Strategies for Pre-training Graph Neural Networks

Mehmet F. Demirel

April 15, 2021

Why?

- Make accurate predictions on test data that is distributionally different from training data.
- Scarce task-specific labels
- **Solution:** Pre-train on related task, and fine-tune it for task of interest.

Graphs

Success in vision and NLP, but few studies generalize pre-training to graph data.

Negative Transfer

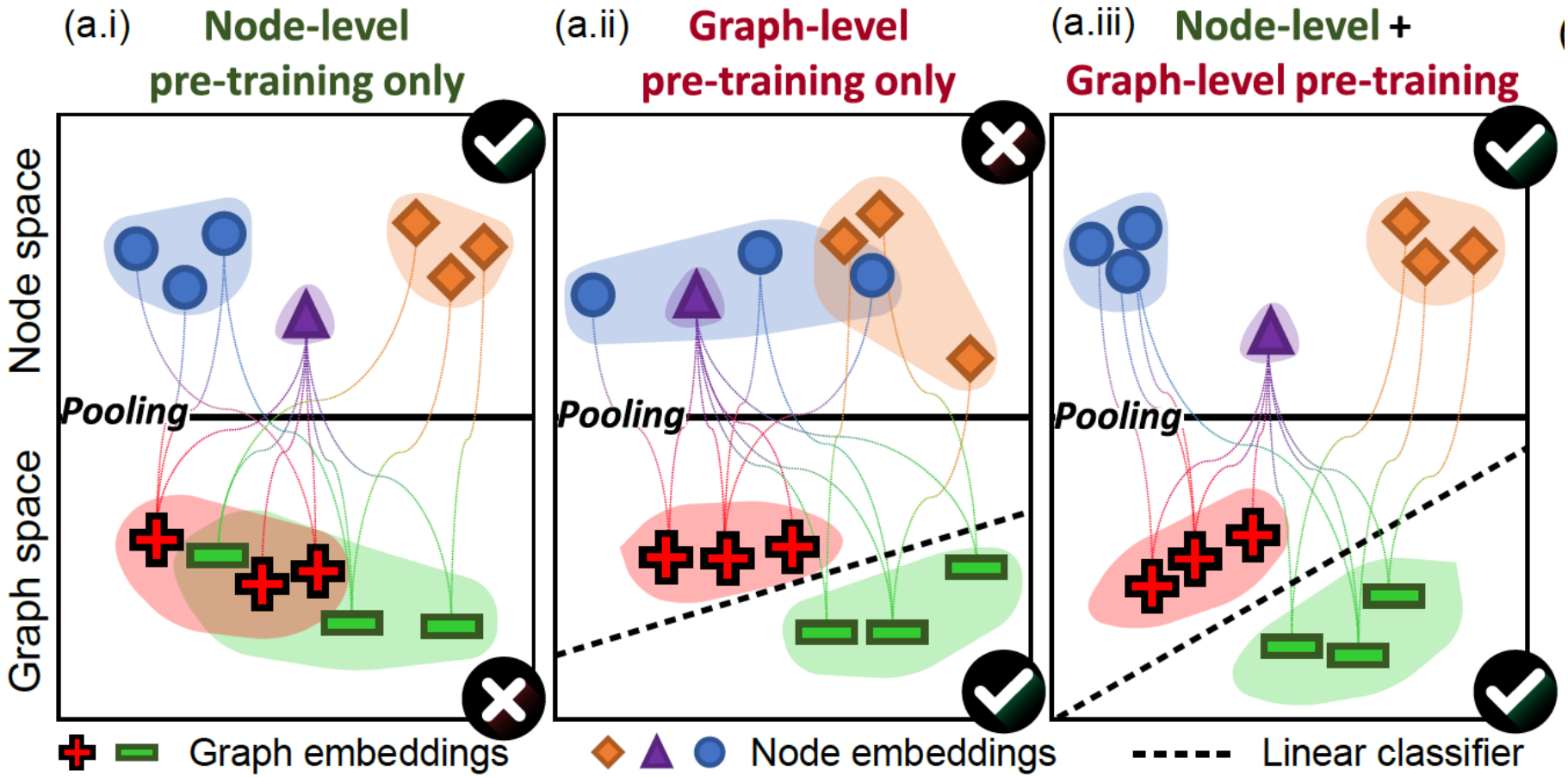
- Good transfer learning doesn't just depend the number of labeled pre-training data that is from the same domain as the task of interest.
- Domain expertise is needed.
- If pre-training data is not correlated with task of interest, the generalization after transfer can be harmed (i.e. *negative transfer*).

