ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Designing Robust Graph Neural Network against Distribution Shift

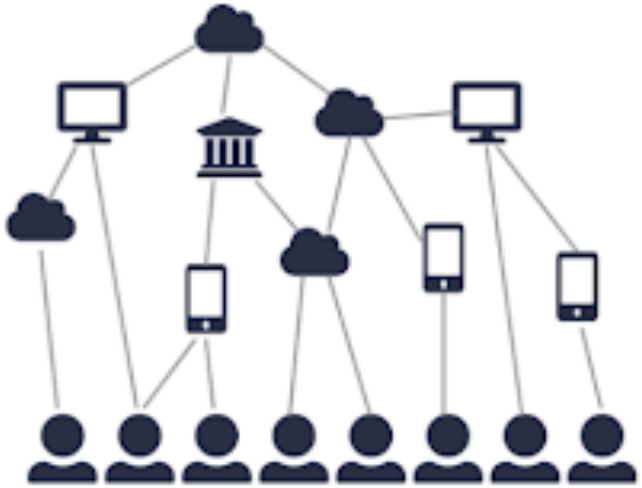
Qi Zhu

Department of Computer Science

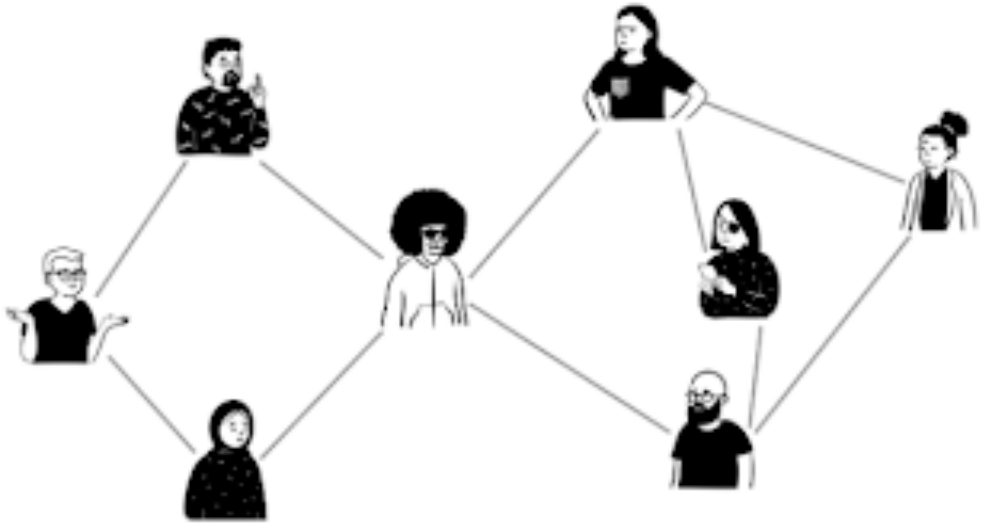
University of Illinois Urbana-Champaign

qiz3@Illinois.edu

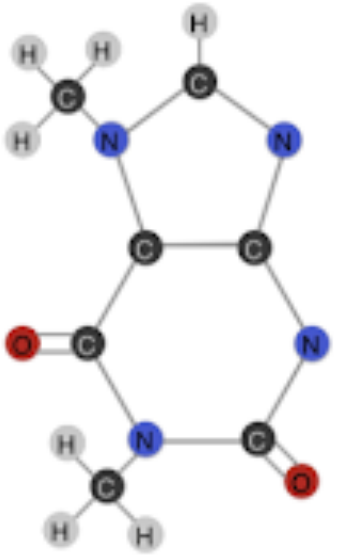
Graphs are ubiquitous



web graph



social networks



molecules

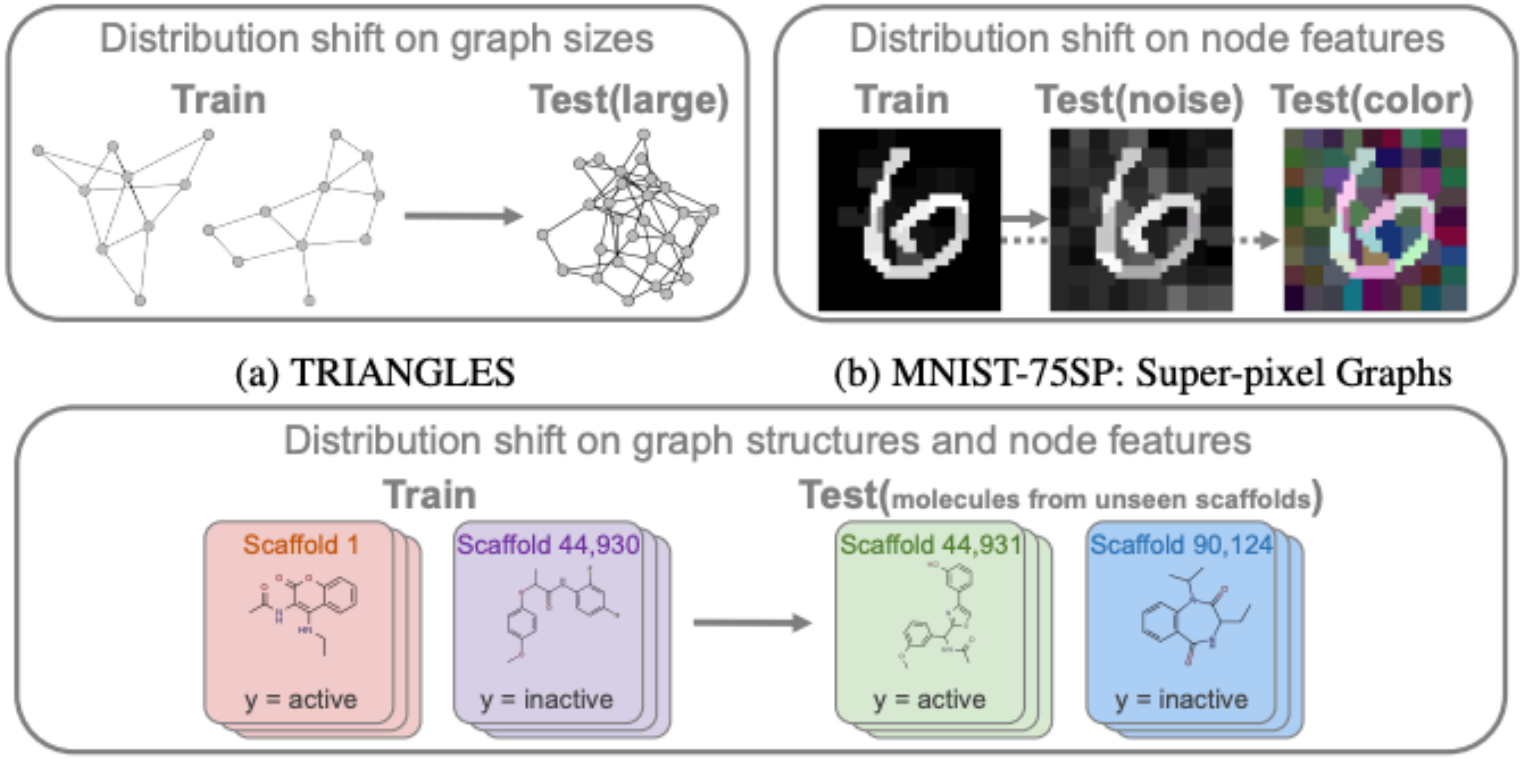
picture credit (https://ai.googleblog.com/2022/03/robust-graph-neural-networks.html)

Machine leaning on graphs

Main ML tasks in graphs

- Node classification
  - Rumor and fake news detection
- Link Prediction
  - Friend recommendation
- Graph property prediction
  - Molecular property
  - Molecular dynamics

Challenges of distribution shifts



(c) OGB Molecule Dataset [27]. For validating OOD generalization, this dataset is split based on the scaffolds (i.e., two-dimensional structural frameworks) of molecules. The testing set consists of structurally distinct molecules with scaffolds that are not in the training set.

picture credit (https://arxiv.org/pdf/2112.03806.pdf)

This Talk

- Pre-training / self-supervised learning
  - How to improve GNN generalization during testing time ?
- Design robust GNNs for semi-supervised node classification
  - Handling localized training data
  - Handling more node-level distribution shifts
- Distribution shifts on graph-level tasks
- Topics not covered
  - Robustness towards adversarial attacks
  - Representation power of GNNs (e.g. expressiveness, invariance, equivariance)

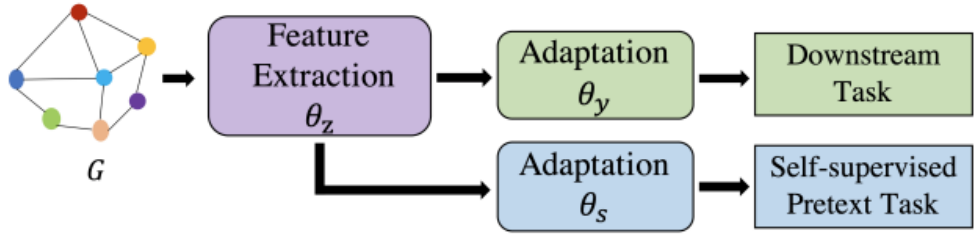
Outline

- **Pre-training / self-supervised learning**
  - How to improve GNN generalization during testing time ?
- Design robust GNNs for semi-supervised node classification
  - Handling localized training data
  - Handling more node-level distribution shifts
- Distribution shifts on graph-level tasks

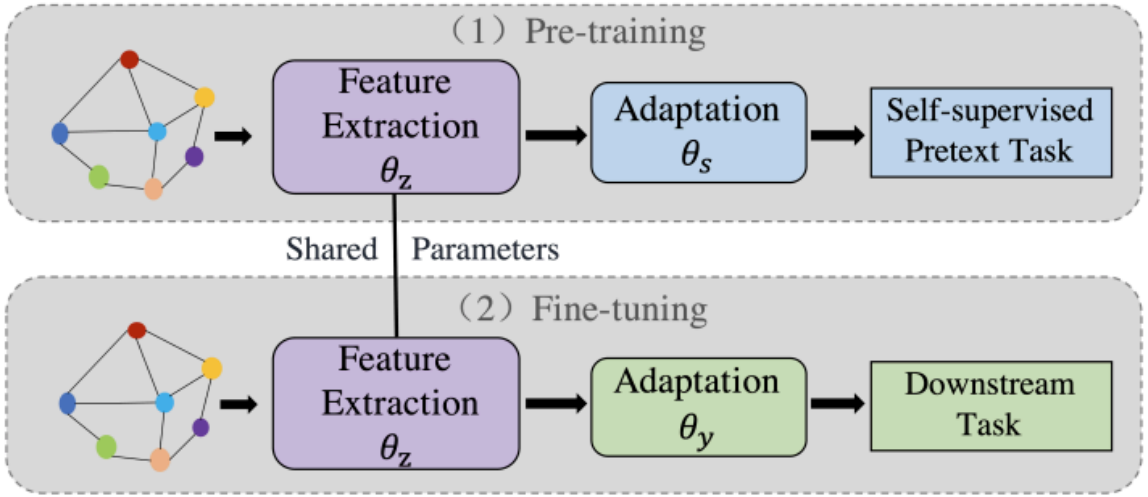
Main paradigm for pre-training in GNN

- Joint SSL training

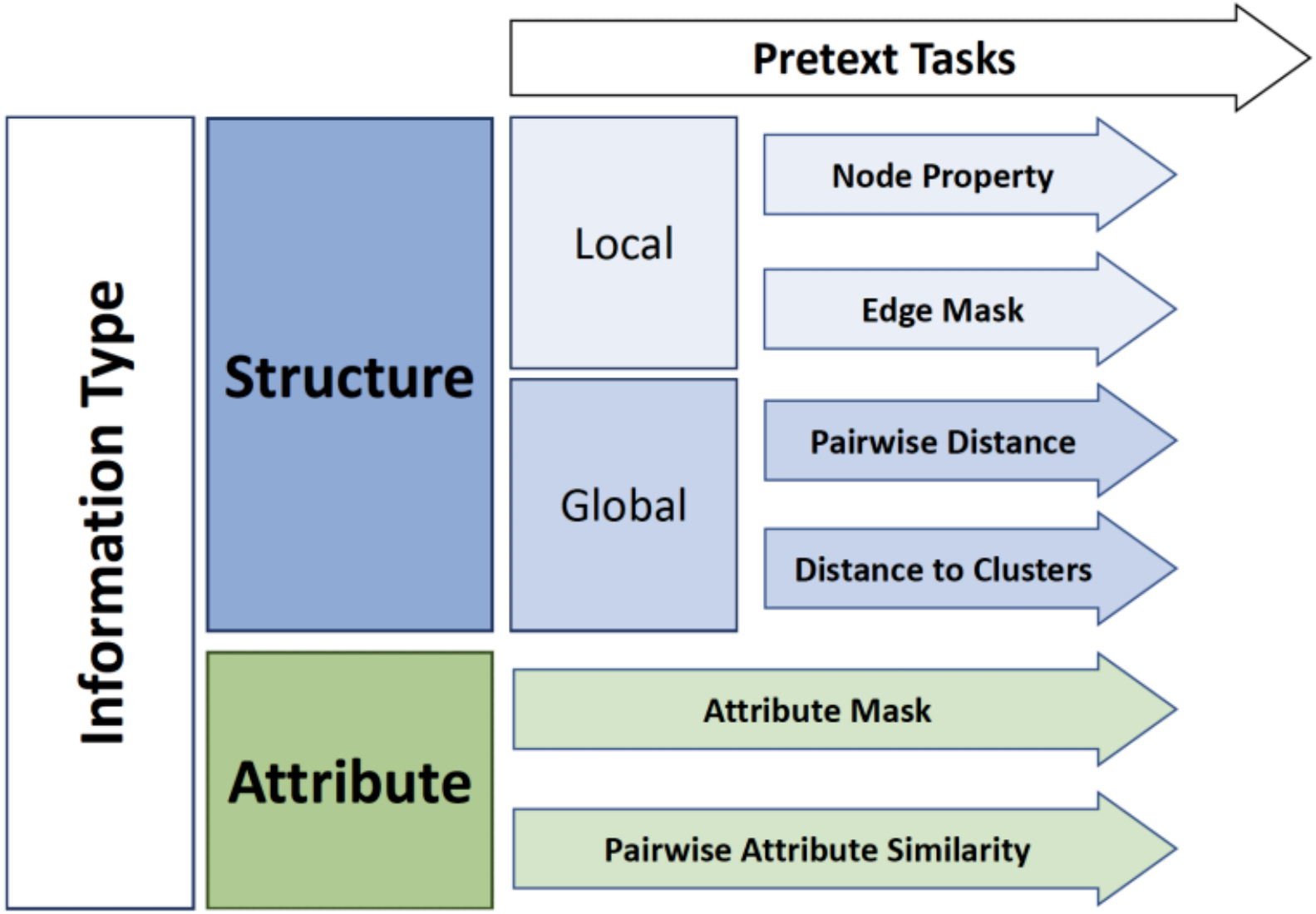
$$\min_{\theta,\theta'} \mathcal{L}_{task}\left(\theta, \mathbf{A}, \mathbf{X}, \mathcal{D}_L\right) + \lambda \mathcal{L}_{self}\left(\theta', \mathbf{A}, \mathbf{X}, \mathcal{D}_U\right)$$



