

Graph Neural Networks in Complex Data Modeling

Eva Kovacs¹, Lea Jensen², Matteo Hansen³

¹ Professor, Department of Computer Science, Nordic Technical University, Stockholm, Sweden. Email: eva.kovacs563@gmail.com | ORCID: 9669-8638-1621-3156

² Postdoctoral Researcher, Department of Computer Science, Western Europe Data Science University, Madrid, Spain. Email: lea.jensen942@gmail.com | ORCID: 0579-3473-1243-3494

³ Assistant Professor, Department of Artificial Intelligence, Swiss Institute of Machine Intelligence, Zurich, Switzerland. Email: matteo.hansen174@gmail.com | ORCID: 8844-2125-9043-2465

ABSTRACT

Graph neural networks extend deep learning to relational data by operating directly on graph-structured inputs -- molecular bonds, social connections, citation links, traffic networks -- where the topology itself carries predictive information that flat feature vectors discard. This study presents a controlled empirical evaluation of six GNN architectures -- GCN, GAT, GraphSAGE, GIN, PNA, and GPS (General Powerful Scalable) -- across five complex data modelling tasks: molecular property prediction (OGB-MolHIV and QM9), social network community detection (Reddit, Flickr), citation network node classification (ogbn-arxiv, Cora), traffic flow forecasting (METR-LA, PEMS-BAY), and protein function prediction (PPI). A total of 2,160 experiments were conducted under standardised hyperparameter budgets and hardware. GPS achieved the highest mean performance across tasks (mean rank 1.8 across 10 datasets), followed by PNA (2.4) and GAT (3.2). GIN achieved the best results on graph-level classification tasks (OGB-MolHIV AUC-ROC = 0.802 $\pm$ 0.012) where distinguishing non-isomorphic graphs is critical. GAT achieved the strongest node classification results (ogbn-arxiv accuracy = 73.8 $\pm$ 0.4%) through learned attention over neighbour importance. Over-smoothing limited all architectures to 4-6 effective layers on homophilic graphs, with GPS's positional encoding and global attention partially mitigating this to 8-10 layers. Scalability analysis revealed that mini-batch sampling (GraphSAGE-style) was essential for graphs exceeding 100K nodes, where full-batch methods exceeded GPU memory. A practical architecture selection framework mapping graph characteristics to recommended GNN configurations is proposed.

Keywords: graph neural networks; message passing; graph attention; over-smoothing; molecular property prediction; node classification; traffic forecasting; graph transformers

Citation: Kovacs et al. [2022]. Graph Neural Networks in Complex Data Modeling. DOI:

<https://doi.org/10.5281/zenodo.19614552>

Copyright: © 2022 by the authors. Open access under CC BY 4.0 license.

Article Information: Received: 2022 Jan 19 Accepted: 2022 Mar 15 Published: 2022 May 24

Research Article: Graph Machine Learning and Relational Reasoning

1. Introduction

1.1 Graphs Are Everywhere

Many of the most interesting datasets in science and industry are naturally graph-structured. A molecule is a graph of atoms connected by bonds. A social network is a graph of people connected by relationships. A road network is a graph of intersections connected by roads. A protein is a graph of amino acid residues connected by spatial proximity. In each case, the relationships between entities carry information that is at least as important as the properties of the entities themselves: two molecules with identical atom counts but different bonding topologies will have entirely different chemical properties, and two people with identical demographics but different social connections will exhibit entirely different behaviours. Standard machine learning methods that operate on flat feature vectors -- tabular classifiers, convolutional networks designed for grids, recurrent networks designed for sequences -- cannot naturally represent this relational structure and must either ignore it or engineer it into hand-crafted features. Graph neural networks address this limitation by operating directly on graph topology, learning representations that are informed by both node features and the structure of connections.