Method

- Overview
- Node-level Pre-training
  - Context Prediction
  - Attribute Masking
- Graph-level Pre-training
  - Supervised Graph-level Property Prediction

Method: Overview

Main idea: pre-train a GNN **both** at the node and graph level.

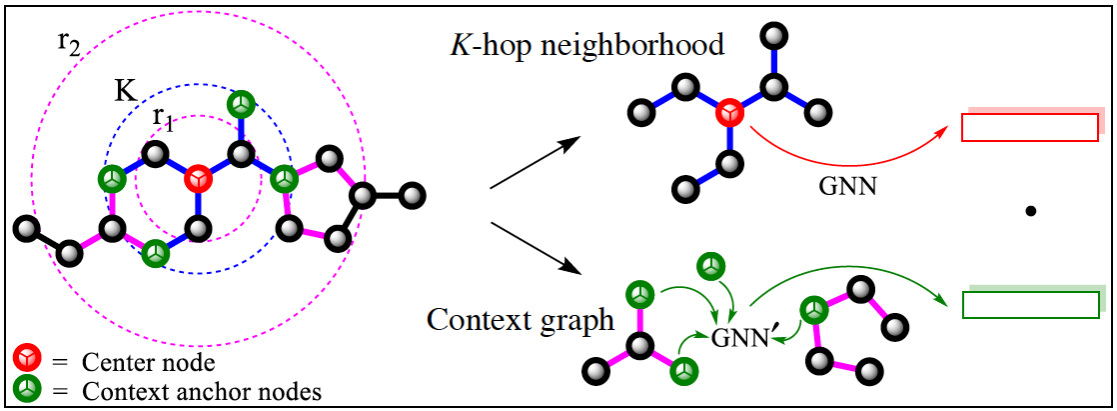


Method

- Overview
- Node-level Pre-training
  - Context Prediction
  - Attribute Masking
- Graph-level Pre-training
  - Supervised Graph-level Property Prediction

Method: Context Prediction

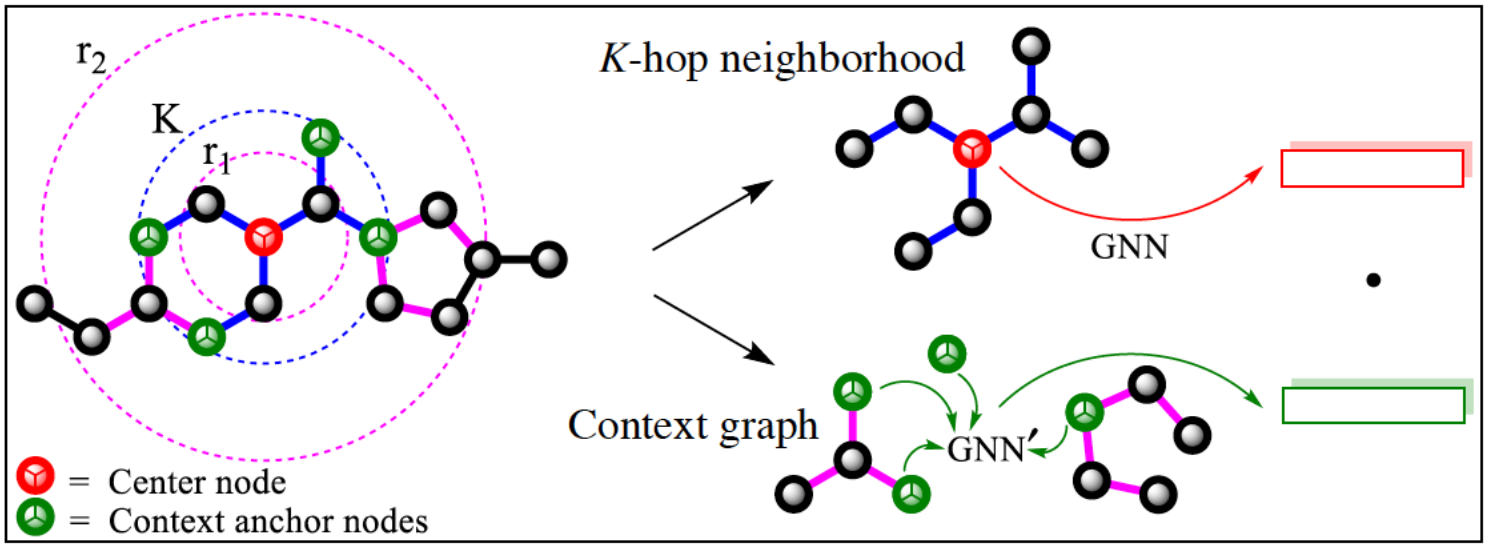
- K-**hop neighborhood of vertex** v: $h_v^{(K)}$—obtained by using a K-layer GNN (main GNN)
- **Context graph of vertex** v: The graph structure that surrounds v's neighborhood. It's a subgraph that lies between r_1-hops and r_2-hops away form v where $r_1 < K$.
- **Context anchor node:** Common in both neighborhood and context graph