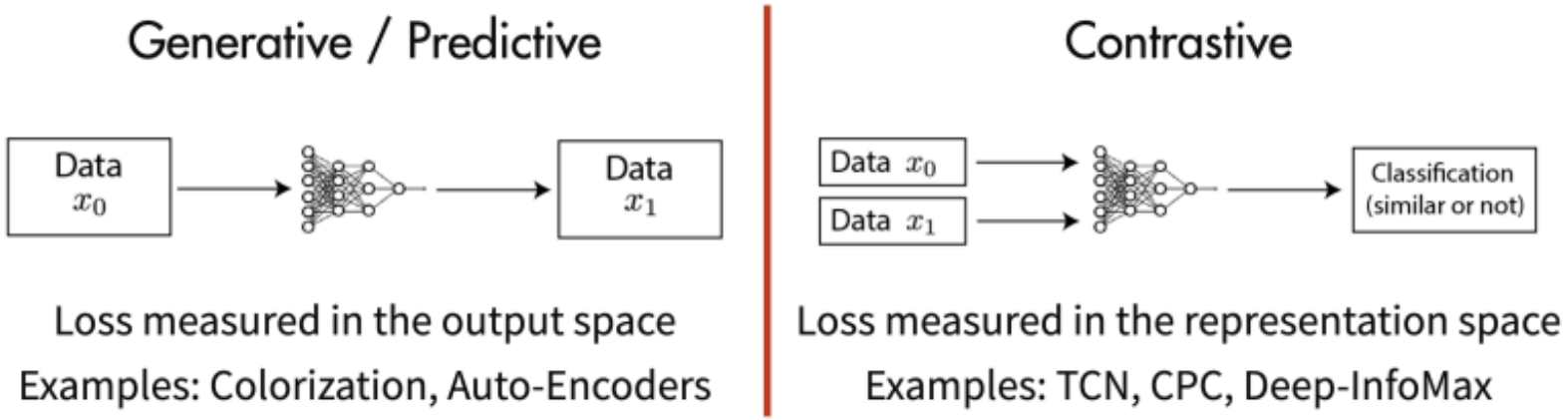
- Two-stage training



Examples of pretext tasks



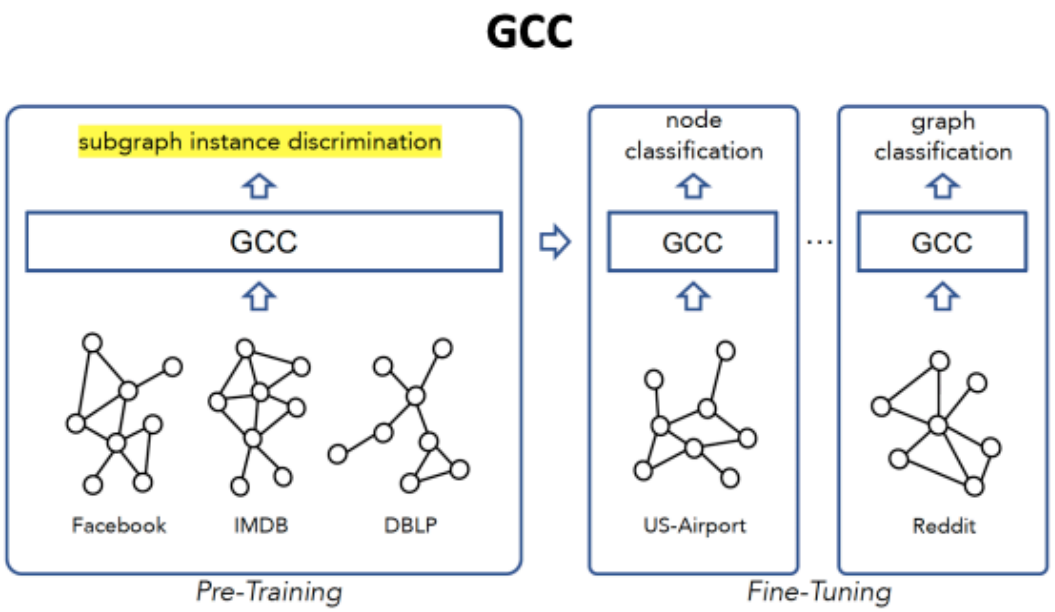
Predictive and contrastive pre-training



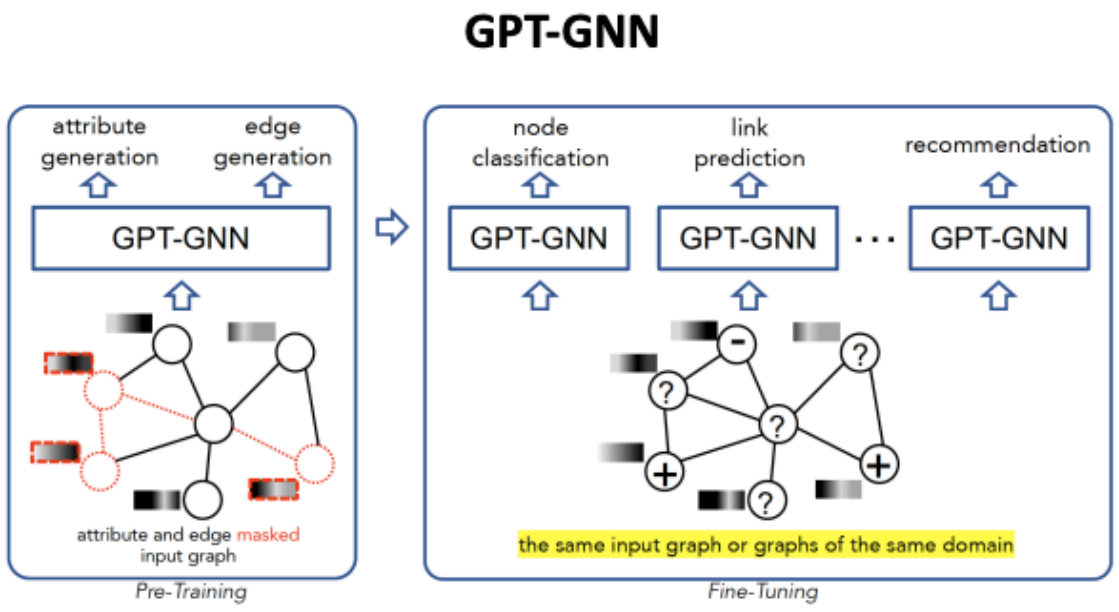
- **Given:** $X = \{x, \ x^+, \ x_1^-, \ldots, x_{N-1}^-\}$; Similarity function $s(\cdot)$ (e.g., cosine similarity)
- **Goal:** $s\big(f(x), f(x^+)\big) > s(f(x), f(x^-))$
- **Contrastive/InfoNCE Loss**

$$\mathcal{L}_N = -\mathbb{E}_X \left[\log \frac{\exp\left(s\big(f(x), f(x^+)\big)\right)}{\exp\left(s\big(f(x), f(x^+)\big)\right) + \sum_{j=1}^{N-1} \exp\left(s\left(f(x), f(x_j^-)\right)\right)} \right]$$

Different pre-training settings

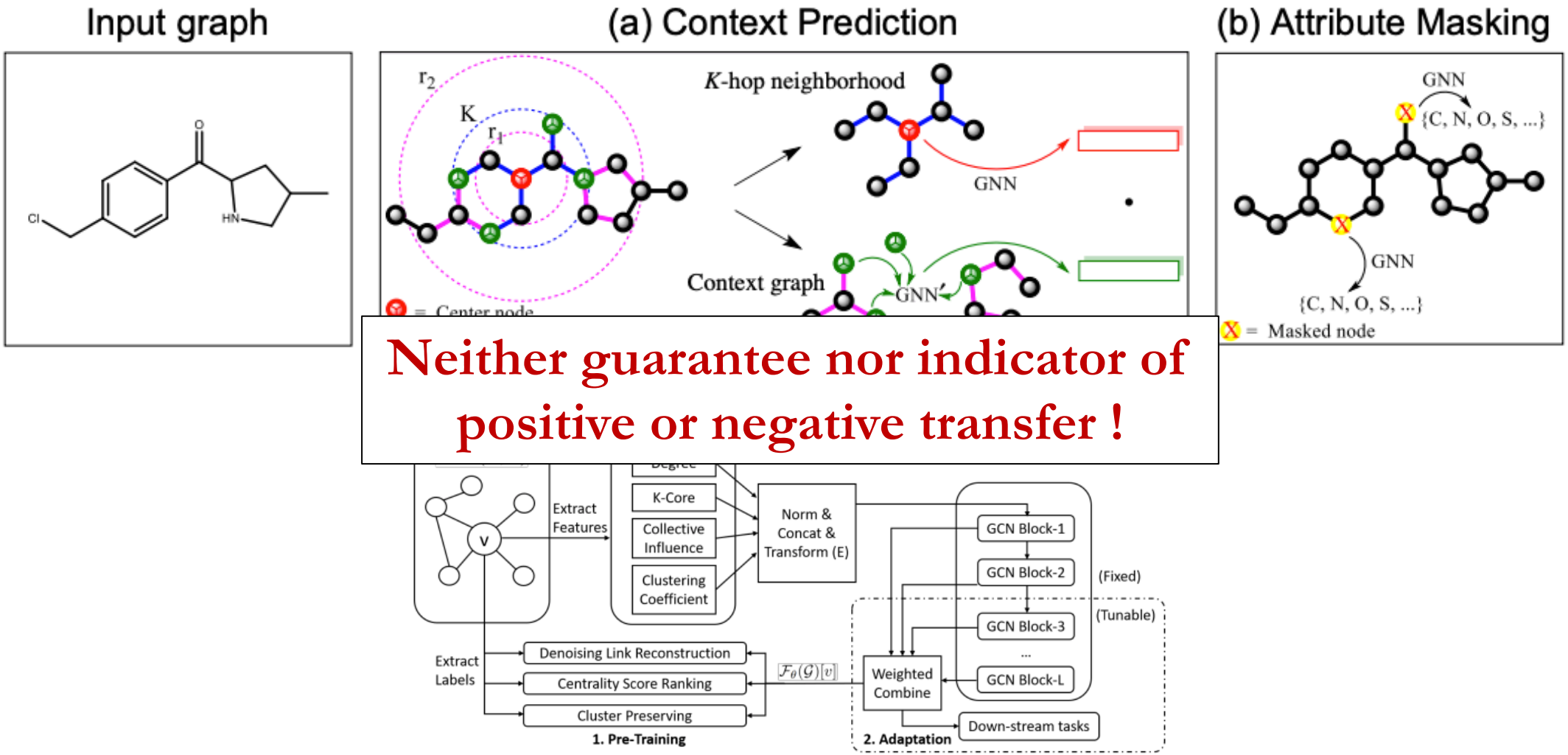


- To pre-train from **some graphs**
- To fine-tune for **unseen tasks on unseen graphs**



- To pre-train from **one graph**
- To fine-tune for **unseen tasks on the same graph or graphs of the same domain**

GNN Pre-training vs. distribution shifts

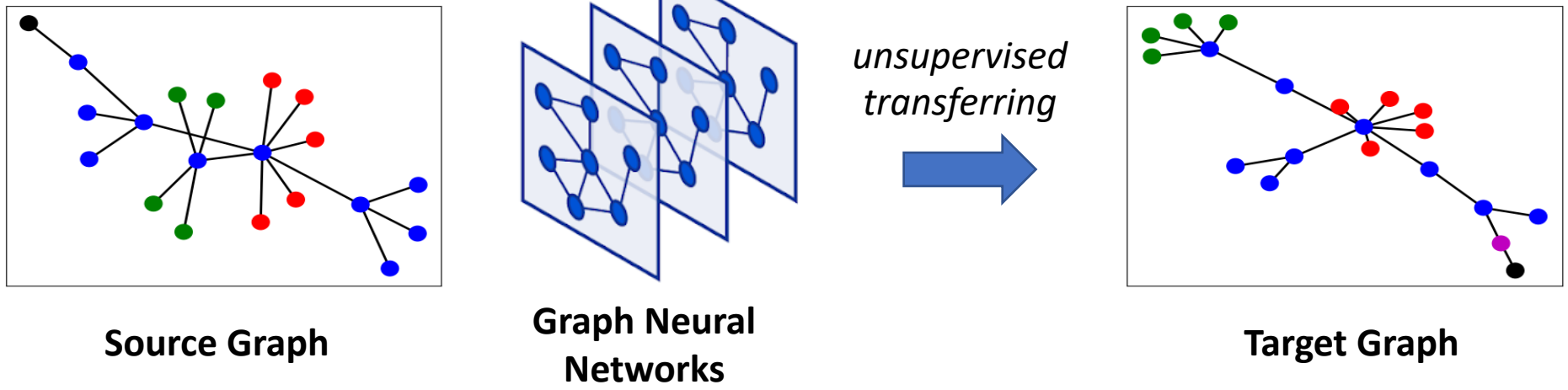


Pre-Training Graph Neural Networks for Generic Structural Feature Extraction

Outline

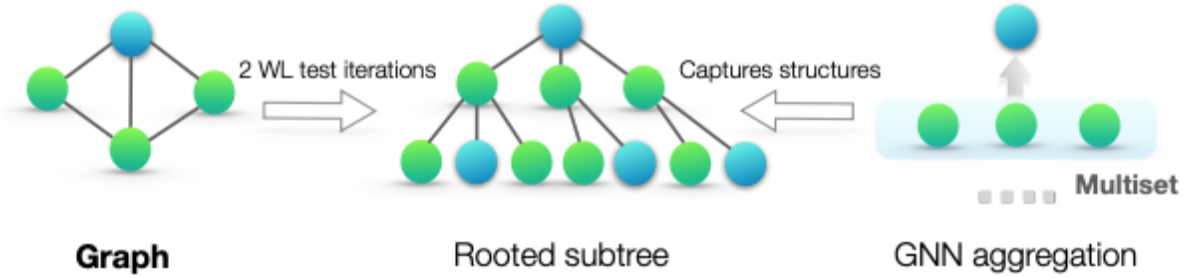
- Pre-training / self-supervised learning
  - **How to improve GNN generalization during testing time ?**
    - Transfer learning of graph neural networks with ego-graph information maximization (Neurips 21')
- Design robust GNNs for semi-supervised node classification
  - Handling localized training data
  - Handling more node-level distribution shifts
- Distribution shifts on graph-level tasks

