UNIVERSITY OF CAMBRIDGE

Task-Agnostic Graph Neural Network Evaluation via Adversarial Collaboration

CST Part III Project Presentation

Victor Zhao

8 June 2022

Department of Computer Science and Technology

How to Evaluate a GNN?

- Common approach: train GNNs on some node/graph classification/regression tasks in some datasets, and then compare their performances on a leaderboard
- If model A's performance is 0.1 higher than model B, does that mean model A is better?
- Previous graph datasets: Cora, CiteSeer, PubMed, …
  …but they have soon become deprecated!
- Current widely-accepted GNN benchmarks: OGB, ZINC, …
  …can they be guaranteed ever-lasting?
- **We need a task-agnostic GNN evaluation method to fully exploit the datasets**

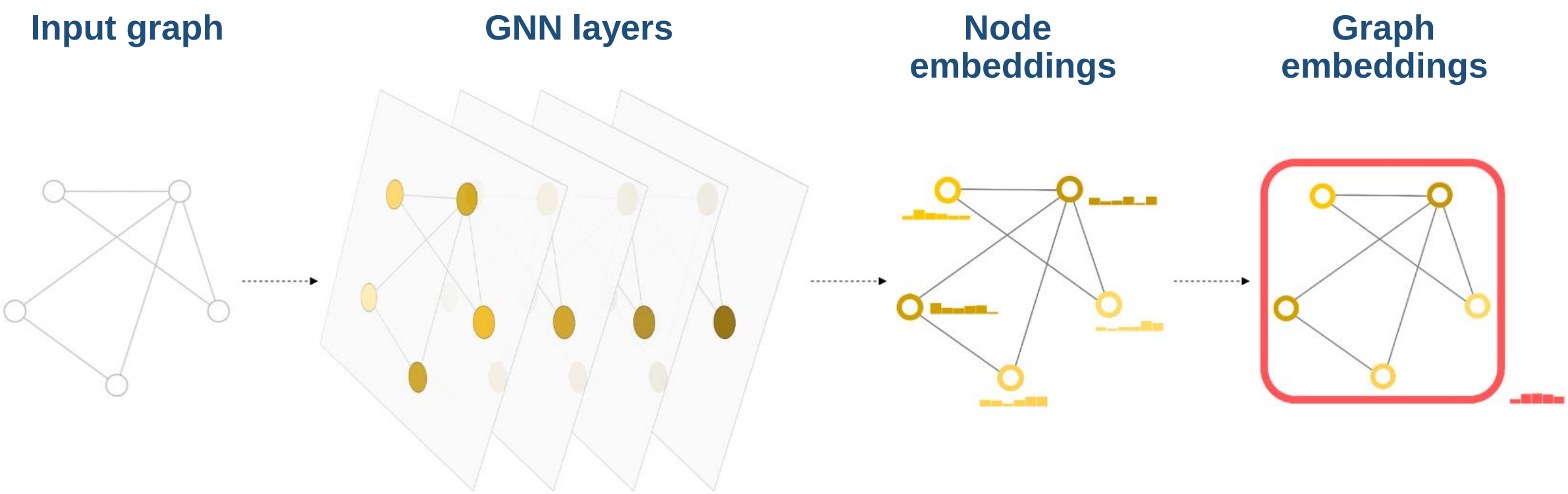
Self-Supervised Learning on Graphs

- Current state-of-the-arts: apply handcrafted augmentations to graphs, and maximise the mutual information between positive pairs, like what people did to images
- You can’t simply add noises to graphs and hope it to work the same as images!
- 3D Infomax: mutual information maximisation between embeddings from 2D and 3D views of molecular graphs
    - Does not require augmentations, but relies on the physical/mathematical properties of molecules
    - Not generalisable
- **We need a principled SSL method that does not require handcrafted augmentations, and can be generalised to various graph types**

