UNIVERSITY OF NOTRE DAME

Subgraph Pooling: Tackling Negative Transfer on Graphs

Zehong Wang [1] Zheyuan Zhang [1] Chuxu Zhang [2] Yanfang Ye [1]

[1]University of Notre Dame, Indiana, USA [2]Brandeis University, Massachusetts, USA

IJCAI JEJU 2024

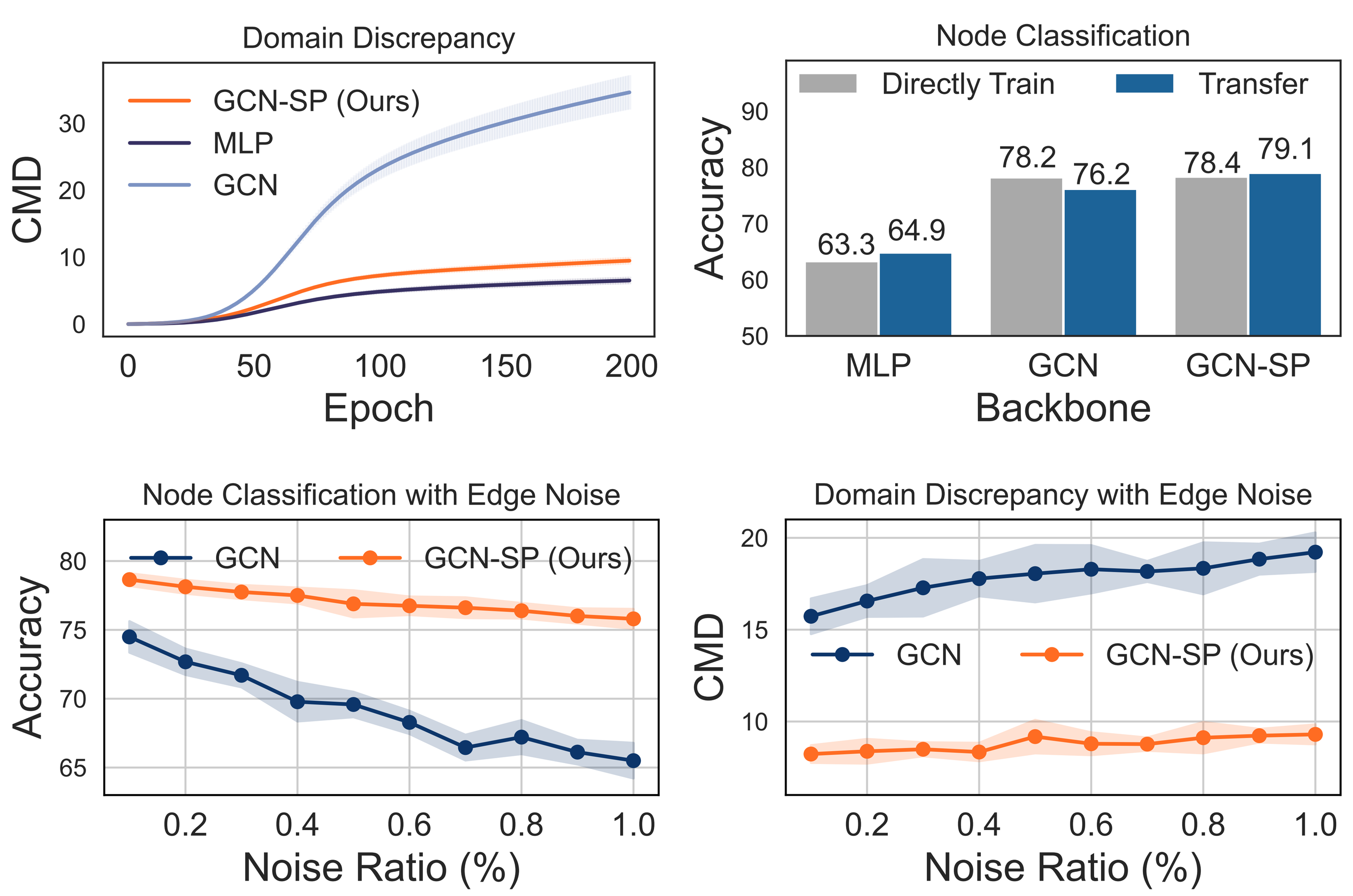
Introduction

Transfer learning aims to improve the performance on the target by leveraging knowledge from different yet related sources. However, when the source and target are not closely related, the target performance may be adversely affected, a phenomenon known as negative transfer. In this work, we reveal that, unlike image and text, **negative transfer commonly occurs in graph-structured data**, even when source and target graphs share semantic similarities. We identify the underlying causes of the issue and provide new insight to solve this.

Contribution

- **Negative Transfer in GNNs.** We find that structural differences between the source and target intensify distribution shifts, as the aggregation process of GNNs is highly sensitive to variations in graph structures. To address this issue, we present a novel insight: for semantically similar graphs, although structural differences lead to significant distribution shift in node embeddings, their impact on subgraph embeddings is marginal.
- **Subgraph Pooling to Tackle Negative Transfer.** Building upon this insight, we introduce plug-and-play modules Subgraph Pooling (SP) and Subgraph Pooling++ (SP++) to mitigate the negative transfer. The key idea is to transfer subgraph information across source and target to prevent the distribution shift.
- **Generality and Effectiveness.** Subgraph Pooling is straightforward to implement and introduces no additional parameters. It involves simple sampling and pooling operations, making it easily applicable to any GNN backbone. We conduct extensive experiments to demonstrate that our method can significantly surpass existing baselines under multiple transfer learning settings.