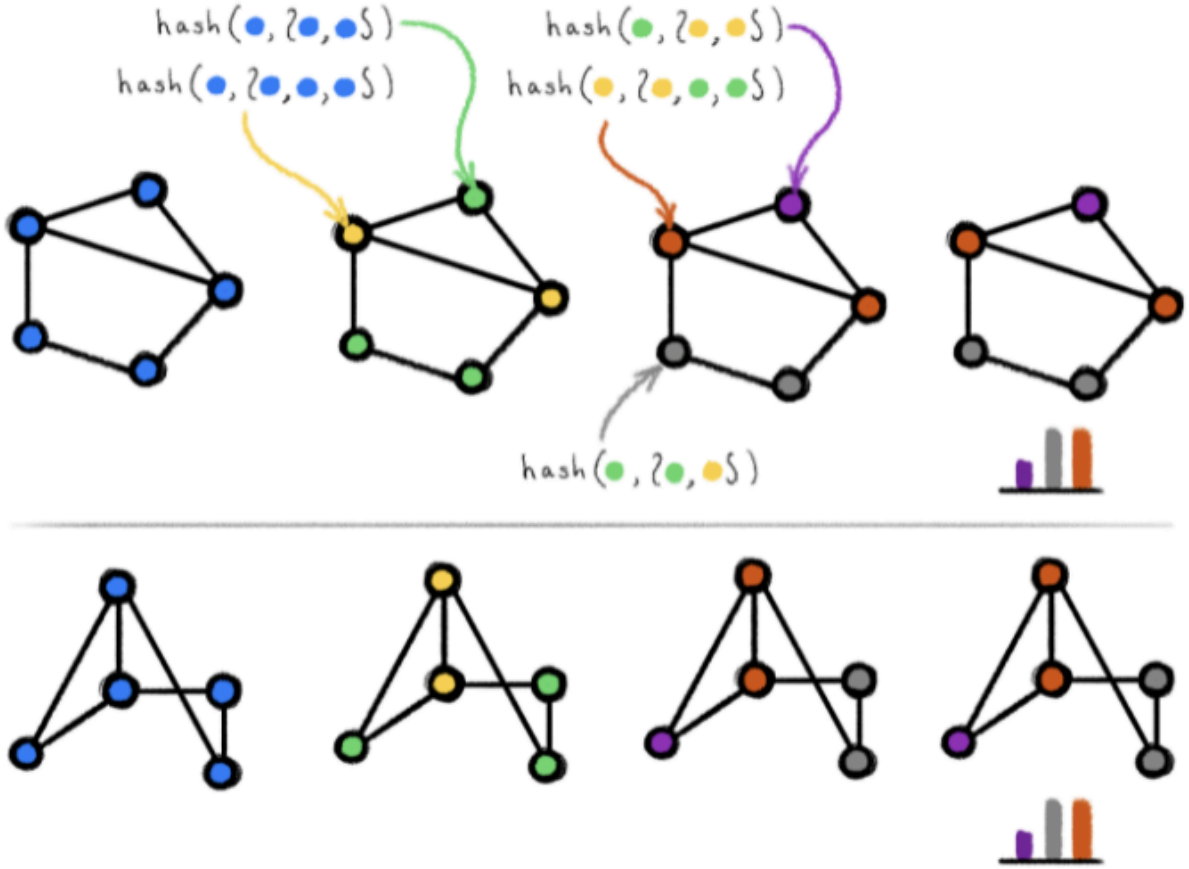
A transfer learning perspective on GNNs



Graph Similarity as an indicator

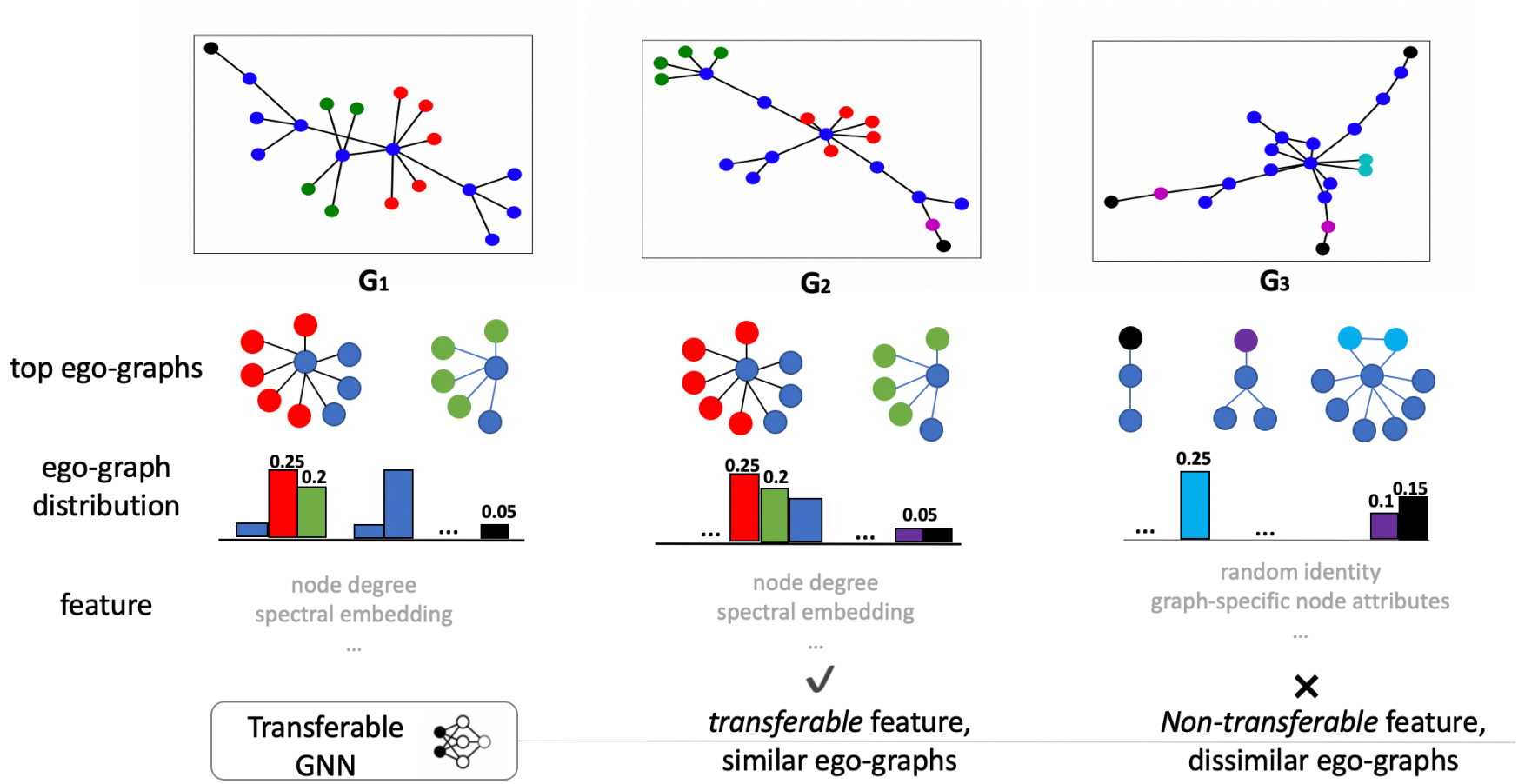
- WL-test use rooted subtree to distinguish different graphs.





Can we use rooted subtree (ego-graph) to measure the similarity between graphs ?

Ego-graph distribution difference as indicator



A natural view of graph neural network is a function F over graph(ego-graph) and node features.
Hence, transferability is measured upon domain (feature) discrepancy.

Definition of structural information

Definition 3.1 (K-hop ego-graph). *We call a graph* $g_i = \{V(g_i), E(g_i)\}$ *a* k*-hop ego-graph centered at node* v_i *if it has a* k*-layer centroid expansion [4] such that the greatest distance between* v_i *and any other nodes in the ego-graph is* k*,* i.e. $\forall v_j \in V(g_i), |d(v_i, v_j)| \leq k$*, where* $d(v_i, v_j)$ *is the graph distance between* v_i *and* v_j*.*

Definition 3.2 (Structural information). *Let* $\mathcal{G}$ *be a topological space of sub-graphs, we view a graph* G *as samples of* k*-hop ego-graphs* $\{g_i\}_{i=1}^n$ *drawn* i.i.d. *from* $\mathcal{G}$ *with probability* μ*,* i.e.*,* $g_i \overset{\text{i.i.d.}}{\sim} \mu \ \forall i = 1, \cdots, n$*. The structural information of* G *is then defined to be the set of* k*-hop ego-graph of* $\{g_i\}_{i=1}^n$ *and their empirical distribution.*

Design of transferable learning objective

- Motivation: if self-supervised model approximates the ego-graph distribution of the source graph. The inference error on target graph ε_t therefore, captures the structural difference if ε_s is small.
- We further use empirical loss different Δl between source and target graph to evaluate the potential of such transfer.

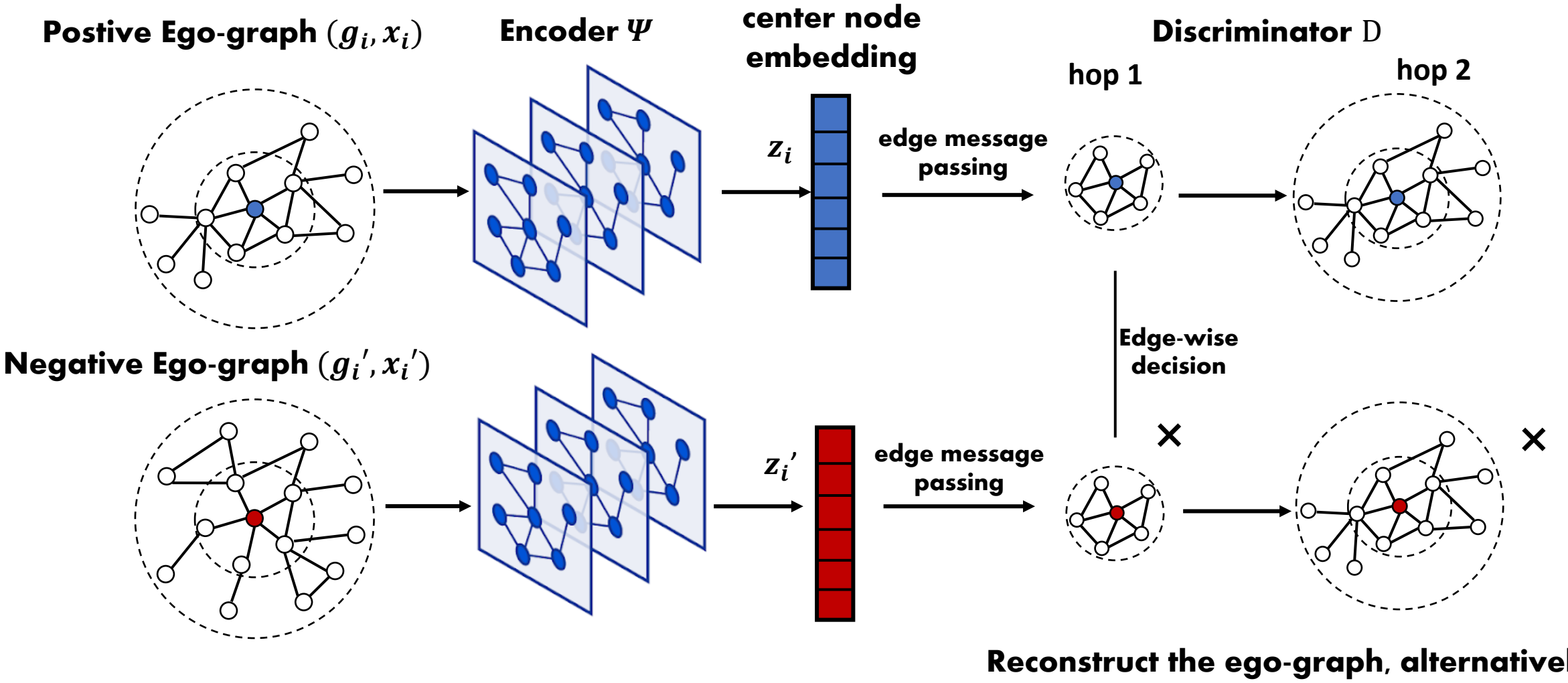
Ego-graph Information Maximization (EGI)

- To capture the joint distribution of structural information and node features, an idea GNN maximize the mutual information between structural information $\{g_i, x_i\}$ and its output Ψ. Such that,

$$\mathcal{I}^{(\mathrm{JSD})}(\mathcal{G}, \Psi) = \mathbb{E}_{\mathbb{P}}\left[-\mathrm{sp}\left(-T_{\mathcal{D},\Psi}(g_i, \Psi(g_i, x_i))\right)\right] - \mathbb{E}_{\mathbb{P}\times\tilde{\mathbb{U}}}\left[\mathrm{sp}\left(T_{\mathcal{D},\Psi}(g_i, \Psi(g_i', x_i'))\right)\right]$$

- Discriminator D is asked to distinguish the samples from joint distribution and product of two marginal distributions.