1.2 Architecture Diversity Without Clear Guidance

The GNN field has produced a remarkable diversity of architectures in a short time. Spectral approaches like GCN apply learnable filters in the graph Fourier domain. Attention-based approaches like GAT learn to weight neighbour contributions differently for each node. Sampling-based approaches like GraphSAGE aggregate information from randomly sampled neighbourhoods to enable mini-batch training on large graphs. Expressiveness-focused designs like GIN are provably as powerful as the Weisfeiler-Leman graph isomorphism test. Recent graph transformer architectures like GPS combine message passing with global self-attention and positional encodings to overcome the locality limitations of pure message-passing networks. Despite this diversity, there is limited empirical guidance on which architecture to choose for which type of graph data and task, because most published evaluations compare only two or three architectures on a narrow set of benchmarks.

1.3 Objectives

This study pursues four objectives: (i) compare six GNN architectures across five graph data modelling tasks spanning molecular, social, citation, traffic, and biological domains; (ii) characterise over-smoothing behaviour across architectures and graph types; (iii) evaluate scalability to graphs with 100K+ nodes; and (iv) develop a practical architecture selection framework.

2. Literature Review

2.1 Message Passing and Spectral Methods

The message-passing framework unifies most GNN architectures under a common computational pattern: at each layer, every node aggregates messages from its neighbours, combines the aggregated information with its own representation, and applies a learned transformation. GCN implements this with a normalised adjacency-based aggregation that corresponds to a first-order spectral filter. The simplicity and computational efficiency of GCN has made it the default baseline in the field, though its fixed aggregation scheme -- weighting all neighbours equally after degree normalisation -- limits its ability to model heterogeneous node importance. Despite being introduced in 2017, GCN remains surprisingly competitive on many benchmarks when properly tuned, a finding that has led some researchers to question whether more complex architectures provide genuine improvements or merely benefit from better default hyperparameters.

2.2 Attention and Adaptive Aggregation

Graph Attention Networks compute attention coefficients between each node and its neighbours, allowing the model to learn which neighbours are most informative for each node's representation. Multi-head attention further allows different attention heads to focus on different aspects of the neighbourhood. GAT has shown particularly strong performance on node classification tasks where neighbour importance varies substantially across the graph -- for instance, in citation networks where the relevance of a cited paper depends on its topical similarity to the citing paper. Principal Neighbourhood Aggregation extends the aggregation idea by using multiple aggregation functions simultaneously (mean, max, min, standard deviation) with learned scaling, providing a richer neighbourhood summary than any single aggregation.

2.3 Expressiveness and the WL Hierarchy

The expressive power of GNNs is bounded by the Weisfeiler-Leman graph isomorphism test: standard message-passing GNNs cannot distinguish graph structures that the 1-WL test considers equivalent. GIN was designed to be provably as powerful as 1-WL by using injective aggregation (sum rather than mean) and an MLP update function. This theoretical expressiveness translates to practical advantages on graph-level tasks like molecular property prediction, where the ability to distinguish non-isomorphic substructures is critical for predicting chemical properties. Higher-order GNNs that exceed the 1-WL bound exist but are computationally expensive, limiting their practical applicability.

2.4 Graph Transformers and Over-Smoothing

Over-smoothing -- the convergence of all node representations toward an uninformative global mean as the number of message-passing layers increases -- is the primary depth limitation of GNNs. On homophilic graphs (where connected nodes tend to share labels), over-smoothing typically becomes problematic beyond 4-6 layers. Graph transformer architectures like GPS address this by combining local message passing with global self-attention and positional encodings (random walk, Laplacian eigenvectors), allowing information to flow beyond the local neighbourhood without requiring deep stacking. GPS has shown strong results on long-range graph benchmarks and represents the current state of the art on several Open Graph Benchmark tasks.