Negative Transfer on Graphs



Structural differences between the source (DBLP) and target (ACM) amplify the distribution shift on nodes embeddings. **Above:** We illustrate the discrepancy (CMD value) between node embeddings of the source and target during pre-training, and compare the performance of direct training on the target (gray) and transferring knowledge from the source to the target (blue). A large discrepancy results in negative transfer. **Bottom:** We introduce structural noise in the target graph through random edge permutation. Even minor permutations can enlarge the discrepancy (and thus aggravate negative transfer) in vanilla GCN, yet our method effectively mitigates this.

New Insight to Mitigate Negative Transfer

For semantically similar graphs, although structural differences lead to significant distribution shift in node embeddings, their impact on subgraph embeddings is marginal.

Node-level Discrepancy. For nodes $u \in \mathcal{V}^s$ in source graph and $v \in \mathcal{V}^t$ in target graph, we have

$$\mathbb{E}_{u\in\mathcal{V}^s, v\in\mathcal{V}^t} \frac{\mathbf{z}_u^T \mathbf{z}_u}{\mathbf{z}_u^T \mathbf{z}_v} \geq \lambda, \tag{1}$$

where λ denotes the node-level discrepancy.

Subgraph-level Discrepancy. For node $u \in \mathcal{V}^s$ with surrounding subgraph $\mathcal{S}_u^s = (\mathcal{V}_u^s, \mathcal{E}_u^s)$ and node $v \in \mathcal{V}^t$ with surrounding subgraph $\mathcal{S}_v^t = (\mathcal{V}_v^t, \mathcal{E}_v^t)$, we have

$$\mathbb{E}_{u\in\mathcal{V}^s, v\in\mathcal{V}^t} \left\| \frac{1}{n_u^s+1} \sum_{i\in\mathcal{V}_u^s} \mathbf{z}_i - \frac{1}{m_v^t+1} \sum_{j\in\mathcal{V}_v^t} \mathbf{z}_j \right\| \leq \epsilon \tag{2}$$

where $n_u^s = |\mathcal{V}_u^s|$, $m_v^t = |\mathcal{V}_v^t|$, and ϵ denotes the subgraph-level discrepancy.