EGI Model Optimization



Transferability of EGI

Theorem A.2. *Let $G_a = \{(g_i, x_i)\}_{i=1}^n$ and $G_b = \{(g_{i'}, x_{i'})\}_{i'=1}^m$ be two graphs and node features are structure-respecting with $x_i = f(L_{g_i}), x_{i'} = f(L_{g_{i'}})$ for some function $f : \mathbb{R}^{|V(g_i)| \times |V(g_i)|} \to \mathbb{R}^d$. Consider GCN Ψ_θ with k layers and a 1-hop polynomial filter ϕ,the empirical performance difference of Ψ_θ with $\mathcal{L}_{\text{EGI}}$ satisfies*

$$|\mathcal{L}_{\text{EGI}}(G_a) - \mathcal{L}_{\text{EGI}}(G_b)| \leq \mathcal{O}\left(\frac{1}{nm}\sum_{i=1}^{n}\sum_{i'=1}^{m}[M + C\lambda_{\max}(L_{g_i} - L_{g_{i'}}) + \tilde{C}\lambda_{\max}(\tilde{L}_{g_i} - \tilde{L}_{g_{i'}}))]\right), \tag{1}$$

where M is dependant on Ψ, $\mathcal{D}$, node features, and the largest eigenvalue of L_{g_i} and $\tilde{L}_{g_i}$. C is a constant dependant on the encoder, while $\tilde{C}$ is a constant dependant on the decoder. With a slight abuse of notation, we denote $\lambda_{\max}(A) := \lambda_{\max}(A^T A)^{1/2}$. Note that, in the main paper, we have $C := M + C\lambda_{\max}(L_{g_i} - L_{g_{i'}})$, and $\Delta_{\mathcal{D}}(G_a, G_b) := \tilde{C}\lambda_{\max}(\tilde{L}_{g_i} - \tilde{L}_{g_{i'}})$.

- The above theorem states the empirical risk difference on source and target graph are bounded by the Laplacian difference on in-degree and out-degree adjacency matrices.
- Specifically, the EGI bound term $\Delta_D(G_a, G_b)$ describes the transferability of the EGI objective.

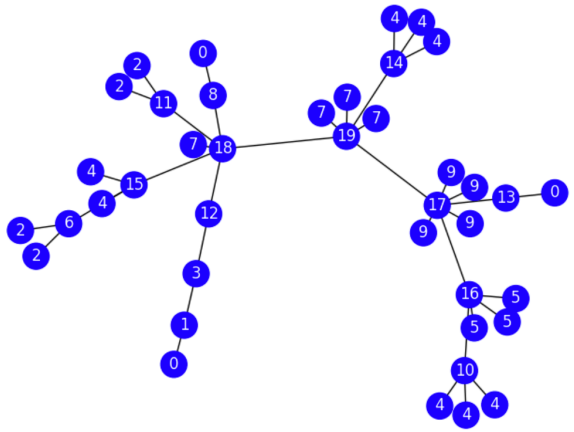
Application of EGI

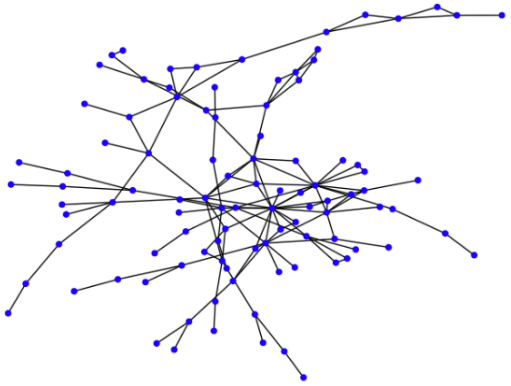
- Usage of EGI
  - Have a series of similar large graph on different task, train EGI embedding on any of the graph and get transferable embedding easily.
- Usage of EGI gap term $\Delta_D(G_a, G_b)$
  - *point-wise pre-judge: compute the term between source and target graph to assess the potential of positive transfer (< 1.0 in practice)*
  - *pair-wise pre-selection: when multiple source graphs are available G_a^1, G_a^2, G_a^n select most suitable source graph G_a^* with the smallest EGI gap Δ_D*

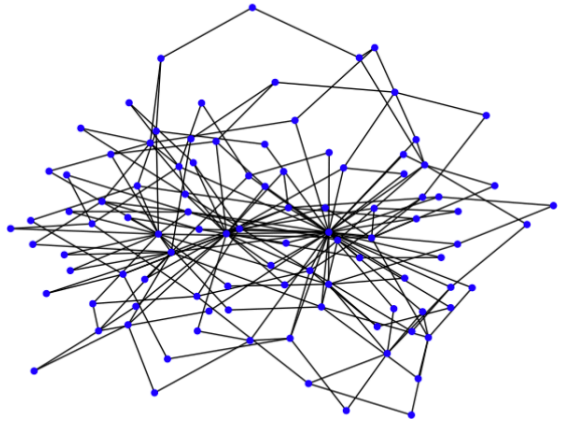
Experiments

- Synthetic Experiment
  - Limit the power of rooted subtree by number of hop and still try to find structural equivalent nodes
- Unsupervised Transfer on node classification
  - Train self-supervised encoder on source graph. Obtain node embeddings on target graph without fine-tuning.
- Few-shot fine-tuning on relation classification
  - Jointly train the encoder and task-specific loss

Synthetic experiments







Synthetic task: finding structural equivalent nodes

(a) Forest-fire graph example

(b) Barabasi-albert graph example

Method	transferable features			non-transferable feature			structural difference	
	F-F	B-F	δ(acc.)	F-F	B-F	δ(acc.)	$\Delta_{\mathcal{D}}$(F,F)	$\Delta_{\mathcal{D}}$(B,F)
GIN (untrained)	0.572	0.572	/	0.358	0.358	/		
VGAE (GIN)	0.498	0.432	+0.066	0.240	0.239	0.001		
DGI (GIN)	0.578	0.591	-0.013	0.394	0.213	+0.181	0.752	0.883
EGI (GIN)	**0.710**	0.616	+0.094	0.376	0.346	+0.03		

Real Data Experiments

Task: Unsupervised transferring on role identification

Dataset: Airport (USA, Europe, Brazil), role – level of popularity

Table 2: Results of role identification with direct-transfering on the Airport dataset. The performance reported (%) are the average over 100 runs. The scores marked with ** passed t-test with $p < 0.01$ over the second best results.

Method	Europe (source)		USA (target)		Brazil (target)	
	node degree	uniform	node degree	uniform	node degree	uniform
features	52.81	20.59	55.67	20.22	67.11	19.63
GIN (untrained)	55.75	53.88	61.56	58.32	70.04	70.37
GVAE (Kipf & Welling, 2016)	53.90	21.12	55.51	22.39	66.33	17.70
DGI (Velickovic et al., 2019)	57.75	22.13	54.90	21.76	67.93	18.78
MaskGNN (Hu et al., 2019a)	56.37	55.53	60.82	54.64	66.71	74.54
ContextPredGNN (Hu et al., 2019a)	52.69	49.95	50.38	54.75	62.11	70.66
Structural Pre-train (Hu et al., 2019b)	56.00	53.83	62.17	57.49	68.78	72.41
EGI	**59.15****	54.98	**64.55****	57.40	**73.15****	70.00

Common self-supervised algorithms such as DGI and GVAE fails to positive transfer.

Real Data Experiments

Task: Unsupervised transferring + fine-tuning on Link Prediction

Dataset: knowledge graph (YAGO)

Post-fine-tuning: use transferred encoder Ψ

Joint-fine-tuning: jointly optimize the EGI and task objective on target

Method	post-fine-tuning		joint-fine-tuning	
	AUROC	MRR	AUROC	MRR
No pre-train	0.6866	0.5962	N.A.	N.A
GVAE [24]	0.7009	0.6009	0.6786	0.5676
DGI [45]	0.6885	0.5861	0.6880	0.5366
Mask-GIN [19]	0.7041	0.6242	0.6720	0.5603
ContextPred-GIN [19]	0.6882	0.6589	0.5293	0.3367
EGI	**0.7389****	**0.6695**	**0.7870****	**0.7289****

Model Analysis

- Efficient Computation of term Δ_D
  - Enumerating every single pair of ego-graph between source and target graph can easily blow up the memory (N by M pairs – N,M is the number of nodes).
  - In practice, we can estimate it by uniformly down sample such pairs

Sampling frequency	Europe-USA	Europe-Brazil
100 pairs	0.872±0.039	0.854±0.042
1000 pairs	0.859±0.012	0.848±0.007
Full	0.869	0.851

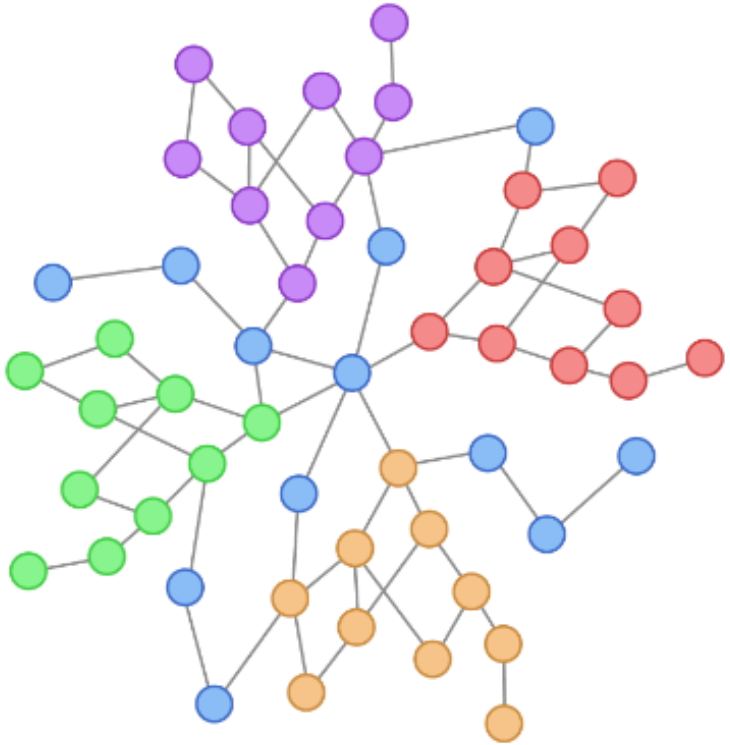
- Relation to the depth of rooted subtree (ego-graph)

	Europe (source)	USA (target)	Brazil (target)
Method	acc	acc, Δ_D	acc, Δ_D
EGI (k=1)	58.25	60.08, 0.385	60.74, 0.335
EGI (k=2)	59.15	64.55, 0.869	73.15, 0.851
EGI (k=3)	57.63	64.12, 0.912	72.22, 0.909

Outline

- Pre-training / self-supervised learning
  - How to understand GNN generalization during testing time ?
- Design robust GNNs for semi-supervised node classification
  - **Handling localized training data**
    - Shift-robust gnns: overcoming the limitations of localized graph training data (Neurips 21')
  - Handling more node-level distribution shifts
- Distribution shifts on graph-level tasks

What is localized training data?



picture credit (https://ai.googleblog.com/2022/03/robust-graph-neural-networks.html)

Localized data is covariate shift

- A general graph neural network layer, final representation $Z = H^k$

$$H^k = \sigma(\tilde{A}H^{k-1}\theta^k)$$

- To learn a semi-supervised classifier, cross-entropy loss function l is widely used

$$\mathcal{L} = \frac{1}{M}\sum_{i=1}^{M} l(y_i, z_i),$$

- Data-shift [1] happens when the training data is biased from testing
  - $\mathbf{Pr}_{\text{train}}(X, Y) \neq \mathbf{Pr}_{\text{test}}(X, Y)$
  - In a neural network, we care about the shift happens in the last hidden activated layer Z, i.e. $\mathbf{Pr}_{\text{train}}(Z, Y) \neq \mathbf{Pr}_{\text{test}}(Z, Y)$
  - Covariate shift assumes, $\mathbf{Pr}_{\text{train}}(Y|Z) = \mathbf{Pr}_{\text{test}}(Y|Z)$, such that,

$$\Pr_{\text{train}}(Z, Y) \neq \Pr_{\text{test}}(Z, Y) \rightarrow \Pr_{\text{train}}(Z) \neq \Pr_{\text{test}}(Z)$$

[1] Quiñonero-Candela, Joaquin, et al., eds. *Dataset shift in machine learning*. Mit Press, 2009.

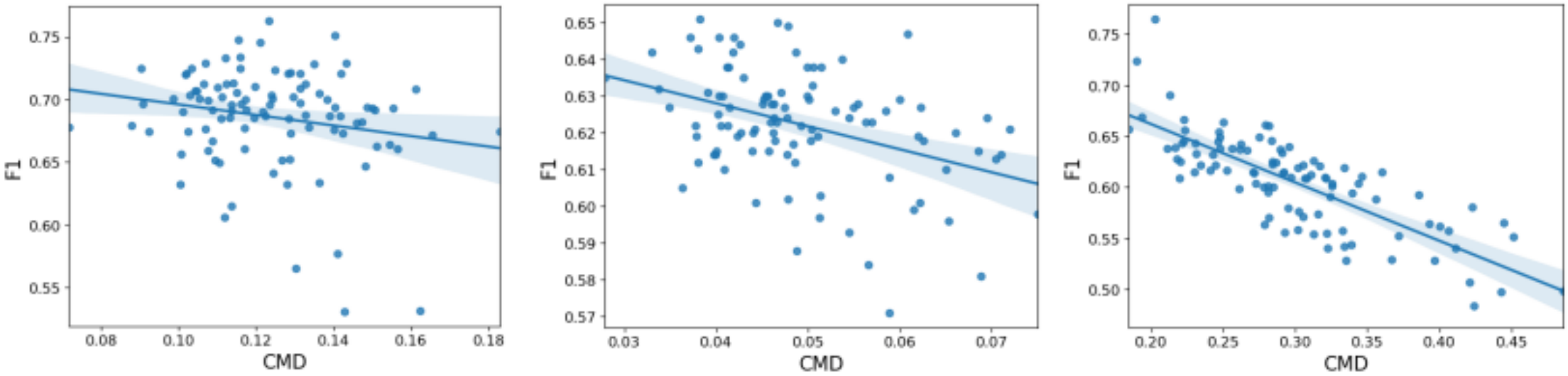
Quantify the distribution shift

- Assume two sets of representation vectors are generated by probability distribution p and q, a valid discrepancy metric measures the distribution shifts, CMD [1] for example,

$$\mathrm{CMD} = \frac{1}{|b-a|}\left\|\mathrm{E}\left(p\right)-\mathrm{E}\left(q\right)\right\|_2 + \sum_{k=2}^{\infty}\frac{1}{|b-a|^k}\left\|c_k\left(p\right)-c_k\left(q\right)\right\|_2,$$

[1] Zellinger, Werner, et al. "Central Moment Discrepancy (CMD) for Domain-Invariant Representation Learning." ICLR, 2016.