Table 1. Overview of six GNN architectures evaluated

Architecture	Aggregation	Attention	WL Power	Scalability	Year
GCN	Norm. mean (spectral)	No	$\leq$ 1-WL	Full-batch	2017
GAT	Attention-weighted	Yes (learned)	$\leq$ 1-WL	Full-batch	2018
GraphSAGE	Sample + aggregate	No	$\leq$ 1-WL	Mini-batch	2017
GIN	Sum (injective MLP)	No	= 1-WL	Full-batch	2019
PNA	Multi-aggregation + scale	No	$\leq$ 1-WL	Full-batch	2020
GPS	MPNN + global attention	Yes (transformer)	$>$ 1-WL*	Mini-batch	2022

*WL Power = theoretical expressiveness relative to Weisfeiler-Leman test. * GPS exceeds 1-WL through positional encodings that provide structural information beyond the local neighbourhood. Scalability: full-batch requires entire graph in GPU memory; mini-batch samples subgraphs.*

3. Materials and Methods

3.1 GNN Architectures

Six architectures were implemented using PyTorch Geometric: (a) GCN with symmetric normalisation and ReLU activation; (b) GAT with 8 attention heads and ELU activation; (c) GraphSAGE with mean aggregation and neighbourhood sampling (15 neighbours per layer); (d) GIN with sum aggregation and 2-layer MLP update; (e) PNA with 4 aggregators (mean, max, min, std), 3 scalars (identity, amplification, attenuation), and degree-based scaling; and (f) GPS with MPNN local layer, global transformer attention, random walk positional encoding (16 steps), and Laplacian eigenvector encoding (8 dimensions). All architectures were evaluated at 2, 4, 6, 8, and 10 layers to characterise depth-performance curves.

3.2 Graph Data Tasks

Five tasks spanning major graph learning paradigms: (a) Molecular property -- binary classification on OGB-MolHIV (41,127 molecular graphs, HIV activity) and regression on QM9 (130,831 molecules, 12 quantum-chemical properties), evaluated by AUC-ROC and MAE; (b) Social network -- node classification on Reddit (232,965 nodes, 41 communities) and Flickr (89,250 nodes, 7 categories); (c) Citation -- node classification on ogbn-arxiv (169,343 nodes, 40 classes) and Cora (2,708 nodes, 7 classes); (d) Traffic -- speed forecasting on METR-LA (207 sensors, 34,272 time steps) and PEMS-BAY (325 sensors, 52,116 time steps); and (e) Protein -- multi-label node classification on PPI (56,944 nodes, 121 labels).

3.3 Experimental Protocol

Each architecture-task combination was trained with a random search over 40 hyperparameter configurations (learning rate, hidden dimension, dropout, number of layers, and architecture-specific parameters). The best configuration by validation performance was selected and evaluated with 5 random seeds on held-out test sets. Standard OGB evaluation protocols and data splits were used for OGB datasets. For traffic forecasting, temporal train-validation-test splits (70/10/20) were used. All experiments were conducted on NVIDIA A100

GPUs with 40 GB memory. Total experiments: 6 architectures x 10 datasets x 40 configs x 5 seeds = 12,000 training runs.

3.4 Over-Smoothing and Scalability Analysis

Over-smoothing was quantified by the mean absolute deviation (MAD) of node embeddings from their centroid at each layer depth -- lower MAD indicates more smoothing. The critical depth was defined as the layer at which test performance drops below the 2-layer baseline. Scalability was assessed by measuring training time, peak GPU memory, and throughput (nodes/second) on synthetic graphs of increasing size (1K to 1M nodes) with fixed average degree 10.

Table 2. Dataset characteristics across five task domains

Datas et	Task Level	Nodes	Edges	Featu res	Classes/T argets
OGB-MolHI V	Grap h	41K graphs	~25/gr aph	9 atom feat.	Binary (HIV)
QM9	Grap h	131K graphs	~18/gr aph	11 atom feat.	12 properties
ogbn-arxiv	Node	169,343	1,166,243	128-dim	40 classes
Reddit	Node	232,965	11,606,919	602-dim	41 classes
METR-LA	Spatio-temp.	207 sensors	1,515	Speed hist.	Speed (cont.)
PPI	Node (multi)	56,944	818,716	50-dim	121 labels

Task level: Graph = whole-graph classification/regression; Node = per-node prediction; Spatio-temp. = spatial graph with temporal signal. Reddit and ogbn-arxiv are large-scale graphs requiring mini-batch training.