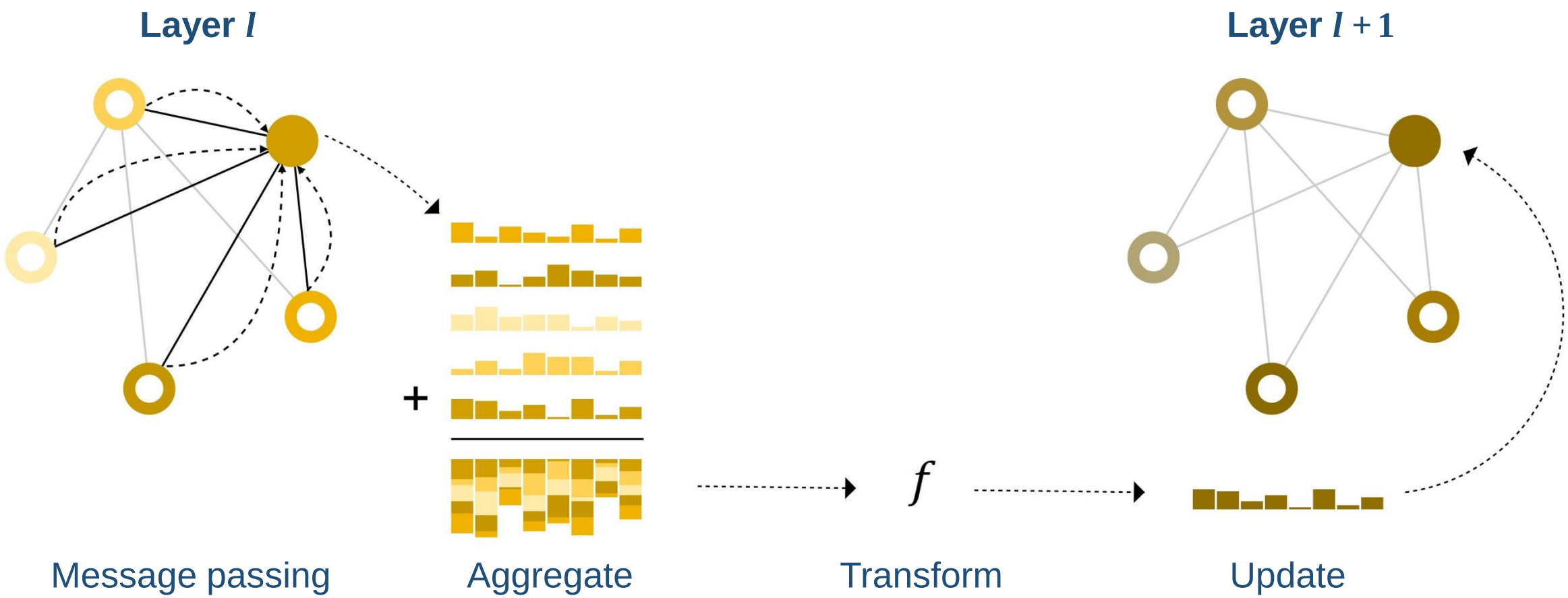
Our Solution: GraphAC (Graph Adversarial Collaboration)

- Key idea:
  - Have two GNNs compete against each other on the same unlabelled graphs
  - Each GNN needs to predict the other GNN's graph embeddings from its own graph embeddings
  - Make the more expressive GNN win by producing more complex and informative graph embeddings
- Conceptually novel, principled, and task-agnostic
- No need for handcrafted augmentations!
- Adversarial: prevent the other GNN from predicting the GNN's own graph embeddings
  Collaboration: produce embeddings for the same graphs, and predict each other's graph embeddings