	ACM → DBLP	DBLP → ACM	Arxiv T1	Arxiv T3
λ	2.413	2.353	2.134	2.683
ϵ (k-hop)	0.212	0.380	0.191	0.203
ϵ (RW)	0.166	0.322	0.184	0.212

Although node-level discrepancy (λ) between source and target is high, the subgraph-level discrepancy (ϵ) remains low. k-hop and RW (Random Walk) indicate two subgraph sampling methods.

Proposed Methods: Subgraph Pooling and Subgraph Pooling++

The key idea is to transfer subgraph-level knowledge across graphs. This is applicable for arbitrary GNNs by adding a subgraph pooling layer at the end of backbone. Specifically, in the SP layer, we first sample the subgraphs around nodes and then perform pooling to generate subgraph embeddings for each node. The choice of sampling and pooling functions can be arbitrary. Here we consider a straightforward k-hop subgraph sampling:

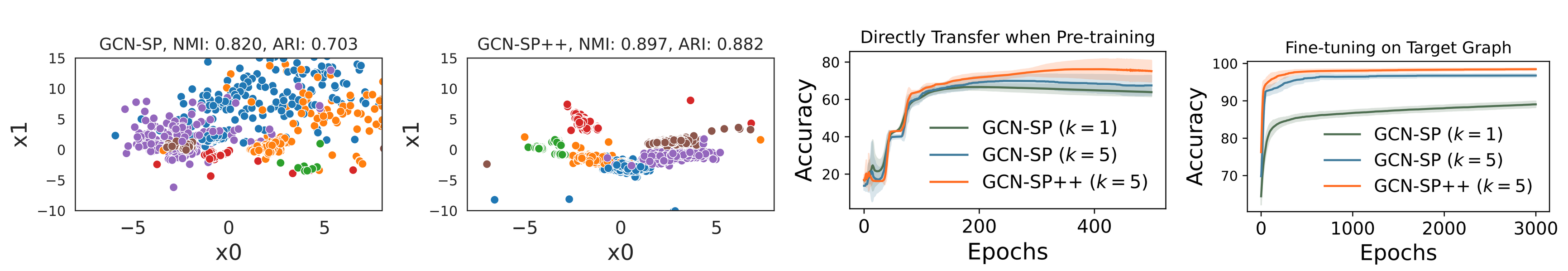
$$\mathcal{N}_s(i) = \text{Sample}_{k\text{-hop}}(\mathcal{G}, i). \tag{3}$$

Subsequently, we pool the subgraph for each node:

$$\mathbf{h}_i = \frac{1}{|\mathcal{N}_s(i)|+1} \sum_{j\in\mathcal{N}_s(i)\cup i} w_{ij}\mathbf{z}_j. \tag{4}$$

where $\mathbf{h}_i \in \mathbf{H}$ represents the subgraph embeddings (the new embeddings for each node), utilized in training the classifier $g(\cdot)$. w_{ij} denotes the pooling weight, which can be either learnable or fixed. Empirically, the MEAN pooling is effective enough to achieve desirable transfer performance.

However, the performance of SP is highly related to the sampled subgraph, which may lead to potential over-smoothing when two nodes share an identical subgraph. To mitigate this, we propose **Subgraph Pooling++ (SP++) that uses random walk to sample neighborhoods around the target nodes**, enforcing the distinction in subgraphs.



A Theoretical Understanding

Theorem 1

For node $u \in \mathcal{V}^s$ in the source graph and $v \in \mathcal{V}^t$ in the target graph, considering the MEAN pooling function, the subgraph embeddings are $\mathbf{h}_u = \frac{\mathbf{z}_u + \sum_{i\in\mathcal{N}_s(u)}\mathbf{z}_i}{n+1}$, $\mathbf{h}_v = \frac{\mathbf{z}_v + \sum_{j\in\mathcal{N}_s(v)}\mathbf{z}_j}{m+1}$ where $n = |\mathcal{N}_s(u)|$, $m = |\mathcal{N}_s(v)|$. We have

$$\|\mathbf{h}_u - \mathbf{h}_v\| \leq \|\mathbf{z}_u - \mathbf{z}_v\| - \Delta, \tag{5}$$

where $\Delta = \frac{(n\|\mathbf{z}_u - \mathbf{z}_v\| - \frac{m-n}{m+1}\|\mathbf{z}_v\|)}{n+1}$ denotes the discrepancy margin.