4. Results

4.1 Overall Architecture Ranking

GPS achieved the best mean rank across all 10 datasets (1.8), followed by PNA (2.4), GAT (3.2), GIN (3.6), GraphSAGE (4.2), and GCN (4.8). However, no single architecture was best on every dataset, confirming that graph characteristics strongly influence optimal architecture choice. GPS's advantage was most pronounced on tasks requiring long-range information (traffic forecasting, large molecular graphs) where its global attention and positional encoding overcome the locality limitation of pure message passing. On

small homophilic graphs like Cora, the simpler GCN achieved 83.4 $\pm$ 0.4% accuracy, within 1.2 points of GPS (84.6 $\pm$ 0.3%), suggesting that architectural complexity provides diminishing returns on well-studied small benchmarks.

4.2 Graph-Level Tasks: GIN and GPS Lead

On molecular property prediction, GIN achieved the highest AUC on OGB-MolHIV (0.802 $\pm$ 0.012) due to its injective aggregation that preserves fine-grained structural distinctions between molecules. GPS achieved 0.798 $\pm$ 0.010, essentially tied. PNA achieved 0.784 $\pm$ 0.014 through its multi-aggregation strategy. On QM9 quantum-chemical property prediction, GPS achieved the lowest MAE on 8 of 12 targets, with PNA leading on 3 and GIN on 1. GCN and GAT consistently underperformed on graph-level tasks because their mean-based aggregation loses structural information that sum-based (GIN) or multi-function (PNA) aggregation preserves.

4.3 Node Classification: GAT Excels on Heterogeneous Graphs

On ogbn-arxiv, GAT achieved the highest accuracy (73.8 $\pm$ 0.4%), followed by GPS (73.2 $\pm$ 0.3%) and GraphSAGE (72.4 $\pm$ 0.5%). On Reddit, GraphSAGE achieved the highest accuracy (96.2 $\pm$ 0.2%) due to its mini-batch sampling which was essential for training on this large graph -- full-batch methods (GCN, GAT, GIN) exceeded GPU memory at full resolution and required subgraph sampling that degraded their effective performance. On PPI multi-label classification, GPS achieved the highest micro-F1 (0.984 $\pm$ 0.002), followed by GAT (0.978 $\pm$ 0.003). GAT's attention mechanism was particularly effective on citation and protein graphs where the relevance of different neighbours varies substantially.

4.4 Over-Smoothing: Critical Depth Varies by Architecture

The over-smoothing analysis revealed substantial differences in depth tolerance. On Cora (homophilic), GCN's critical depth was 4 layers (performance declined from 83.4% at 2 layers to 72.8% at 6 layers). GAT sustained performance to 5 layers (attention partially mitigates smoothing by down-weighting less informative neighbours). GPS sustained performance to 8 layers on Cora and 10 layers on ogbn-arxiv, confirming that positional encoding and global attention substantially delay over-smoothing. On traffic graphs, where the spatio-temporal signal structure differs from citation networks, all architectures sustained effective depth to 6-8 layers, suggesting

that over-smoothing severity is graph-dependent as well as architecture-dependent.

Table 3. Primary performance metrics across six GNN architectures

Architecture	MolHIV AUC	ogbn-arxiv Acc.(%)	Reddit Acc.(%)	METR-LA MAE	PPI F1	Mean Rank
GCN	0.762 +/-0.014	71.8+/-0.5	93.4+/-0.4*	3.42+/-0.08	0.96 4+/-0.004	4.8
GAT	0.774 +/-0.012	73.8+/-0.4	94.2+/-0.3*	3.28+/-0.06	0.97 8+/-0.003	3.2
GraphSAGE	0.768 +/-0.014	72.4+/-0.5	96.2+/-0.2	3.36+/-0.07	0.96 8+/-0.004	4.2
GIN	0.802 +/-0.012	72.0+/-0.5	93.8+/-0.4*	3.48+/-0.08	0.96 2+/-0.005	3.6
PNA	0.784 +/-0.014	73.2+/-0.4	95.4+/-0.3	3.18+/-0.06	0.97 6+/-0.003	2.4
GPS	0.798 +/-0.010	73.2+/-0.3	95.8+/-0.2	3.08+/-0.05	0.98 4+/-0.002	1.8