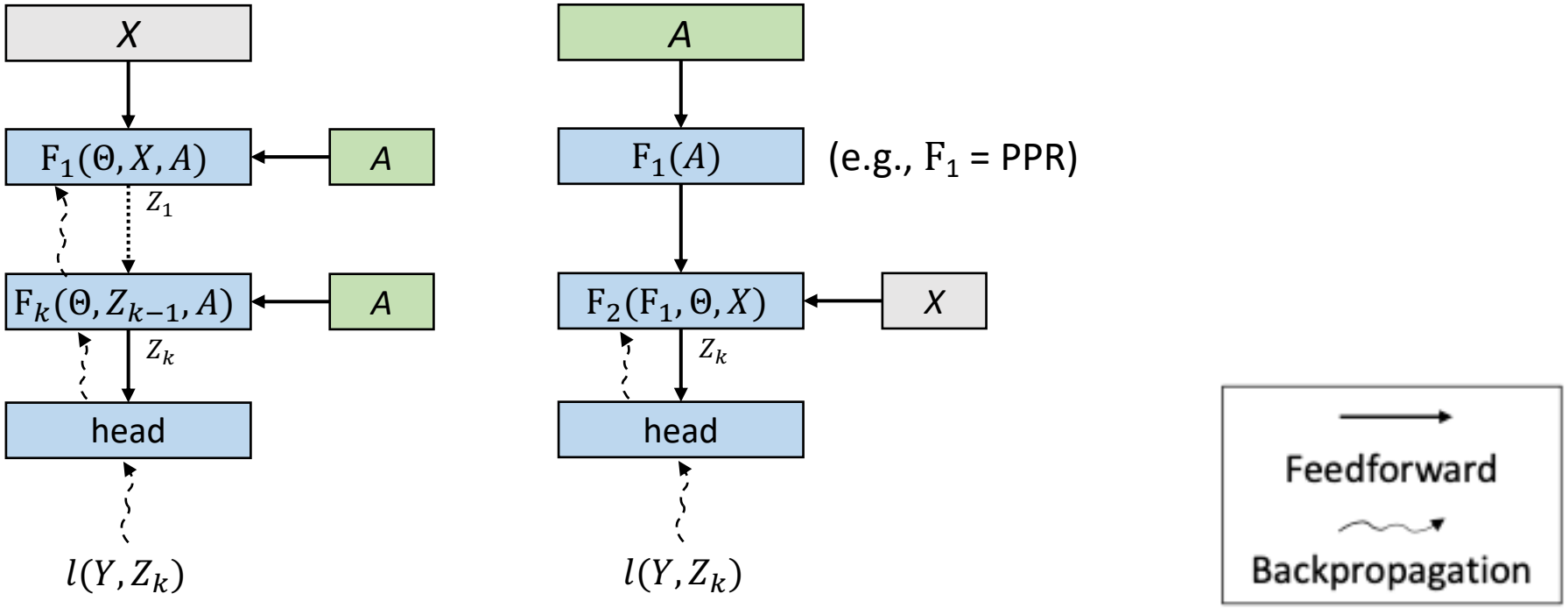
Negative effect of distribution shifts



(a) Cora **(b)** Citeseer **(c)** Pubmed

Distribution shift (CMD) between training and testing data could be a good indicator of performance (F1) !

Two major variants of GNNs



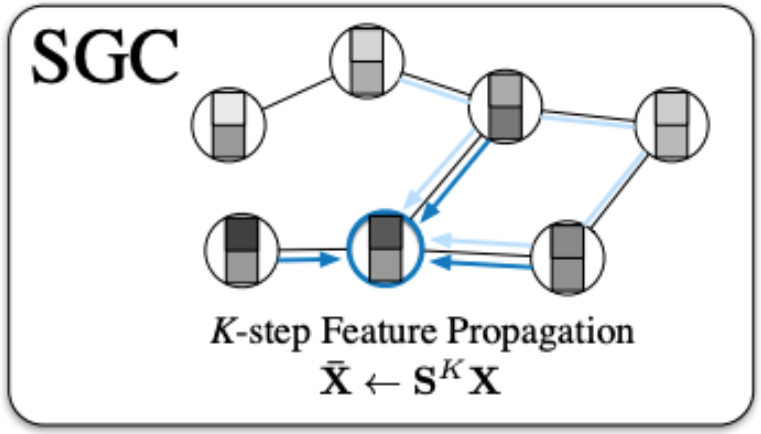
Traditional GNN

Linearized GNN

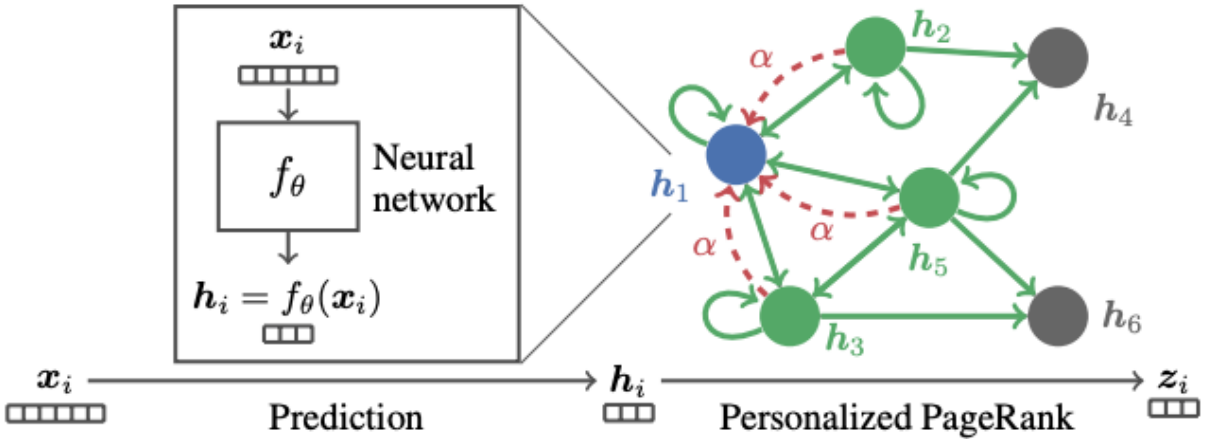
Standard GNNs: the graph inductive bias $\tilde{A}$ is differentiable

Linearized GNNs: the graph inductive bias $\tilde{A}$ is **not** differentiable

Examples of linearized (shallow) models



SGC [1]



APPNP [2], PPRGo [3]

Complexity of neural networks do not grow as number of propagations increase !

[1] Wu, Felix, et al. "Simplifying graph convolutional networks." *ICML*, 2019.
[2] Klicpera, Johannes, Aleksandar Bojchevski, and Stephan Günnemann. "Predict then Propagate: Graph Neural Networks meet Personalized PageRank." *ICLR*, 2018.
[3] Bojchevski, Aleksandar, et al. "Scaling graph neural networks with approximate pagerank." *KDD*, 2020.

Standard GNN – regularization on Z

$$\Phi = F(\Theta, A)$$

- Φ is fully differentiable. We sample an IID data of the same size of training data and minimize the distribution shift between Z_{train} and Z_{IID}

$$\mathcal{L} = \frac{1}{M}\sum_{i} l(y_i, z_i) + \lambda \cdot d(Z_{\text{train}}, Z_{\text{IID}}).$$

$$d_{\text{CMD}}(Z_{\text{train}}, Z_{\text{IID}}) = \frac{1}{b-a}\|\mathbf{E}(Z_{\text{train}}) - \mathbf{E}(Z_{\text{IID}})\| + \sum_{k=2}^{\infty}\frac{1}{|b-a|^k}\|c_k(Z_{\text{train}}) - c_k(Z_{\text{IID}})\|,$$

Linearized GNN – instance re-weighting

$$\Phi = F_2(\Theta, F_1\ (A))$$

$$\mathcal{L} = \frac{1}{M}\beta_i l(y_i, \Phi(h_i)),$$

- We use importance sampling to mitigate the shift, calculate the instance weight via kernel mean matching [1],

$$\min_{\beta_i} \| \frac{1}{M}\sum_{i=1}^{M}\beta_i\psi(h_i) - \frac{1}{M'}\sum_{i=1}^{M'}\psi(h_i')\|^2, \ \textbf{s.t.}\ B_l \leq \beta < B_u$$

[1] Gretton, Arthur, et al. "Covariate shift by kernel mean matching." *Dataset shift in machine learning* 3.4 (2009): 5

Shift-Robust training framework

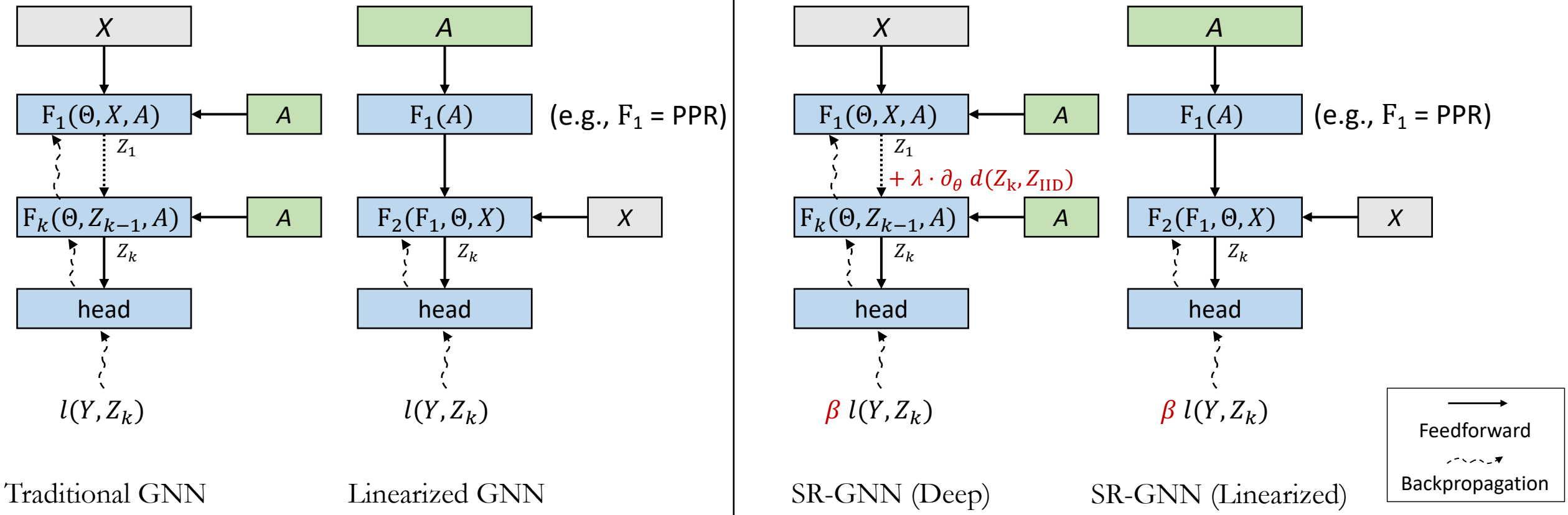
$$\mathcal{L}_{\text{SR-GNN}} = \frac{1}{M}\beta_i l(y_i, \Phi(x_i, A)) + \lambda \cdot d(Z_{\text{train}}, Z_{\text{IID}}).$$

- We choose APPNP [1] (a linearized model) as a concrete example that both techniques can be applied

$$\Phi_{\text{APPNP}} = \underbrace{\left((1-\alpha)^k \tilde{A}^k + \alpha \sum_{i=0}^{k-1} (1-\alpha)^i \tilde{A}^i\right)}_{\text{approximated personalized page rank}} \underbrace{\mathbf{F}(\Theta, X).}_{\text{feature encoder}}$$

[1] Klicpera, Johannes, Aleksandar Bojchevski, and Stephan Günnemann. "Predict then Propagate: Graph Neural Networks meet Personalized PageRank." *ICLR,* 2018.

Shift-Robust training framework



Biased training set creation

- The localized training data in real-world applications is not easy to control the degree of bias. We propose a scalable biased training data generation process based on fast Personalized Page Rank computation [1].

Algorithm 1: Biased Training Set Creation PPR-S(γ, ϵ, α)

Given a class c, label ratio τ, graph size N;
Initialize the biased training set X = { } ;
while *len(X)*$< N \cdot \tau$ **do**
    Sample node i of class c, compute its top-γ entries in $\pi_i^{\text{ppr}}(\epsilon)$ via [2];
    if $\pi_i^{ppr}(\epsilon)$ *has* γ *non-zero entries* **then**
        X.add($\pi_i^{\text{ppr}}(\epsilon)$) ;
    end
end

[1] Andersen, Reid, Fan Chung, and Kevin Lang. "Local graph partitioning using pagerank vectors." FOCS, 2006.

Biased training data example

(a) IID sample **(b)** Biased sample **(c)** PPR-score on biased sample

Figure 1: A biased sample on Cora dataset for one class, orange indicates the training data, red indicates the initial seed used in our PPR-S sampler. The PPR-score is presented in figure (c).

Experimental result on small benchmarks

Table 1: Semi-supervised classification on three different citation networks using biased training samples. Our proposed framework (SR-GNN) outperforms **all** baselines on biased training input.