Corollary 1

If either of the following conditions is satisfied ($|\mathcal{N}_s(u)| \geq |\mathcal{N}_s(v)|$ or $|\mathcal{N}_s(u)|$ is sufficiently large), the inequality $\|\mathbf{h}_u - \mathbf{h}_v\| \leq \|\mathbf{z}_u - \mathbf{z}_v\|$ strictly holds.

Corollary 2

If the following condition is satisfied ($|\mathcal{N}_s(u)| < |\mathcal{N}_s(v)|$), the inequality $\|\mathbf{h}_u - \mathbf{h}_v\| \leq \|\mathbf{z}_u - \mathbf{z}_v\|$ strictly holds when $\lambda \geq 2$, even in extreme case where $|\mathcal{N}_s(u)| \to 0$ and $|\mathcal{N}_s(v)| \to \infty$.

Experimental Results

Model	Citation		Arxiv			
	DBLP → ACM	ACM → DBLP	Time 1	Time 2	Time 3	Time 4
No Transfer	78.23 ± 0.41	97.19 ± 0.18	69.60 ± 0.31			
ERM	76.19 ± 0.92	83.07 ± 0.90	65.73 ± 0.57	66.18 ± 0.48	68.67 ± 0.32	70.33 ± 0.29
Multi-task	76.32 ± 2.79	84.56 ± 1.04	50.32 ± 2.17	52.77 ± 2.82	60.02 ± 0.99	67.62 ± 0.75
EERM	67.96 ± 7.30	90.03 ± 5.30	55.25 ± 2.03	57.47 ± 0.59	63.25 ± 0.54	65.26 ± 0.63
GTrans	77.53 ± 1.94	95.19 ± 0.69	65.95 ± 0.12	66.64 ± 0.51	<u>69.51 ± 0.39</u>	<u>71.54 ± 0.30</u>
GNN-SP	<u>79.18 ± 0.40</u>	<u>97.54 ± 1.01</u>	<u>67.76 ± 0.23</u>	<u>68.36 ± 0.33</u>	69.03 ± 0.63	69.75 ± 0.56
GNN-SP++	**79.20 ± 0.23**	**98.20 ± 0.54**	**71.43 ± 0.52**	**72.75 ± 1.24**	**74.04 ± 0.83**	**75.17 ± 0.21**

Model	Airport					
	Brazil → Europe	USA → Europe	Brazil → USA	Europe → USA	USA → Brazil	Europe → Brazil
No Transfer	48.63 ± 3.70		59.18 ± 1.76		52.36 ± 6.46	
ERM	45.00 ± 2.95	39.29 ± 3.96	47.83 ± 1.92	47.79 ± 4.66	39.53 ± 7.70	44.62 ± 4.24
Multi-task	48.55 ± 1.48	47.61 ± 2.02	48.73 ± 2.01	<u>50.96 ± 2.12</u>	52.17 ± 2.13	52.92 ± 6.07
EERM	<u>48.77 ± 2.85</u>	46.88 ± 4.70	<u>48.91 ± 4.19</u>	48.36 ± 3.74	45.67 ± 3.68	46.65 ± 5.93
GTrans	48.50 ± 1.31	47.49 ± 2.41	48.84 ± 0.99	48.88 ± 1.25	52.30 ± 1.50	53.00 ± 4.12
GNN-SP	48.76 ± 2.61	**51.30 ± 2.22**	46.06 ± 5.44	49.85 ± 5.55	<u>55.47 ± 5.90</u>	<u>54.72 ± 5.48</u>
GNN-SP++	**50.90 ± 3.93**	<u>50.40 ± 2.27</u>	**51.06 ± 6.17**	**53.87 ± 5.99**	**57.08 ± 6.13**	**55.23 ± 9.36**