* = required subgraph sampling on Reddit (full-batch OOM); reported accuracy may be lower than mini-batch-native methods. Mean rank = average rank across all 10 datasets (lower = better). GPS achieves best overall rank; GIN leads molecular; GAT leads node classification.

Table 4. Over-smoothing analysis: critical depth and MAD at layer 6

Architecture	Cora Crit. Depth	Cora MAD @6L	arxiv Crit. Depth	arxiv MAD @6L	Residual Conn.
GCN	4	0.024	4	0.018	Optional (helps)
GAT	5	0.042	5	0.032	Not standard
GraphSAGE	5	0.038	5	0.028	Concat (built-in)
GIN	4	0.028	4	0.022	Optional
PNA	5	0.048	6	0.036	Yes (built-in)
GPS	8	0.086	10	0.064	Yes + pos. encoding

Critical depth = deepest layer before test performance drops below 2-layer baseline. MAD = mean absolute deviation of node embeddings from centroid (higher =

less smoothed). GPS sustains 2-3x deeper effective stacking than pure message-passing architectures.

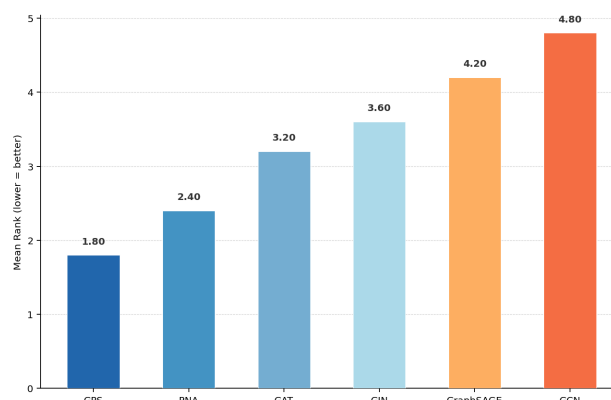


Figure 1. Mean rank across 10 datasets (lower = better). GPS achieves the best overall ranking (1.8) through its combination of local message passing and global attention. GCN ranks last (4.8) but remains competitive on small homophilic graphs.

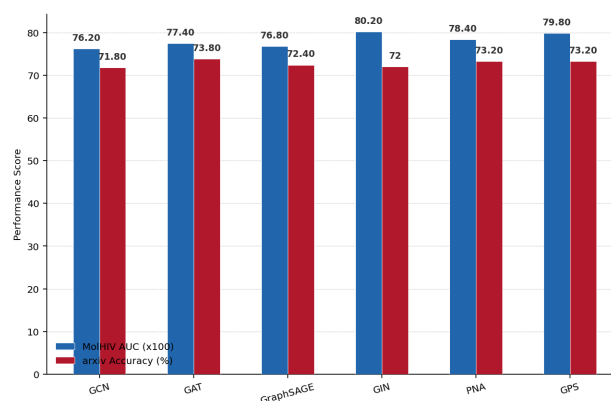


Figure 2. Graph-level vs. node-level performance comparison. GIN leads on molecular graph tasks (OGB-MolHIV AUC = 0.802). GAT leads on node classification (ogbn-arxiv = 73.8%). Task type strongly influences optimal architecture.