Method	Cora			Citeseer			PubMed		
	Micro-F1↑	Macro-F1↑	ΔF1 ↓	Micro-F1↑	Macro-F1↑	ΔF1 ↓	Micro-F1↑	Macro-F1↑	ΔF1 ↓
GCN (IID)	80.8 ± 1.6	80.1 ± 1.3	0	70.3 ± 1.9	66.8 ± 1.3	0	79.8 ± 1.4	78.8 ± 1.4	0
Feat.+MLP	49.7 ± 2.5	48.3 ± 2.2	31.1	55.1 ± 1.3	52.7 ± 1.3	25.2	51.3 ± 2.8	41.8 ± 6.2	28.5
Emb.+MLP	57.6 ± 3.0	56.2 ± 3.0	23.2	38.5 ± 1.2	38.6 ± 1.1	31.8	60.4 ± 2.1	56.6 ± 2.0	19.4
DGI	71.7 ± 4.2	69.2 ± 3.7	9.1	62.6 ± 1.6	60.0 ± 1.6	7.6	58.0 ± 5.3	52.4 ± 8.3	21.8
GCN	67.6 ± 3.5	66.4 ± 3.0	13.2	62.7 ± 1.8	60.4 ± 1.6	7.6	60.6 ± 3.8	56.0 ± 6.0	19.2
GAT	58.4 ± 5.7	58.5 ± 5.0	22.4	58.0 ± 3.5	55.0 ± 2.7	12.3	55.2 ± 3.7	46.0 ± 6.4	14.6
SGC	70.2 ± 3.0	68.0 ± 3.8	10.6	65.4 ± 0.8	62.5 ± 0.8	4.9	61.8 ± 4.5	57.4 ± 7.2	18.0
APPNP	71.3 ± 4.1	69.2 ± 3.4	9.5	63.4 ± 1.8	61.2 ± 1.6	6.9	63.4 ± 4.2	58.7 ± 7.0	16.4
w.o. KMM	72.1 ± 4.4	69.8 ± 3.7	8.7	63.9 ± 0.7	61.8 ± 0.6	6.4	69.4± 3.4	67.6 ± 4.0	10.4
w.o. CMD	72.0 ± 3.2	69.5 ± 3.7	8.8	66.1 ± 0.9	63.4 ± 0.9	4.2	66.4 ± 4.0	64.0 ± 5.5	13.4
SR-GNN (Ours)	**73.5 ± 3.3**	**71.4± 3.5**	**7.3**	**67.1 ± 0.9**	**64.0 ± 0.9**	**3.2**	**71.3 ± 2.2**	**70.2 ± 2.4**	**8.5**

SR-GNN outperforms other GNN baselines by accurately eliminating at least (~40%) of the negative effect.

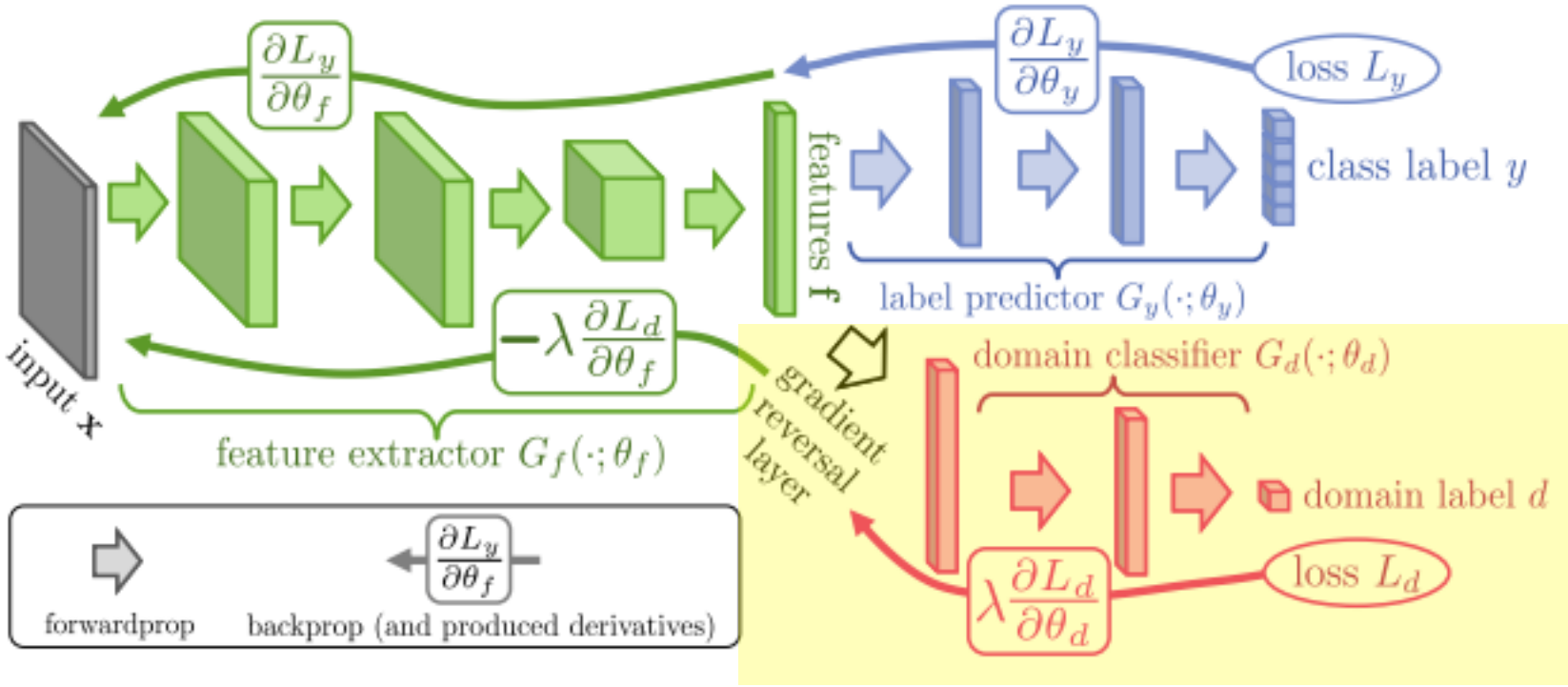
Experimental result on large benchmark

Table 2: Semi-supervised classification on ogb-arxiv varying label ratio.

label(%) / Method	1 % Accuracy	1 % Δ ↓	5 % Accuracy	5 % Δ ↓
GCN (IID)	66.0± 0.6	0	69.1± 0.6	0
Feat.+MLP	45.5± 0.6	21.5	43.7± 0.3	25.4
Emb.+MLP	51.1± 1.3	14.9	56.9± 0.8	13.2
DGI	44.8± 3.0	21.2	49.7± 3.3	19.4
GCN	59.3± 1.2	6.7	65.3 ± 0.6	3.8
GAT	58.6± 1.0	7.4	63.4 ± 1.0	5.7
SGC	59.0± 0.7	7.0	64.2 ± 1.3	4.9
APPNP	59.8± 1.1	6.2	65.1 ± 2.6	4.0
w.o. KMM	60.6± 0.2	5.4	65.1±1.8	4.0
w.o. CMD	61.0± 0.3	5.0	65.8±2.0	3.3
SR-GNN (Ours)	**61.6±0.6**	**4.4**	**66.5±0.6**	**2.6**

SR-GNN improve 2% absolute accuracy and eliminate ~30% of the negative effect by biased data.

Comparison with domain adversarial network



- DANN [1] is a method that uses an adversarial domain classifier to encourage similar feature distributions between different domains.

[1] Ganin, Yaroslav, et al. "Domain-adversarial training of neural networks." *JMLR*, 2016.

Comparison with domain adversarial network

Table 6: Comparison of Domain-Adversarial Neural Network (DANN) and CMD regularizer used in SR-GNN with biased training data.

Method	Cora		Citeseer		PubMed	
	Micro-F1↑	Macro-F1↑	Micro-F1↑	Macro-F1↑	Micro-F1↑	Macro-F1↑
GCN	68.3	67.2	62.4	60.2	59.2	53.8
DANN	69.8	68.5	63.8	61.0	64.8	61.8
CMD (Ours)	**71.0**	**69.4**	**65.0**	**62.3**	**67.5**	**66.2**
APPNP	71.3	69.2	63.9	61.6	64.8	60.4
DANN	71.6	69.5	64.3	61.8	67.8	65.4
CMD (Ours)	**72.4**	**70.1**	**65.0**	**62.4**	**70.4**	**68.7**

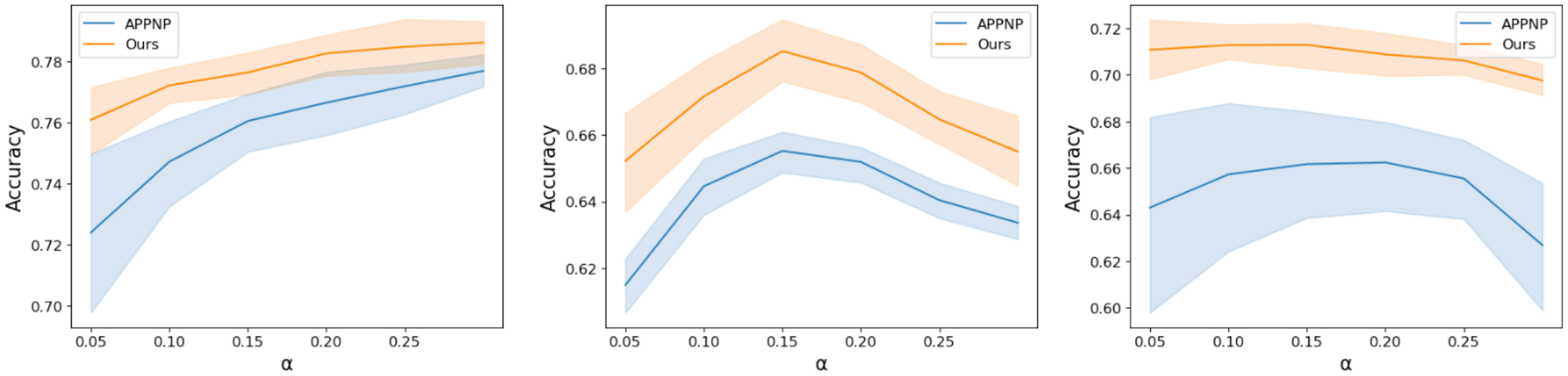
Under semi-supervised setting, the performance of DANN is more sensitive to the domain loss. CMD regularizer performs better with more robust weight selection. Not that CMD regularizer is one component of the proposed SR-GNN.

Apply Shift-Robust on other GNN instances

Table 3: Comparison of baseline and our SR(Shift-Robust) version ($\Delta(\%)$ -relative loss with biased sample) .

Method	Cora			Citeseer			PubMed		
	Micro-F1↑	Macro-F1↑	$\Delta(\%)$	Micro-F1↑	Macro-F1↑	$\Delta(\%)$	Micro-F1↑	Macro-F1↑	$\Delta(\%)$
GCN (IID)	80.8	80.1	0%	70.3	66.8	0%	79.8	78.8	0%
GCN	67.6	66.4	-12%	62.7	60.4	-8%	60.6	56.0	-19%
SR-GCN	**69.6**	**68.2**	-10%	**64.7**	**62.0**	-6%	**67.0**	**65.2**	-13%
DGI (IID)	80.6	79.3	0%	70.8	66.7	0%	77.6	77.0	0%
DGI	71.7	69.2	-9%	62.6	60.0	-8%	58.0	52.4	-20%
SR-DGI	**74.3**	**72.6**	-6%	**65.8**	**62.6**	-6%	**62.0**	**57.8**	-16%

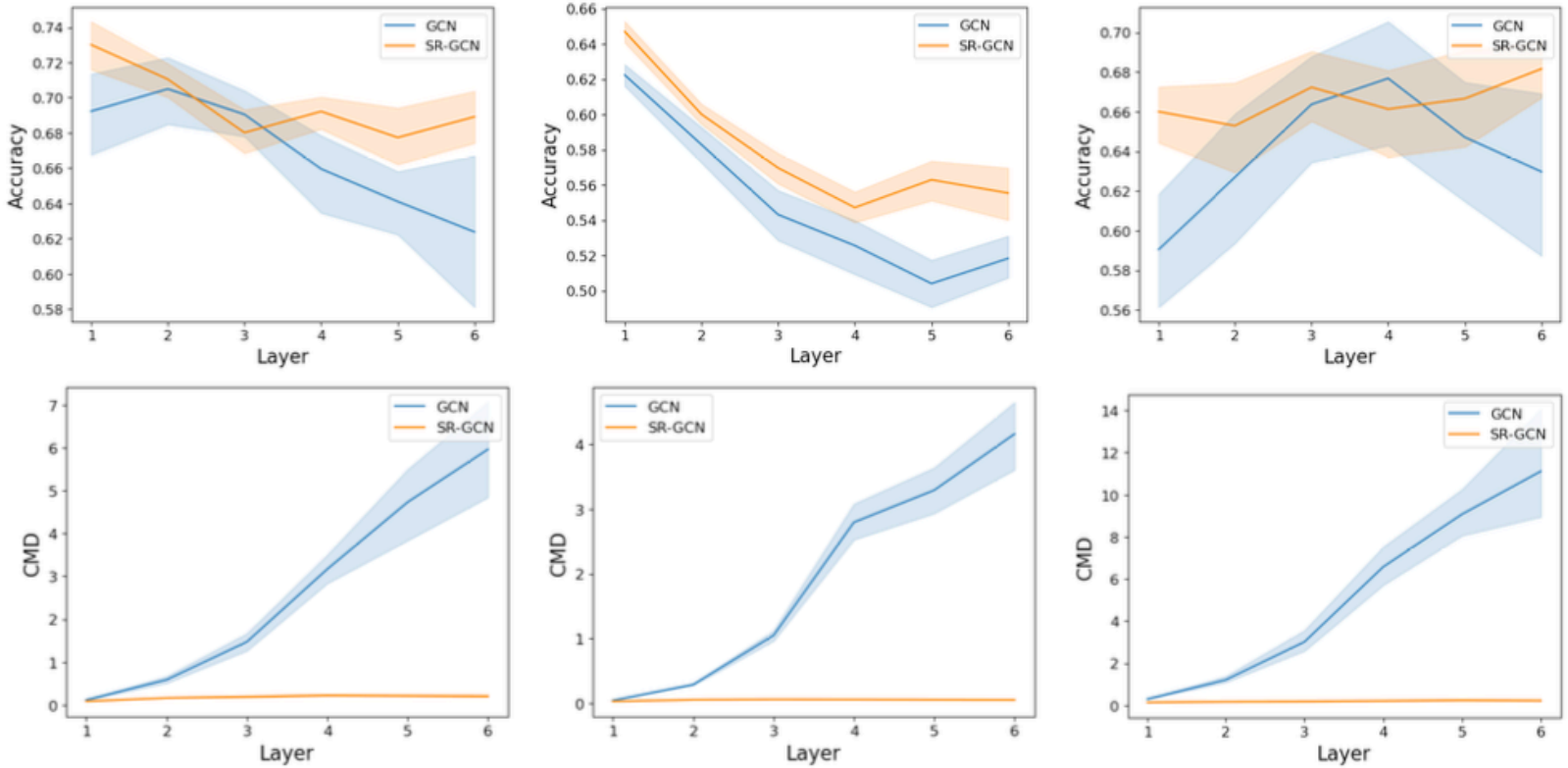
Varying α in biased training set creation



(a) Cora **(b)** Citeseer **(c)** Pubmed

α is the termination probability in PPR. Larger α means more localized PPR-neighbors.