Context Prediction

Method: Context Prediction

- Use an auxiliary GNN (**context GNN**) to obtain node embeddings in the context graph.
- Average the embeddings of *context anchor nodes* to get context embedding of vector v: c_v^G



Context Prediction

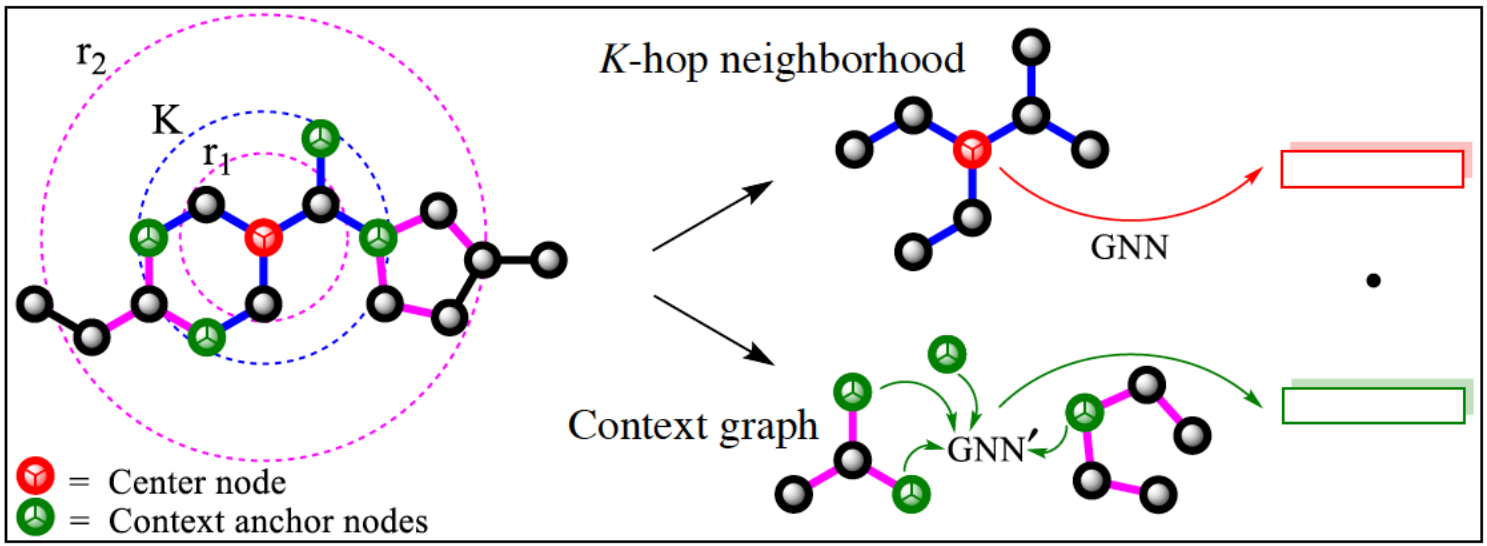
Method: Context Prediction

Negative sampling

- Train both GNN and GNN' with the objective of

$$\sigma(h_v^{(K)}, c_{v'}^{G'}) \approx \mathbf{1}_{\{v \text{ and } v' \text{ are the same nodes}\}}$$

- **Positive pair:** $v' = v, G' = G$.
- **Negative pair:** Randomly sample v' from randomly-chosen G'