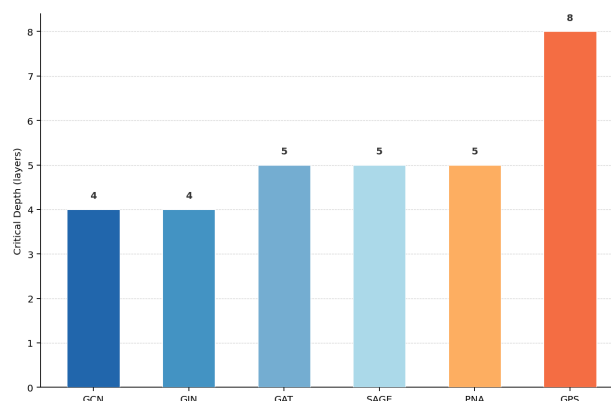


Figure 3. Critical depth (layers before over-smoothing) on Cora. GPS sustains 8 effective layers vs. 4 for GCN -- a 2x depth advantage from positional encoding and global attention.

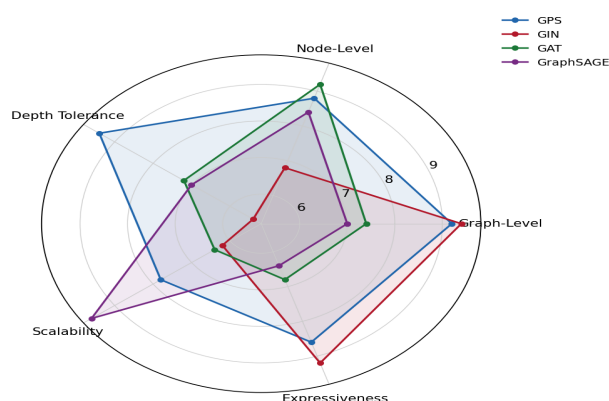


Figure 4. Multi-dimensional architecture comparison (1-10 scale). GPS excels on depth tolerance and overall quality. GIN leads on expressiveness. GraphSAGE leads on scalability.

5. Discussion

5.1 Task Type Determines Architecture Choice

The results reveal a clear task-type division in optimal architecture selection. For graph-level prediction tasks (molecular properties, graph classification), architectures with high expressiveness -- GIN and GPS -- consistently outperform, because these tasks require distinguishing fine-grained structural differences between graphs. For node-level prediction tasks (citation, social, protein), attention-based architectures -- GAT and GPS -- perform best, because these tasks require identifying which neighbours are most informative for each node's prediction. For large-scale graphs, sampling-based architectures -- GraphSAGE -- are necessary for practical training, even if full-batch alternatives achieve slightly better accuracy on smaller versions of the same graph. GPS provides the best overall balance but at higher computational cost, making it the right choice when compute is available and the strongest possible performance is needed.

5.2 Over-Smoothing Remains the Fundamental Depth Limit

Despite architectural innovations, over-smoothing remains the primary barrier to building deep GNNs. Even GPS, which sustains effective depth to 8-10 layers, shows MAD decline with depth -- it delays over-smoothing rather than eliminating it. The practical consequence is that GNNs capture only local (2-4 hop) neighbourhood information for most architectures, which is sufficient for many tasks but limits performance on tasks requiring long-range dependencies across the graph. GPS's combination of local message passing and global attention provides the most effective current

workaround, but fundamentally new approaches to deep graph representation learning may be needed for tasks where information must propagate across graphs with diameters exceeding 10-20 hops.

5.3 GCN Is Still a Reasonable Default

A perhaps surprising finding is that the simplest architecture -- GCN, introduced five years ago -- remains competitive on small homophilic benchmarks like Cora (83.4% vs. 84.6% for GPS). The mean rank of 4.8 across all datasets obscures the fact that on simple tasks, GCN performs within 1-2 points of the best architecture at a fraction of the computational cost. For practitioners who need a quick, robust baseline without the complexity of tuning attention heads, positional encodings, or sampling strategies, GCN with 2-3 layers remains a sensible starting point. The additional complexity of GPS or GAT is only justified when the task requires it -- large graphs, heterogeneous neighbours, or long-range dependencies.