SR-GNN on deeper models

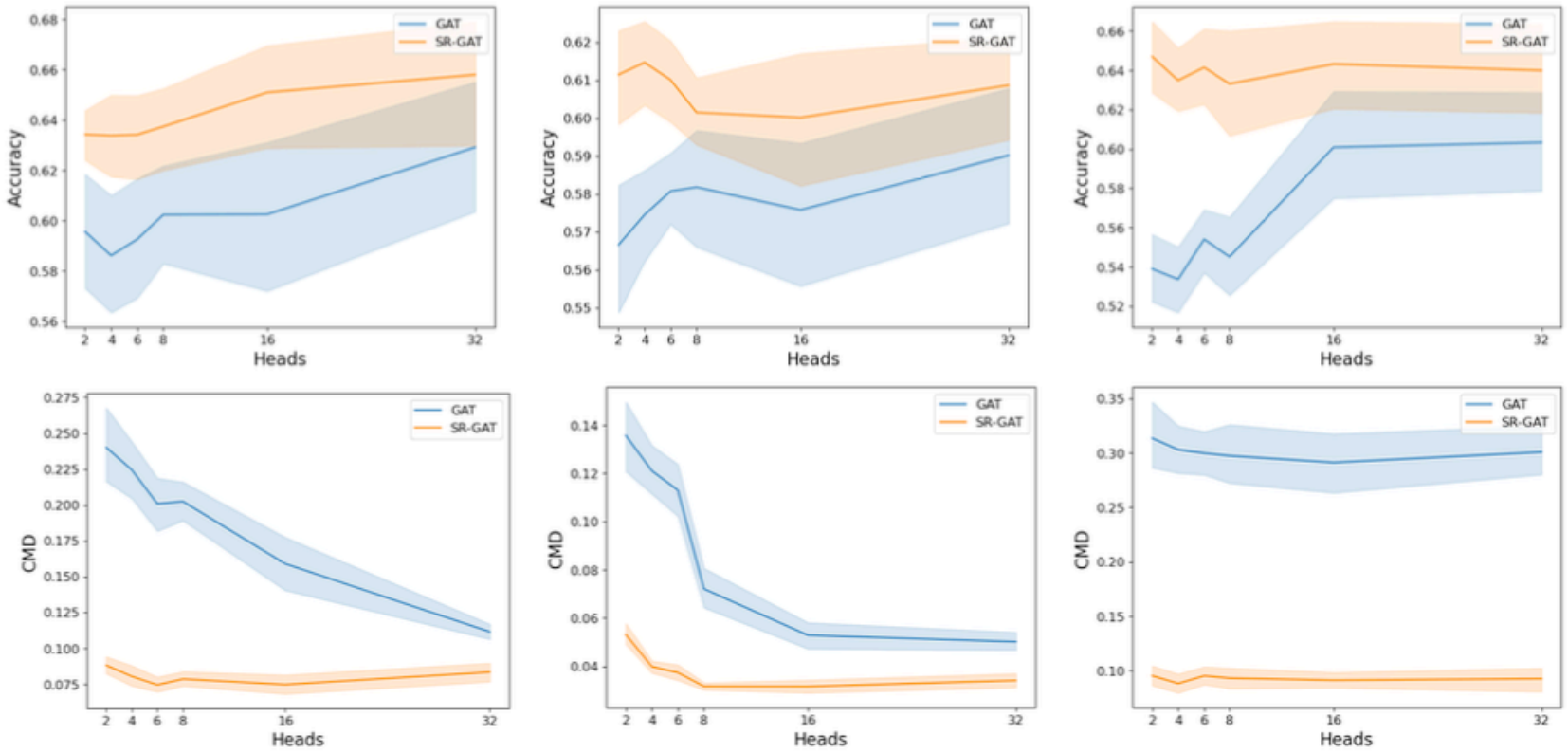


(a) Cora (b) Citeseer (c) Pubmed

Figure 2: Comparison of GCN vs. SR-GCN model performance with the the same parameters. Our shift-robust algorithm boosts the performance (top) consistently by reducing the distribution shifts (bottom).

Larger shift presented in deeper models! SR-GNN consistently works.

SR-GNN on wider models



(a) Cora (b) Citeseer (c) Pubmed

Figure 3: Comparison of GAT vs. SR-GAT model performance under increasing attention heads. Our shift-robust algorithm boosts the performance (upper) consistently by reducing the distribution shifts (lower).

Smaller distributional-shift in wider models.

Outline

- Pre-training / self-supervised learning
  - How to understand GNN generalization during testing time ?
- Design robust GNNs for semi-supervised node classification
  - Handling localized training data
  - **Handling more node-level distribution shifts**
    - Shift-Robust Node Classification via Graph Adversarial Clustering (Preprint)
- Distribution shifts on graph-level tasks

Literature on node-level shift as OOD

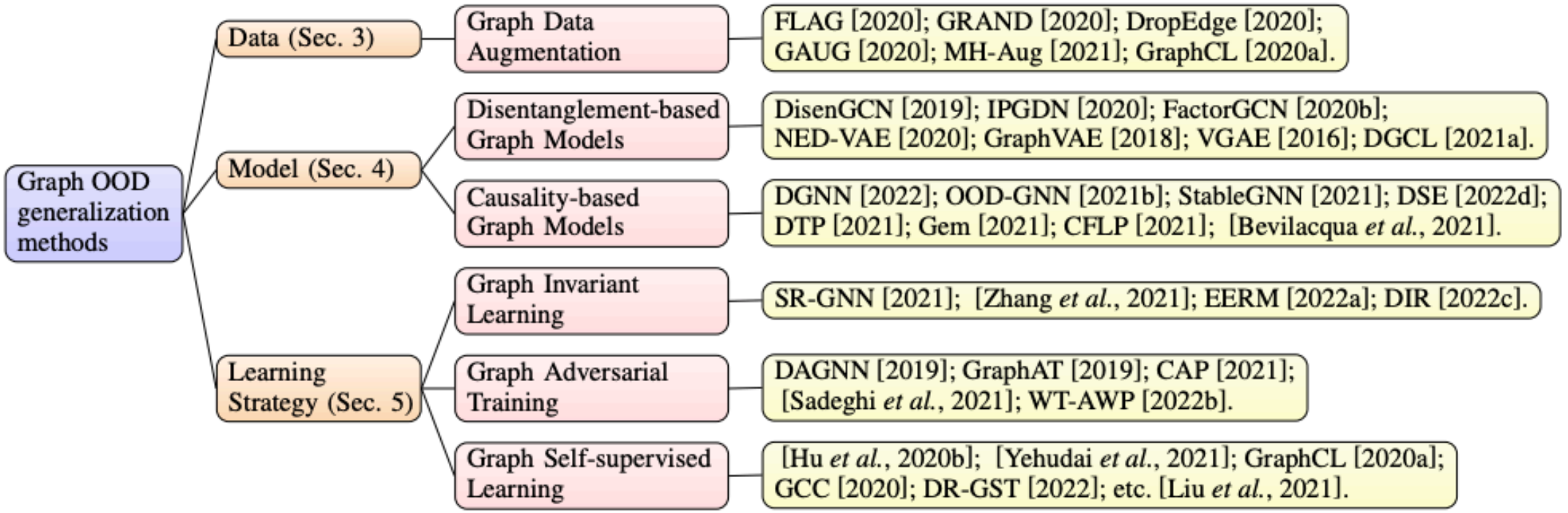


Figure 1: Taxonomy of graph OOD generalization methods.

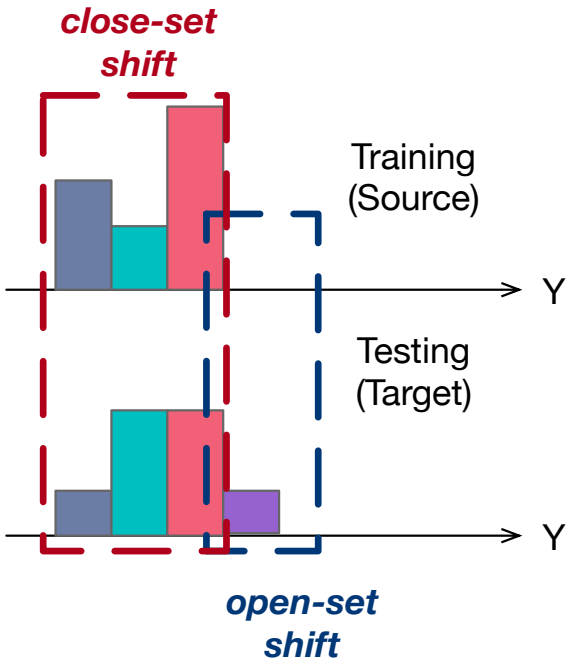
picture credit (https://arxiv.org/pdf/2202.07987.pdf)

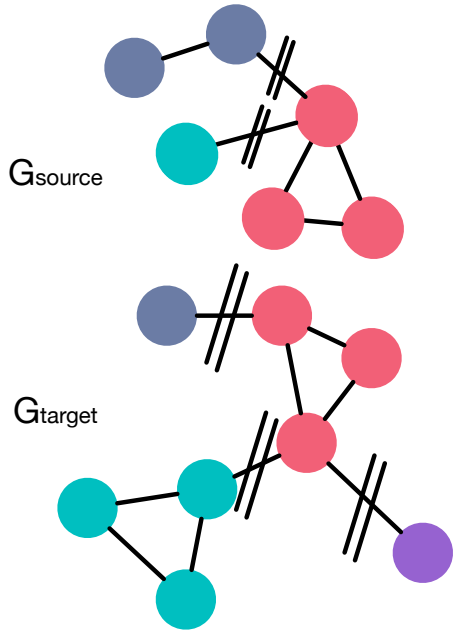
Handling more node-level shift

- Close-set shift
  - Covariate shift

    $\mathbf{Pr}_{\text{train}}(Y|Z) = \mathbf{Pr}_{\text{test}}(Y|Z)$
  - Conditional shift

    $\mathbf{Pr}_{\text{train}}(Y|Z) \neq \mathbf{Pr}_{\text{test}}(Y|Z)$
- Open-set shift
  - Reject unknown classes





	SRNC	PGL [17]	UDA [32]	SRGNN [39]	EERM [34]
open-set	✔	✔			
covariate	✔		✔	✔	✔
conditional	✔	✔			✔

A unified domain adaptation framework

- Distribution shifts are mitigated if we can sample from true target data distribution.

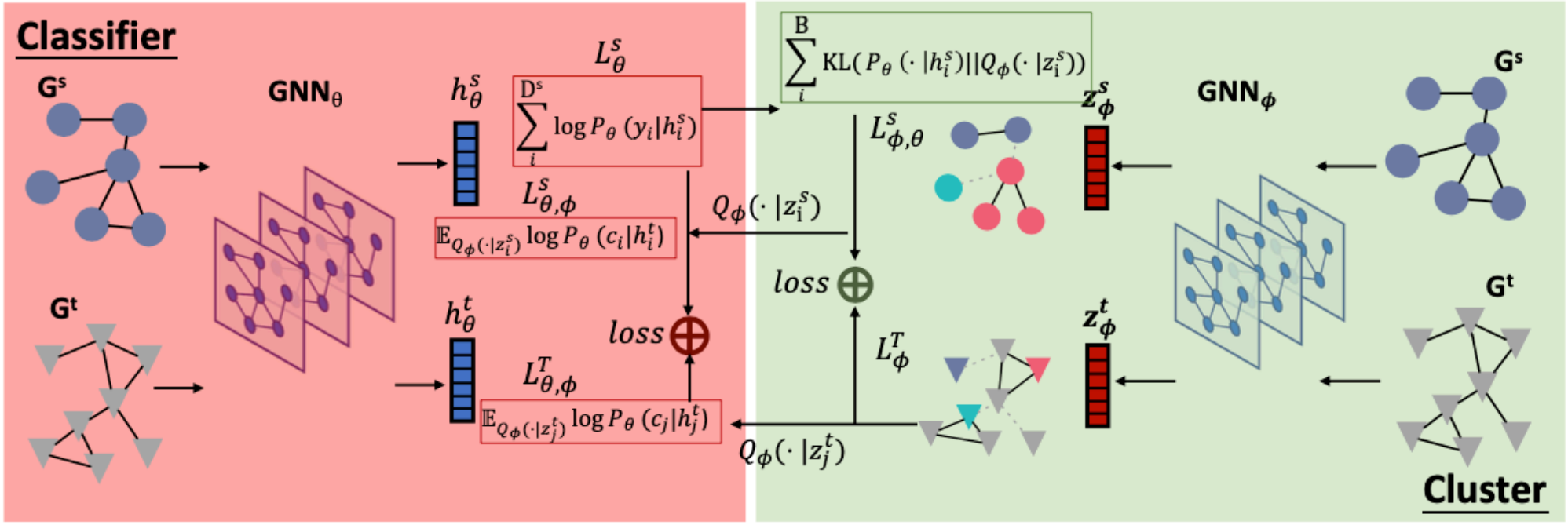
$$\mathcal{L}_\Theta = \underbrace{\sum_{i=1}^{|D^s|} -\log \mathbf{P}_\Theta(y_i^s|h_i^s)}_{\mathcal{L}_\Theta^s} + \underbrace{\mathbb{E}_{(h_j^t,\hat{y}_i{}^t)\sim \mathbf{Q}_\Phi(\cdot|h_j^t)}\left[-\log \mathbf{P}_\Theta(\hat{y}_i{}^t|h_j^t)\right]}_{\mathcal{L}_{\Theta|\Phi}^t},$$

- Can graph homophily help we estimate the target data distribution ?

Definition 3.2 (graph homophily ratio [38]). *The graph homophily ratio* $\tau_h = \frac{\{(u,v):(u,v)\in\mathcal{E}\wedge y_u=y_v\}}{|\mathcal{E}|}$ *is the fraction of edges in a graph which connect nodes with the same class label.*

Graph clustering on target graph

- Same label nodes are densely connected. Ideal graph clustering breaks the heterophily edges and keep the homophily edges.
- We match the identity of class and cluster by Variational Co-training.



Improvements on open-set shift

Table 5: Open-set classification on three different citation networks. Numbers reported are all percentage (%).