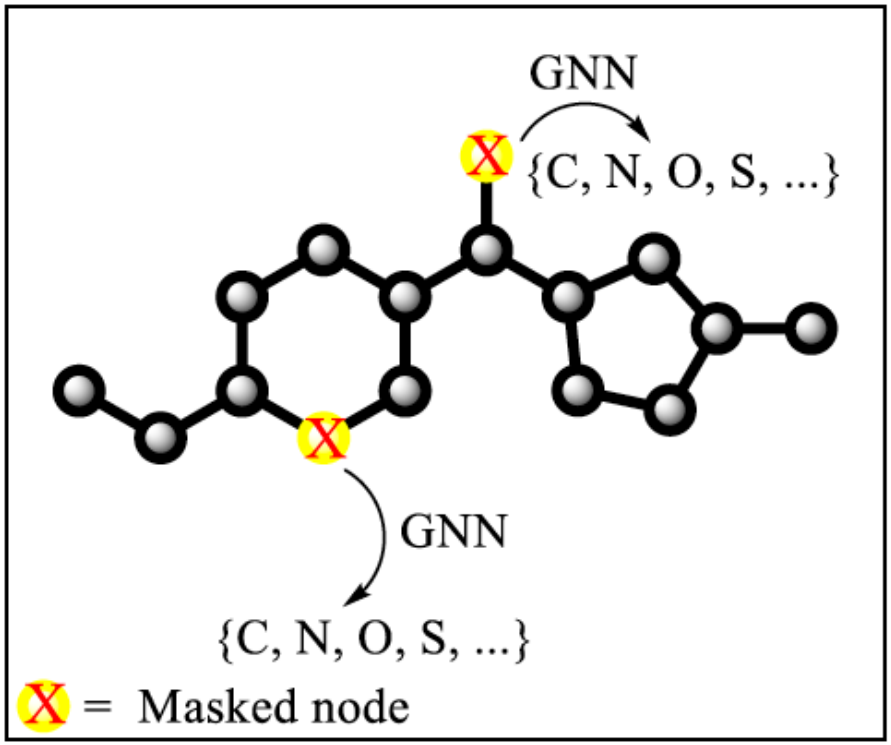
Context Prediction

Method

- Overview
- Node-level Pre-training
  - Context Prediction
  - Attribute Masking
- Graph-level Pre-training
  - Supervised Graph-level Property Prediction

Method: Attribute Masking

- Mask node/edge attributes, and let GNNs predict the masked attributes based on neighboring structure.
- Masking edge attributes



Context Prediction

Method

- Overview
- Node-level Pre-training
  - Context Prediction
  - Attribute Masking
- Graph-level Pre-training
  - Supervised Graph-level Property Prediction

- Graph-level representation h_G is directly used for fine-tuning on downstream prediction tasks. So, it is *important to directly encode domain-specific information into* h_G.
- Pre-train graph representation using graph-level multi-task supervised pre-training to jointly predict a diverse set of supervised labels of individual graphs.

- **Problem:** Only performing multi-task graph-level pre-training can fail to give transferable graph representations as some supervised pre-training tasks might be unrelated to the task of interest.
- **Solution?**: Pick relevant tasks (*costly*)
- **Solution!**: Apply node-level pre-training methods first before performing graph-level pre-training.

Final Method

- Pre-trained GNN and downstream linear classifier is fine-tuned in an end-to-end manner.