Graph Neural Networks

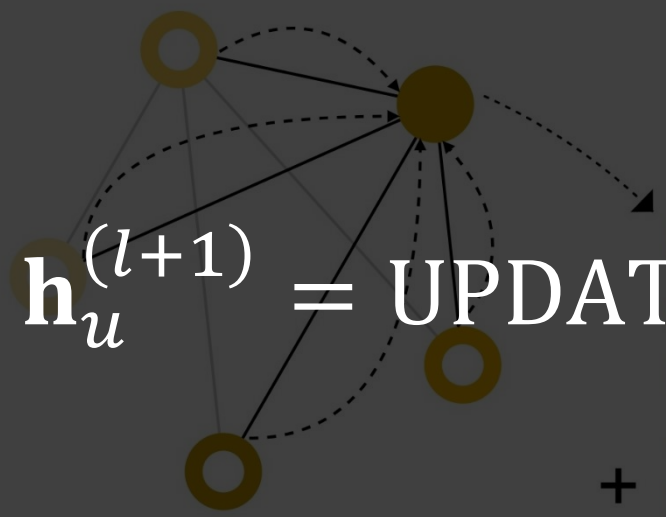


Graph Neural Networks

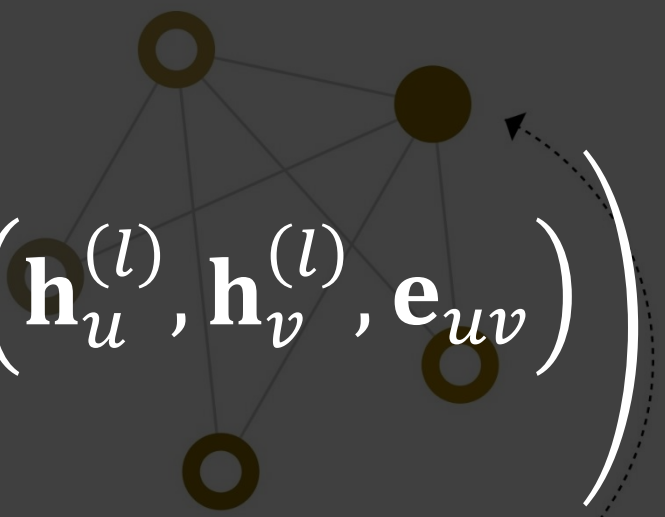


Graph Neural Networks

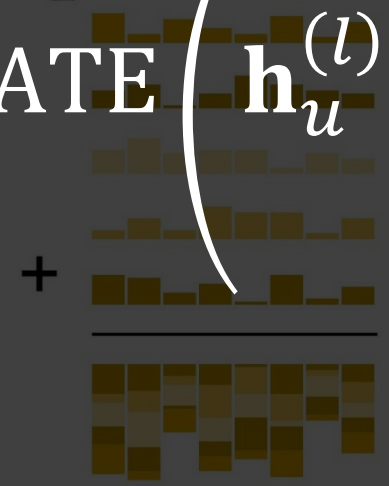
Layer l



Layer $l+1$

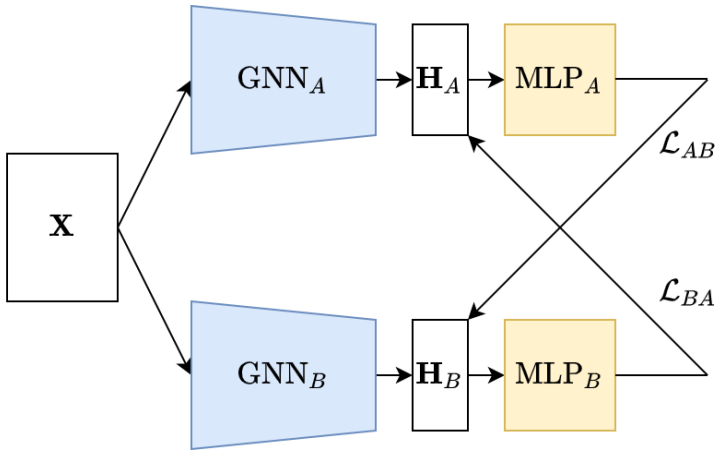


$$\mathbf{h}_u^{(l+1)} = \text{UPDATE}\left(\mathbf{h}_u^{(l)}, \bigoplus_{v \in \mathcal{N}_u} \text{MESSAGE}\left(\mathbf{h}_u^{(l)}, \mathbf{h}_v^{(l)}, \mathbf{e}_{uv}\right)\right)$$



Message passing | Aggregate | Transform | Update

GraphAC Attempt 1: Contestant-Judge Framework (GNNs + MLPs)



$$\mathcal{L}_{\text{MLP}_A} = \mathcal{L}_{AB} = \text{MSE}\left(\mathbf{H}_B, \widehat{\mathbf{H}}_B\right)$$

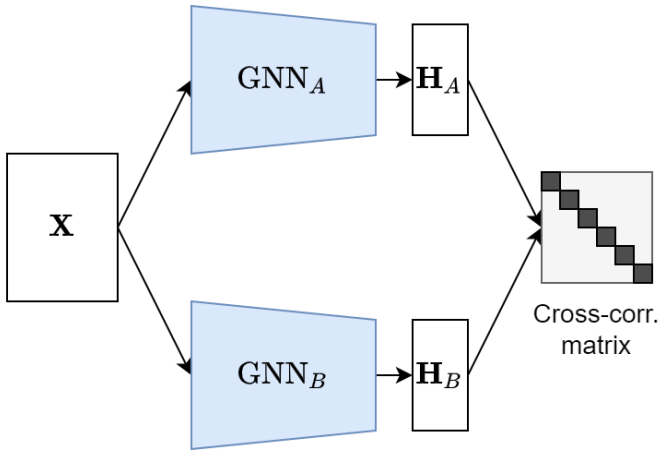
$$\mathcal{L}_{\text{MLP}_B} = \mathcal{L}_{BA} = \text{MSE}\left(\mathbf{H}_A, \widehat{\mathbf{H}}_A\right)$$

$$\mathcal{L}_{\text{GNN}_A} = \mathcal{L}_{AB} - \lambda\mathcal{L}_{BA}$$

$$\mathcal{L}_{\text{GNN}_B} = \mathcal{L}_{BA} - \lambda\mathcal{L}_{AB}$$

Unstable training!