Method	Cora			Citeseer			PubMed		
	Micro-F1↑	Macro-F1↑	ΔF1↓	Micro-F1↑	Macro-F1↑	ΔF1↓	Micro-F1↑	Macro-F1↑	ΔF1↓
DGI-IID	83.4± 3.0	81.1 ± 1.7	0	75.3 ± 2.3	68.9 ± 4.9	0	80.2 ± 0.6	80.2 ± 0.5	0
DGI-THS	70.2 ± 4.2	68.9 ± 4.0	13.2	62.9 ± 5.1	56.4 ± 10.7	12.4	63.8 ± 7.4	58.0 ± 3.1	16.4
DGI-DOC	71.2 ± 3.6	70.7 ± 3.0	12.2	56.6 ± 8.1	57.0 ± 8.5	18.7	58.0 ± 4.6	57.4 ± 2.3	22.2
GCN-IID	82.0 ± 3.0	79.7 ± 1.6	0	75.0 ± 2.4	68.0 ± 4.4	0	79.3 ± 0.3	78.8 ± 0.3	0
GCN-THS	72.4 ± 3.7	71.7 ± 3.7	9.6	66.7 ± 3.4	61.5 ± 7.0	8.3	64.2± 2.8	58.9 ± 6.2	15.1
GCN-DOC	72.8 ± 3.4	72.8 ± 3.0	9.2	66.0 ± 5.0	63.8 ± 7.1	9.0	58.5 ± 7.0	47.5 ± 2.0	20.8
GCN-PGL	72.1 ± 4.4	70.9 ± 4.8	9.9	67.0 ± 5.2	60.0 ± 9.4	8.0	63.6 ± 3.8	57.8 ± 7.0	15.7
OpenWGL	66.7 ± 6.1	64.3 ± 5.7	15.3	64.5 ± 3.8	56.1 ± 7.0	11.5	64.2 ± 2.9	64.1 ± 2.5	15.1
SRNC **w.o** Φ	71.7 ± 6.4	70.2 ± 3.6	10.3	65.5 ± 4.7	56.2 ± 4.5	9.5	65.8 ± 1.6	60.5 ± 7.4	13.5
SRNC Ep.1	76.0 ± 4.7	75.2 ± 2.9	6.0	69.2 ± 5.8	60.4 ± 6.0	1.9	67.3 ± 5.1	68.0 ± 3.9	12.0
SRNC	**77.4 ± 4.0**	**75.9 ± 3.6**	**4.6**	**70.7 ± 4.0**	**63.4 ± 7.4**	**4.3**	**69.1 ± 4.4**	**69.4 ± 2.5**	**10.2**

Open-set distribution shift are challenging and most baselines cannot outperform simple threshold.

Improvements on close-set shift

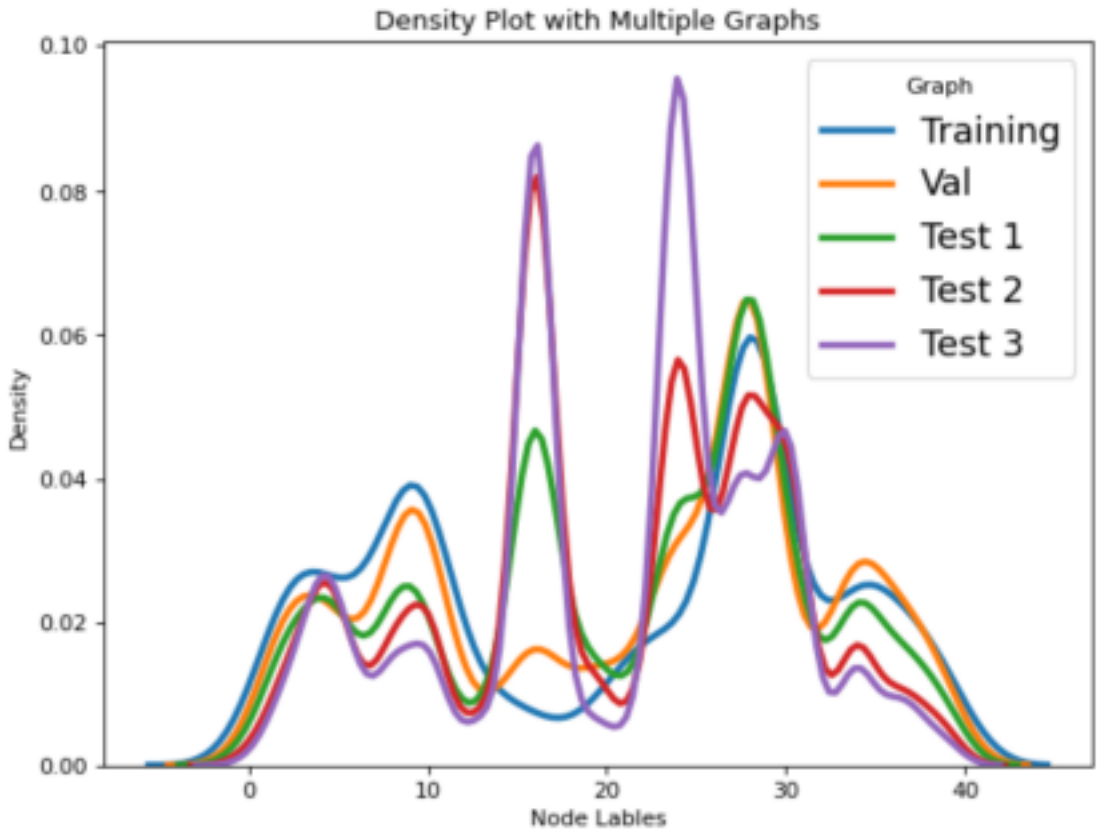


Table 4: Performance under close-set shift on ogb-arxiv.

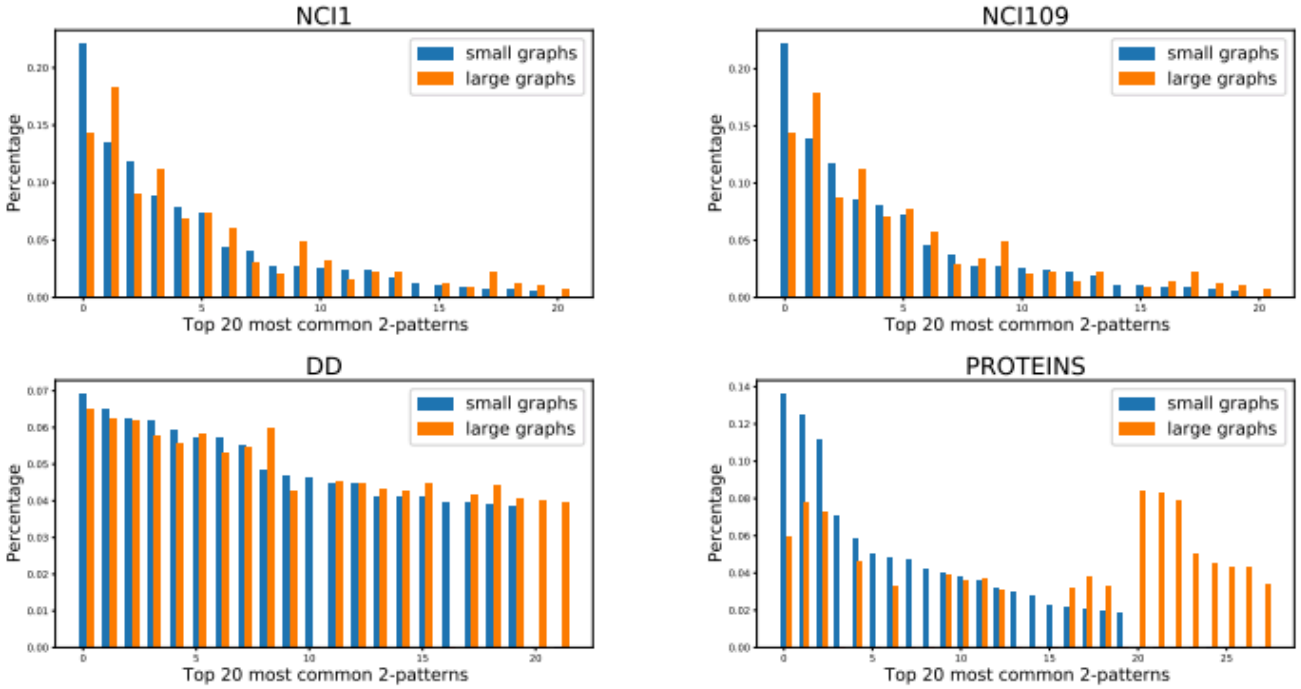
Method	ogb-arxiv		
	2014-2016	2016-2018	2018-2020
DGI	52.6 ± 0.4	48.3 ± 1.9	50.9 ± 1.4
DGI-DANN	48.9 ± 1.5	44.4 ± 3.1	28.2 ± 0.7
DGI-CMD	44.5 ± 0.6	36.5 ± 1.0	31.0 ± 1.9
DGI-SRGNN	50.5 ± 1.8	49.7 ± 2.7	47.7 ± 2.2
GCN	56.2 ± 0.5	55.7 ± 0.8	53.8 ± 1.2
GCN-DANN	54.3 ± 1.0	50.4 ± 3.2	46.2 ± 5.0
GCN-CMD	50.7 ± 0.6	48.7 ± 1.5	50.0 ± 2.3
GCN-SRGNN	54.4 ± 0.6	53.3 ± 1.1	55.0 ± 1.1
GCN-UDA	57.3 ± 0.4	56.5± 0.5	57.5 ± 1.6
GCN-EERM	50.4 ± 1.6	50.4± 2.7	51.0 ± 2.8
SRNC **w.o** Φ	57.3 ± 0.2	58.0 ± 0.8	55.6 ± 1.7
SRNC Ep.1	56.9 ± 0.1	56.0 ± 0.4	54.5 ± 0.1
SRNC	**58.1 ± 0.3**	**58.7 ± 0.8**	**59.1 ± 1.3**

Close-set distribution shift exists in open benchmarks and existing methods can barely improve OOD.

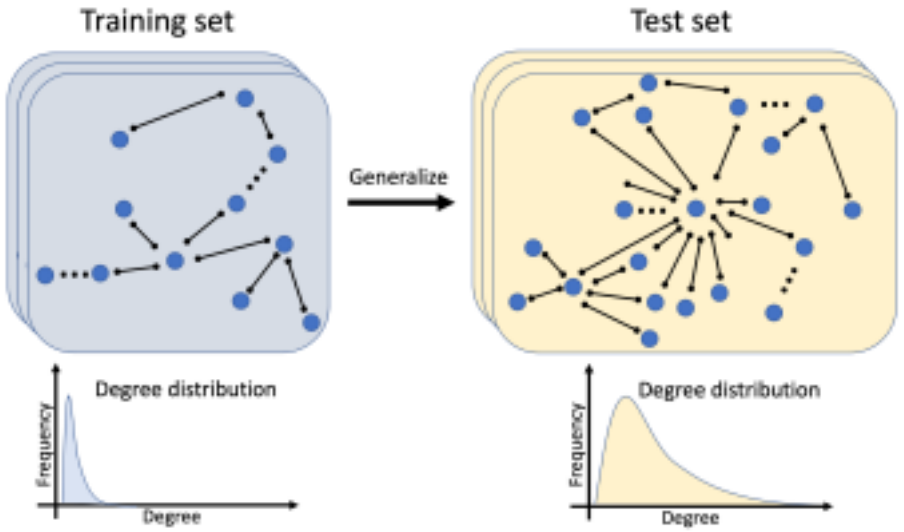
Outline

- Pre-training / self-supervised learning
  - How to understand GNN generalization during testing time ?
- Design robust GNNs for semi-supervised node classification
  - Handling localized training data
  - Handling more node-level distribution shifts
- **Distribution shifts on graph-level tasks**

Distribution shifts on graph-level tasks



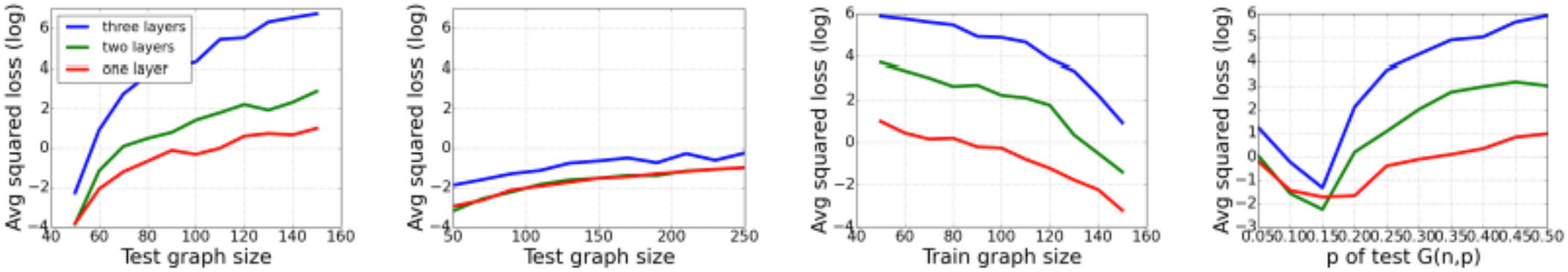
Ego-graph as tokens (same as EGI)



Size Generalization

picture credit (https://arxiv.org/pdf/2112.03806.pdf)

Distribution shifts on graph-level tasks



(a) **varied size and #degree**

(b) **varied size and fixed #degree**

(c) **fixed test**

(d) **generalization**

Theorem: A d-layer GNN has constant outputs for same d-ego graph.

Excessive depth is bad:

if task is solvable by d-layer GNN, there exists d+3 layer GNN fits training distribution but fails on testing distribution.

picture credit (https://arxiv.org/pdf/2112.03806.pdf)

Acknowledgement

- The fantastic Deep Graph Library !
- AWS Machine Learning Research Awards ($50K AWS credits) !!
- My collaborators
  - Jiawei Han, Bryan Perrozzi, Da Zheng, Christos Faloutsos, Luna Dong, Carl J. Yang, Chao Zhang, Yu Shi, Chanyoung Park, Natalia Ponomareva, Haonan Wang

References

- [1] Zhu, Qi, et al. “Shift-robust gnns: Overcoming the limitations of localized graph training data.” *Advances in Neural Information Processing Systems* 34 (2021). [code]
- [2] Zhu, Qi, et al. “Transfer learning of graph neural networks with ego-graph information maximization.” Advances in Neural Information Processing Systems 34 (2021). [code]
- [3] Zhu, Qi, et al. "Shift-Robust Node Classification via Graph Adversarial Clustering." arXiv preprint arXiv:2203.15802 (2022).

QA & Discussion