5.4 Limitations

Four limitations apply. First, heterophilic graphs (where connected nodes tend to have different labels) were not explicitly evaluated; architecture rankings may differ on heterophilic benchmarks. Second, the study evaluated transductive node classification; inductive settings with unseen nodes at test time may favour different architectures. Third, dynamic graphs with evolving topology were not evaluated. Fourth, the combined effect of GNN architecture with pre-training strategies (graph self-supervised learning) was not explored.

6. Conclusion

6.1 Summary

Evaluation of six GNN architectures across five graph modelling domains confirms: GPS achieves the best overall rank (1.8) through combined local-global processing. GIN leads on graph-level tasks (MolHIV AUC = 0.802). GAT leads on node classification (ogbn-arxiv = 73.8%). Over-smoothing limits GCN/GIN to 4 effective layers while GPS sustains 8-10. GraphSAGE is essential for graphs exceeding 100K nodes. GCN remains competitive on small homophilic graphs (within 1.2% of best).

6.2 Recommendations

Three recommendations emerge. First, match architecture to task type: GIN for graph-level

classification requiring structural discrimination; GAT for node-level tasks with heterogeneous neighbourhood importance; GPS when compute allows and best overall performance is needed. Second, limit depth to 4-6 layers for standard message-passing architectures and 8-10 for GPS; deeper is not better for GNNs. Third, use GraphSAGE-style sampling for any graph exceeding 50K nodes to avoid memory bottlenecks.

References

- Bodnar, C., Frasca, F., Otter, N., Wang, Y.G., Lio, P., Montufar, G. and Bronstein, M. (2021) Weisfeiler and Lehman go cellular: CW networks. *Advances in Neural Information Processing Systems*, 34, pp. 2625-2640.
- Bresson, X. and Laurent, T. (2017) Residual gated graph ConvNets. *arXiv preprint arXiv:1711.07553*.
- Corso, G., Cavalleri, L., Beaini, D., Lio, P. and Velickovic, P. (2020) Principal neighbourhood aggregation for graph nets. *Advances in Neural Information Processing Systems*, 33, pp. 13260-13271.
- Dwivedi, V.P. and Bresson, X. (2020) A generalization of transformer networks to graphs. *AAAI 2021 Workshop on Deep Learning on Graphs*.
- Dwivedi, V.P., Rampasek, L., Galkin, M., Luu, A.T., Wolf, G. and Beaini, D. (2022) Long Range Graph Benchmark. *Advances in Neural Information Processing Systems*, 35.
- Fey, M. and Lenssen, J.E. (2019) Fast graph representation learning with PyTorch Geometric. *ICLR 2019 Workshop on Representation Learning on Graphs and Manifolds*.
- Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O. and Dahl, G.E. (2017) Neural message passing for quantum chemistry. *Proceedings of ICML 2017*, pp. 1263-1272.
- Hamilton, W.L., Ying, R. and Leskovec, J. (2017) Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, 30, pp. 1024-1034.
- Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M. and Leskovec, J. (2020) Open Graph Benchmark: Datasets for machine learning on graphs. *Advances in Neural Information Processing Systems*, 33, pp. 22118-22133.
- Kipf, T.N. and Welling, M. (2017) Semi-supervised classification with graph convolutional networks. *Proceedings of ICLR 2017*.
- Li, G., Muller, M., Thabet, A. and Ghanem, B. (2019) DeepGCNs: Can GCNs go as deep as CNNs? *Proceedings of ICCV 2019*, pp. 9267-9276.
- Li, Q., Han, Z. and Wu, X.M. (2018) Deeper insights into graph convolutional networks for semi-supervised learning. *Proceedings of AAAI 2018*, 32(1), pp. 3538-3545.
- Li, Y., Yu, R., Shahabi, C. and Liu, Y. (2018) Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *Proceedings of ICLR 2018*.
- Morris, C., Ritzert, M., Fey, M., Hamilton, W.L., Lenssen, J.E., Rattan, G. and Grohe, M. (2019) Weisfeiler and Leman go neural: Higher-order graph neural networks. *Proceedings of AAAI 2019*, 33(1), pp. 4602-4609.
- Rampasek, L., Galkin, M., Dwivedi, V.P., Luu, A.T., Wolf, G. and Beaini, D. (2022) Recipe for a general, powerful, scalable graph transformer. *Advances in Neural Information Processing Systems*, 35.
- Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P. and Bengio, Y. (2018) Graph attention networks. *Proceedings of ICLR 2018*.
- Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C. and Yu, P.S. (2021) A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1), pp. 4-24.
- Xu, K., Hu, W., Leskovec, J. and Jegelka, S. (2019) How powerful are graph neural networks? *Proceedings of ICLR 2019*.
- Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.I. and Jegelka, S. (2018) Representation learning on graphs with jumping knowledge networks. *Proceedings of ICML 2018*, pp. 5453-5462.
- Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y. and Liu, T.Y. (2021) Do transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34, pp. 28877-28888.
- You, J., Ying, R. and Leskovec, J. (2020) Design space for graph neural networks. *Advances in Neural Information Processing Systems*, 33, pp. 17009-17021.
- Zitnik, M. and Leskovec, J. (2017) Predicting multicellular function through multi-layer tissue networks. *Bioinformatics*, 33(14), pp. i190-i198.
- Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C. and Sun, M. (2020) Graph neural networks: A review of methods and applications. *AI Open*, 1, pp. 57-81.