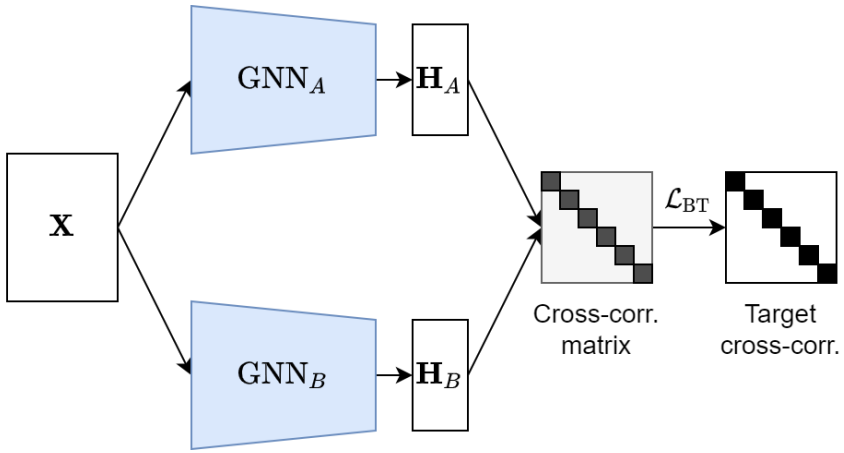
GraphAC Attempt 2: Competitive Barlow Twins



Cross-correlation matrix:

$$C_{ij} = \frac{\sum_b^{N_b} (H_A)_{bi} (H_B)_{bj}}{\sqrt{\sum_b^{N_b} \left((H_A)_{bi}\right)^2} \sqrt{\sum_b^{N_b} \left((H_B)_{bj}\right)^2}}$$

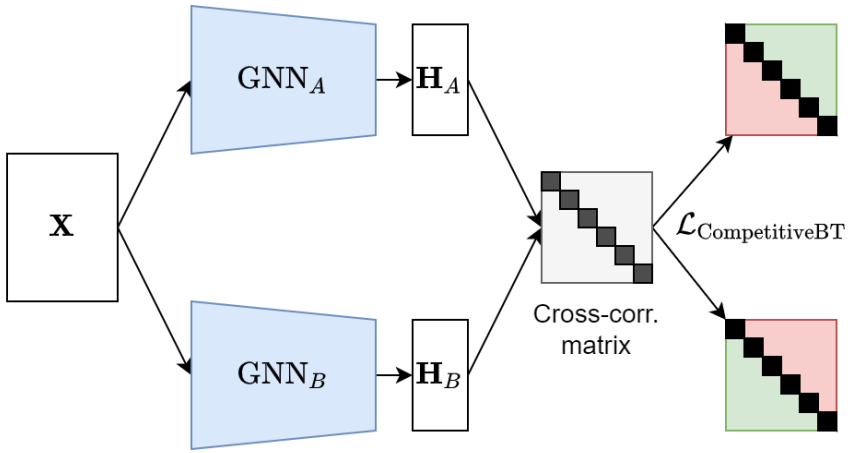
GraphAC Attempt 2: Competitive Barlow Twins



Original Barlow Twins loss (Zbontar et al., 2021):

$$\mathcal{L}_{\mathrm{BT}} = \sum_{i}^{d} (1 - C_{ii})^2 + \lambda \sum_{i}^{d} \sum_{j \neq i}^{d} C_{ij}^2$$

