transfer learning) :
    - Suppose there exist a bond ($\mathrm{F}_{(\text{item 1})}$, $\mathrm{H}_{(\text{item 3})}$).
    - Can we predict the link ($\mathrm{F}_{(\text{item 7})}$, $\mathrm{H}_{(\text{item 9})}$)? Yes, but the network will also predict (wrong) links between ($\mathrm{F}_{(\text{item 7})}$, $\mathrm{H}_{(\text{item 3})}$) and between ($\mathrm{F}_{(\text{item 1})}$, $\mathrm{H}_{(\text{item 9})}$).



Link Prediction with Edge Representation

- How to design expressive GCNs for the link prediction task ?
- Theorem[1] :
  - Link prediction is maximally expressive with a joint representation of nodes (intuitively this encodes the similarity/distance between pair of nodes).
  - This requires $O(n^2)$ edge representations.

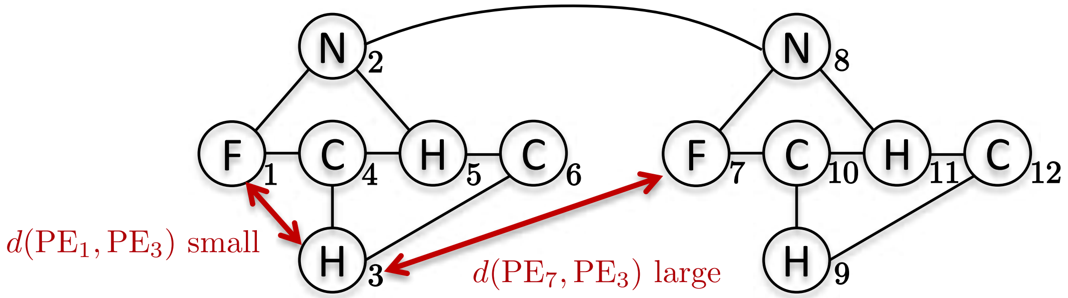


Input edge representation is
$e_{ij}^{l=0} = d_{ij}$ (e.g. shortest distances).

[1] Srinivasan, Ribeiro, On the Equivalence between Positional Node Embeddings and Structural Graph Representations, 2019

Link Prediction with Edge Representation

- Alternate solution :
  - Represent links as joint representations of nodes.
  - Add node positional encodings that are unique and distance-sensitive to differentiate links, like Laplacian PEs.
  - Complexity is $O(E)$, E is the number of edges.



Numerical Results

- We study the impact of edge representation by instantiating three variants :
  - No edge feature/representation : Isotropic models (like vanilla GCNs[1])

$$h_i^{\ell+1} = \sigma\Big(\sum_{j \in \mathcal{N}_i} W^\ell h_j^\ell\Big), \quad \text{identified by (E.Feat,E.Repr}= x,x)$$

  - Edge feature/no edge representation : Anisotropic models (like GAT[2])

$$h_i^{\ell+1} = \sigma\Big(\sum_{j \in \mathcal{N}_i} f_{V^\ell}(h_i^\ell, h_j^\ell) \cdot W^\ell h_j^\ell\Big), \quad \text{with (E.Feat,E.Repr}=\checkmark, x)$$

  - Edge feature and representation : Anisotropic models (like GatedGCNs[3])

$$h_i^{\ell+1} = \sigma\Big(\sum_{j \in \mathcal{N}_i} e_{ij}^\ell \cdot W^\ell h_j^\ell\Big), \; e_{ij}^{\ell+1} = f_{V^\ell}\big(h_i^\ell, h_j^\ell, e_{ij}^\ell\big), \quad \text{with (E.Feat,E.Repr}=\checkmark,\checkmark)$$

	Model	E.Feat.	E.Repr.	L	#Param	Test Acc.±s.d.	Train Acc.±s.d.	#Epochs	Epoch/Total
TSP	GatedGCN	x	x	4	99026	0.646±0.002	0.648±0.002	197.50	150.83s/8.34hr
		✓	x	4	98174	0.757±0.009	0.760±0.009	218.25	197.80s/12.06hr
		✓	✓	4	97858	0.791±0.003	0.793±0.003	159.00	218.20s/9.72hr
	GatedGCN-E	✓	✓	4	97858	**0.808±0.003**	0.811±0.003	197.00	218.51s/12.04hr
	GAT	x	x	4	95462	0.643±0.001	0.644±0.001	132.75	325.22s/12.10hr
		✓	x	4	96182	0.671±0.002	0.673±0.002	328.25	68.23s/6.25hr
		✓	✓	4	96762	0.748±0.022	0.749±0.022	93.00	462.22s/12.10hr
	GAT-E	✓	✓	4	96762	0.782±0.006	0.783±0.006	98.00	438.37s/12.11hr
COLLAB	GatedGCN	x	x	3	26593	35.989±1.549	60.586±4.251	148.00	263.62s/10.90h
		✓	x	3	26715	50.668±0.291	96.128±0.576	172.00	384.39s/18.44hr
		✓	✓	3	27055	**51.537±1.038**	96.524±1.704	188.67	376.67s/19.85hr
	GatedGCN-E	✓	✓	3	27055	47.212±2.016	85.801±0.984	156.67	377.04s/16.49hr
	GAT	x	x	3	28201	41.141±0.701	70.344±1.837	153.50	371.50s/15.97hr
		✓	x	3	28561	50.662±0.687	96.085±0.499	174.50	403.52s/19.69hr
		✓	✓	3	26676	49.674±0.105	92.665±0.719	201.00	349.19s/19.59hr
	GAT-E	✓	✓	3	26676	44.989±1.395	82.230±4.941	120.67	328.29s/11.10hr