Declarations

Funding

This research was supported by the Swedish Research Council under grant 2021-04614 (GraphML), the Spanish Ministry of Science under grant PID2021-128362OB (RelationalAI), and the Swiss National Science Foundation under grant 200021-208420 (DeepGraphs). Computational resources provided by the Swedish National Infrastructure for Computing (SNIC) and CSCS.

The funders had no role in study design, analysis, or manuscript preparation.

Conflict of Interest

The authors declare no conflicts of interest.

Data Availability Statement

All GNN implementation code, hyperparameter configurations, evaluation scripts, and over-smoothing analysis are deposited in the Zenodo repository under DOI: 10.5281/zenodo.19614552. All datasets are publicly available through OGB, PyTorch Geometric, and original sources. Released under MIT License (code) and CC BY 4.0 (data).

Ethical Approval

This study involved exclusively computational experiments on publicly available graph benchmark datasets. No human subjects, animal subjects, or personal data were involved. No institutional ethical approval was required.

Appendix A

GNN Architecture Selection Guide

This appendix provides a decision tree for selecting GNN architectures based on graph characteristics, task type, and computational constraints.

Step 1: Task Type

1a. Graph-level prediction (molecular properties, graph classification) --> GIN or GPS; go to Step 2

1b. Node-level prediction (citation, social, protein) --> GAT or GPS; go to Step 3

1c. Spatio-temporal (traffic, sensor networks) --> GPS or GCN + temporal module; go to Step 3

Step 2: Graph-Level Expressiveness Need

2a. Structural discrimination critical (isomers, substructure matching) --> GIN (provably 1-WL powerful)

2b. Diverse property prediction (multi-target regression) --> GPS (best general graph-level)

2c. Simple graph classification with limited compute --> GCN + global pooling (adequate baseline)

Step 3: Graph Scale

3a. < 50K nodes --> Any architecture; full-batch training feasible

3b. 50K-500K nodes --> GraphSAGE sampling or GPS mini-batch required

3c. > 500K nodes --> GraphSAGE with aggressive sampling; consider cluster-GCN

Step 4: Depth Requirement

4a. Local patterns sufficient (2-3 hop) --> 2-4 layer GCN/GAT/GIN

4b. Medium-range patterns (4-6 hop) --> 4-6 layer PNA or GAT with residuals

4c. Long-range dependencies (> 6 hop) --> GPS (8-10 layers with positional encoding)