Experimental Results

Dataset		BBBP	Tox21	ToxCast	SIDER	ClinTox	MUV	HIV	BACE	**Average**
# Molecules		2039	7831	8575	1427	1478	93087	41127	1513	/
# Binary prediction tasks		1	12	617	27	2	17	1	1	/
Pre-training strategy		Out-of-distribution prediction (scaffold split)								
Graph-level	Node-level									
–	–	65.8 ±4.5	74.0 ±0.8	63.4 ±0.6	57.3 ±1.6	58.0 ±4.4	71.8 ±2.5	75.3 ±1.9	70.1 ±5.4	67.0
–	Infomax	**68.8 ±0.8**	75.3 ±0.5	62.7 ±0.4	58.4 ±0.8	69.9 ±3.0	75.3 ±2.5	76.0 ±0.7	75.9 ±1.6	70.3
–	EdgePred	67.3 ±2.4	76.0 ±0.6	64.1 ±0.6	60.4 ±0.7	64.1 ±3.7	74.1 ±2.1	76.3 ±1.0	79.9 ±0.9	70.3
–	AttrMasking	64.3 ±2.8	76.7 ±0.4	64.2 ±0.5	61.0 ±0.7	71.8 ±4.1	74.7 ±1.4	77.2 ±1.1	79.3 ±1.6	71.1
–	ContextPred	68.0 ±2.0	75.7 ±0.7	63.9 ±0.6	60.9 ±0.6	65.9 ±3.8	75.8 ±1.7	77.3 ±1.0	79.6 ±1.2	70.9
Supervised	–	68.3 ±0.7	77.0 ±0.3	64.4 ±0.4	62.1 ±0.5	57.2 ±2.5	79.4 ±1.3	74.4 ±1.2	76.9 ±1.0	70.0
Supervised	Infomax	68.0 ±1.8	77.8 ±0.3	64.9 ±0.7	60.9 ±0.6	**71.2 ±2.8**	**81.3 ±1.4**	77.8 ±0.9	80.1 ±0.9	72.8
Supervised	EdgePred	66.6 ±2.2	**78.3 ±0.3**	**66.5 ±0.3**	**63.3 ±0.9**	70.9 ±4.6	78.5 ±2.4	77.5 ±0.8	79.1 ±3.7	72.6
Supervised	AttrMasking	66.5 ±2.5	77.9 ±0.4	65.1 ±0.3	**63.9 ±0.9**	**73.7 ±2.8**	**81.2 ±1.9**	77.1 ±1.2	80.3 ±0.9	73.2
Supervised	ContextPred	**68.7 ±1.3**	**78.1 ±0.6**	65.7 ±0.6	62.7 ±0.8	**72.6 ±1.5**	**81.3 ±2.1**	**79.9 ±0.7**	**84.5 ±0.7**	**74.2**

Table 1: **Test ROC-AUC (%) performance on molecular prediction benchmarks using different pre-training strategies with GIN.** The rightmost column averages the mean of test performance across the 8 datasets. The best result for each dataset and comparable results (*i.e.*, results within one standard deviation from the best result) are bolded. The shaded cells indicate negative transfer, *i.e.*, ROC-AUC of a pre-trained model is worse than that of a non-pre-trained model. Notice that node- as well as graph-level pretraining are essential for good performance.

Some important observations

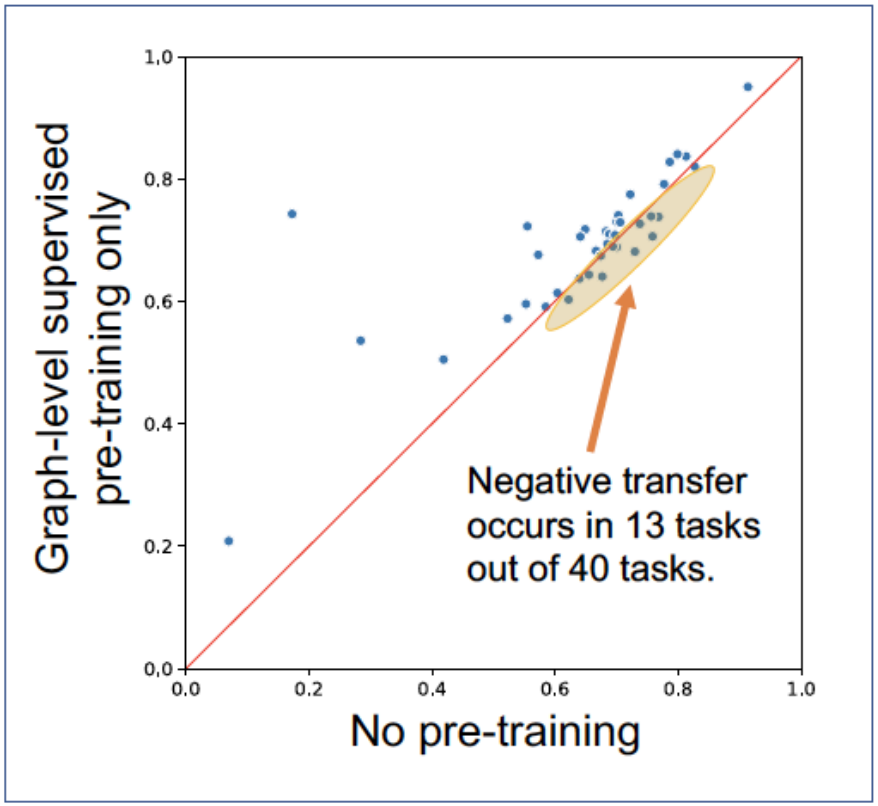
1. Using an expressive GNN model (GIN is the most expressive) is essential to benefit from pre-training. Otherwise, it might even be hurtful.