[1] Kipf, Welling, Semi-supervised classification with graph convolutional networks, 2017
[2] Velickovic, Cucurull, Casanova, Romero, Lio, Bengio, Graph attention networks, 2017
[3] Bresson, Laurent, Residual gated graph convnets, 2017

Outline

- Motivation
- Message-Passing GCNs
- Weisfeiler-Lehman GNNs
- Graph-Agnostic GNNs
- Datasets
- Infrastructure and Experimental Setting
- Benchmarking Results
- Laplacian Positional Encodings
- Link Prediction with Edge Representation
- Conclusion

Conclusion

- Take-home messages[1] :
  - MP-GCNs outperformed WL-GNNs for GINs, 3WL-GNNs, RingGNNs on the 8 datasets used in this benchmark.
  - MP-GCNs benefit from graph sparsity, and universal building blocks w/ batch normalization and residual connection.
  - Anisotropic mechanism improve (isotropic) GCNs.
  - Laplacian eigenvectors improve index positional encodings.
  - Edge representation with positional encodings enhance link prediction.
- Recent works have focused on more efficient WL-inspired techniques[2,3,4,5,6].
- Benchmarking will bridge the gap between theory and practical performances.

[1] Dwivedi, Joshi, Laurent, Bengio, Bresson, Benchmarking graph neural networks, 2020, https://arxiv.org/pdf/2003.00982.pdf
[2] Bouritsas, Frasca, Zafeiriou, Bronstein, Improving Graph Neural Network Expressivity via Subgraph Isomorphism Counting, 2020
[3] Puny, Ben-Hamu, Lipman, From Graph Low-Rank Global Attention to 2-FWL Approximation, 2020
[4] Morris, Rattan, Mutzel, Weisfeiler and Leman go sparse: Towards scalable higher-order graph embeddings, 2020
[5] Corso, Cavalleri, Beaini, Lio, and Velickovic, Principal neighbourhood aggregation for graph nets, 2020
[6] Vignac, Loukas, Frossard, Building powerful and equivariant graph neural networks with message-passing, 2020

Workshop Announcement

https://www.ipam.ucla.edu/programs/workshops/deep-learning-and-combinatorial-optimization

ORGANIZING COMMITTEE

Peter Battaglia (DeepMind Technologies)
Xavier Bresson (Nanyang Technological University, Singapore)
Stefanie Jegelka (Massachusetts Institute of Technology)
Yann LeCun (New York University, Canadian Institute for Advanced Research)
Andrea Lodi (École Polytechnique de Montréal)
Stanley Osher (University of California, Los Angeles (UCLA), Mathematics)
Oriol Vinyals (DeepMind Technologies)
Max Welling (University of Amsterdam)

Speaker List

Shipra Agrawal (Columbia University, Computer Science)
Sanjeev Arora (Princeton University)
Peter Battaglia (DeepMind Technologies)
Xavier Bresson (Nanyang Technological University, Singapore)
Joan Bruna (New York University)
Laurent Charlin (HEC Montréal)
Kyle Cranmer (New York University)
Sanjeeb Dash (IBM Watson Research Center)
Santanu Dey (Georgia Institute of Technology, School of Industrial and Systems Engineering)
Bistra Dilkina (University of Southern California (USC))
Tina Eliassi-Rad (Northeastern University, Computer Science & Network Science)
Emma Frejinger (University of Montreal)
Maxime Gasse (École Polytechnique de Montréal)
Stefano Gualandi (Università di Pavia)
Oktay Gunluk (Cornell University)
Joey Huchette (Rice University)
Stefanie Jegelka (Massachusetts Institute of Technology)
Ron Kimmel (Technion - Israel Institute of Technology, Intel Perceptual Computing)
Zico Kolter (Carnegie Mellon University)
Vladlen Koltun (Intel Corporation)
Wouter Kool (University of Amsterdam)
Andrea Lodi (École Polytechnique de Montréal)
Azalia Mirhoseini (Google Inc.)
Stanley Osher (University of California, Los Angeles (UCLA), Mathematics)
Sebastian Pokutta (Konrad-Zuse-Zentrum für Informationstechnik (ZIB), Department of Mathematics)
Louis-Martin Rousseau (École Polytechnique de Montréal)
Thiago Serra (Bucknell University)
Le Song (Georgia Institute of Technology)
Petar Velickovic (DeepMind Technologies)
Oriol Vinyals (DeepMind Technologies)
Ellen Vitercik (Carnegie Mellon University)

Collaborators


Y. Bengio
MILA


M. Bronstein
Imperial/Twitter


F. Monti
Twitter


C. Joshi
NTU


V.P. Dwivedi
NTU


Y.Y. Leow
NTU


T. Laurent
LMU


A. Szlam
Facebook AI


R. Levie
TAU


M. Defferrard
EPFL


P. Vandergheynst
EPFL


P. Hagmann
UNIL

Thank you

Xavier Bresson

xbresson@ntu.edu.sg

http://www.ntu.edu.sg/home/xbresson

https://github.com/xbresson

https://twitter.com/xbresson

https://www.facebook.com/xavier.bresson.1

https://www.linkedin.com/in/xavier-bresson-738585b