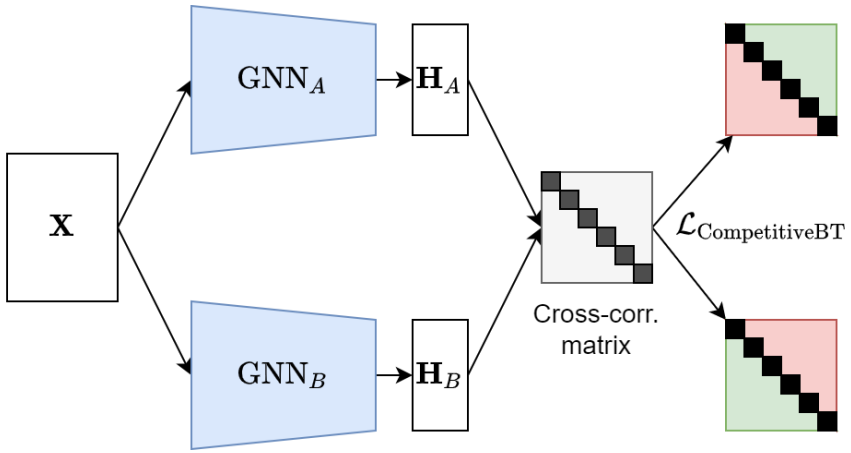
GraphAC Attempt 2: Competitive Barlow Twins



$$\mathcal{L}_{\text{GNN}_A} = \mathcal{L}_{\text{CompetitiveBT}_A} = \sum_{i}^{d}(1 - C_{ii})^2 + \lambda\left(\sum_{i}^{d}\sum_{j>i}^{d} C_{ij}^2 - \mu\sum_{j}^{d}\sum_{i>j}^{d} C_{ij}^2\right)$$

$$\mathcal{L}_{\text{GNN}_B} = \mathcal{L}_{\text{CompetitiveBT}_B} = \sum_{i}^{d}(1 - C_{ii})^2 + \lambda\left(\sum_{j}^{d}\sum_{i>j}^{d} C_{ij}^2 - \mu\sum_{i}^{d}\sum_{j>i}^{d} C_{ij}^2\right)$$

GraphAC Attempt 2: Competitive Barlow Twins



- Stable
- Enables GNNs to predict each other's output graph embeddings
- Enables more expressive GNNs to win

Results

- Evaluated on the `ogbg-molpcba` dataset (437,929 molecular graphs)
- Successfully distinguish different GNNs, and always enable more expressive GNNs to win with statistically significant loss differences
- Switching the order of GNNs does not affect GraphAC's performance
- Allows GNNs with the same expressivity to tie
- Can even produce a total ordering of all GNNs with respect to their expressivity!

Results

- Different numbers of GNN layers (PNAs with 256 hidden dims, aggregators = [max, mean, sum]):

		#Layers in GNN_B				
		2	4	6	8	10
#Layers in GNN_A	2	-0.094	1.402	1.749	1.883	1.977
	4	-1.350	0.078	0.722	0.939	1.345
	6	-1.638	-0.914	0.035	0.701	0.845
	8	-1.837	-1.411	-0.542	0.010	0.516
	10	-1.870	-1.532	-1.177	-0.434	0.063

Results

- Different hidden dimensions (PNAs with 4 layers, aggregators = [max, mean, sum]):

		#Hidden dims in GNN_B				
		16	32	64	128	256
#Hidden dims in GNN_A	16	0.007	1.229	1.964	2.348	2.436
	32	-1.249	-0.016	0.985	1.545	2.102
	64	-2.034	-0.922	-0.019	1.156	1.870
	128	-2.400	-1.671	-1.036	0.092	1.655
	256	-2.560	-2.221	-1.931	-1.443	-0.037

Results

- Different aggregators (PNAs with 4 layers, 64 hidden dims):

		Aggregators in GNN_B			
		[max]	[mean]	[sum]	Combined
Aggregators in GNN_A	[max]	-0.025	0.127	0.322	0.399
	[mean]	-0.154	-0.011	0.228	0.385
	[sum]	-0.329	-0.277	-0.034	0.175
	Combined	-0.342	-0.307	-0.239	-0.019

Results

- Different GNN architectures (PNA, GIN and GCN with 4 layers, 64 hidden dims):

		GNN_B architecture		
		GCN	GIN	PNA
GNN_A architecture	GCN	-0.091	0.483	0.716
	GIN	-0.475	-0.053	0.515
	PNA	-0.652	-0.465	-0.019

Results

- Including the edge features (PNAs with aggregators = [max, mean, sum]):

		#Hidden dimensions		
		64	128	256
#Layers	4	-0.714	-0.750	-0.671
	6	-0.925	-0.558	-0.501
	8	-0.792	-0.309	-0.422

UNIVERSITY OF CAMBRIDGE

Q & A

Department of Computer Science and Technology

UNIVERSITY OF CAMBRIDGE

Special thanks to:

Prof Pietro Liò

Dr Dominique Beaini

Hannes Stärk