	Chemistry			Biology		
	Non-pre-trained	Pre-trained	Gain	Non-pre-trained	Pre-trained	Gain
GIN	67.0	**74.2**	**+7.2**	64.8 ± 1.0	**74.2 ± 1.5**	**+9.4**
GCN	**68.9**	72.2	+3.4	63.2 ± 1.0	70.9 ± 1.7	+7.7
GraphSAGE	68.3	70.3	+2.0	65.7 ± 1.2	68.5 ± 1.5	+2.8
GAT	66.8	60.3	-6.5	**68.2 ± 1.1**	67.8 ± 3.6	-0.4

Table 2: **Test ROC-AUC (%) performance of different GNN architectures with and without pre-training.** Without pre-training, the less expressive GNNs give slightly better performance than the most expressive GIN because of their smaller model complexity in a low data regime. However, with pre-training, the most expressive GIN is properly regularized and dominates the other architectures. For results split by chemistry datasets, see Table 4 in Appendix H. Pre-training strategy for chemistry data: Context Prediction + Graph-level supervised pre-training; pre-training strategy for biology data: Attribute Masking + Graph-level supervised pre-training.

Observations

2. Using only graph-level multi-task supervised pre-training gives limited performance gain and even yields negative transfer on many downstream tasks.



Observations

3. Using only node-level self-supervised pre-training gives limited performance improvement.

Pre-training strategy		Out-of-dist.
Graph-level	Node-level	(species split)
–	–	64.8 ±1.0
–	Infomax	64.1 ±1.5
–	EdgePred	65.7 ±1.3
–	ContextPred	65.2 ±1.6
–	AttrMasking	64.4 ±1.3

Observations

4. Models pre-trained on both node- and graph-level achieve orders-of-magnitude faster training and validation convergence compared to other models.

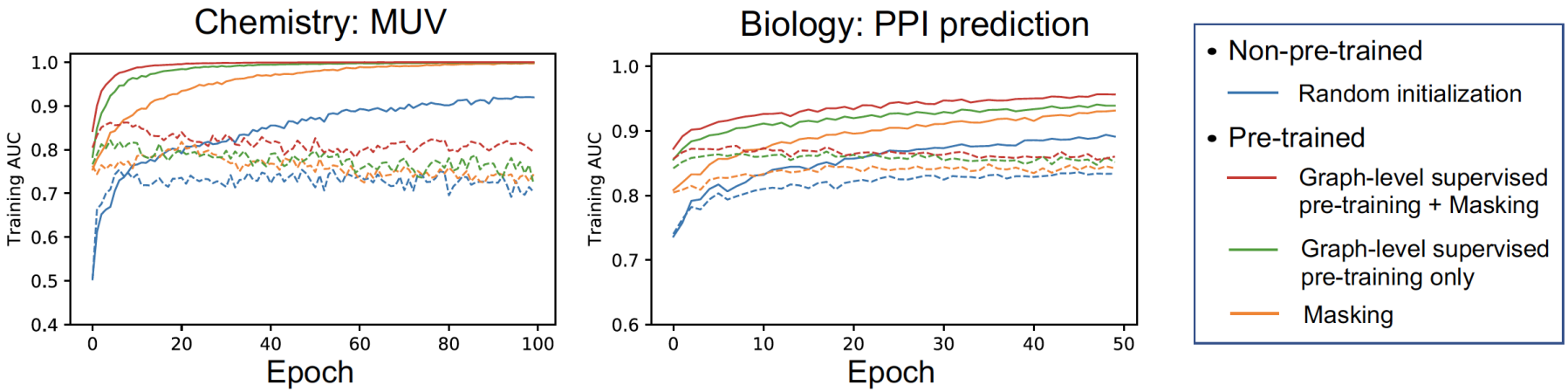


Figure 4: **Training and validation curves of different pre-training strategies on GINs.** Solid and dashed lines indicate training and validation curves, respectively.