清華大學 Tsinghua University

media and network lab

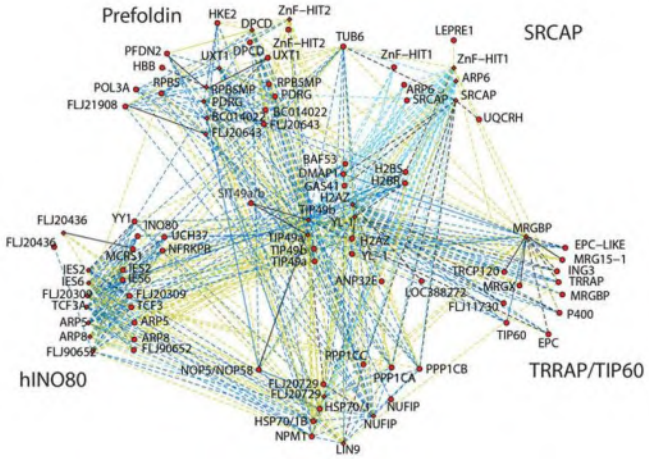
Out-of-distribution Generalized Graph Neural Network

Ziwei Zhang
Tsinghua University

2022.10.29@LOGS

Graphs/Networks

Social Network

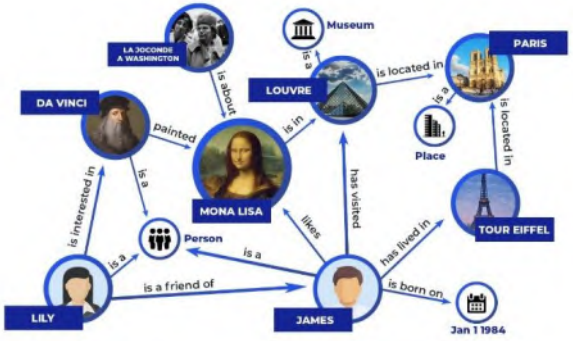


Biology

Logistics

Transaction

Internet of Things



Knowledge Graphs

Picture credit: web search engines

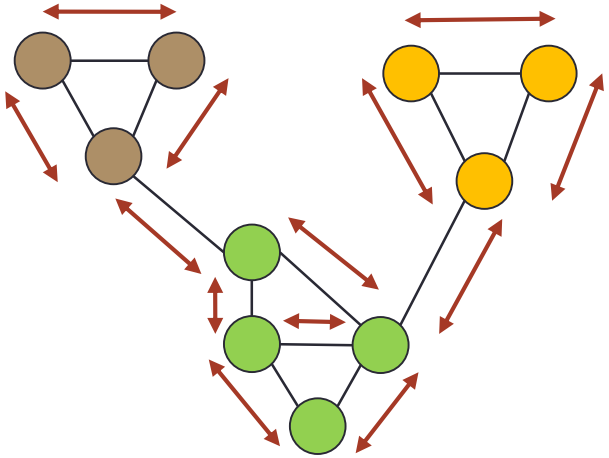
Message-passing GNNs

- Background: message-passing framework of GNNs

$$\mathbf{m}_i^{(l)} = \mathrm{AGG}(\{\mathbf{h}_j^{(l)}, \forall j \in \tilde{\mathcal{N}}_i\})$$

$$\mathbf{h}_i^{(l+1)} = \mathrm{UPDATE}([\mathbf{h}_i^{(l)}, \mathbf{m}_i^{(l)}])$$

  - $\mathrm{h}_i^{(l)}$: the representation of node v_i in the l^{th} layer
  - $\mathrm{m}_i^{(l)}$: the message vector of node v_i in the l^{th} layer

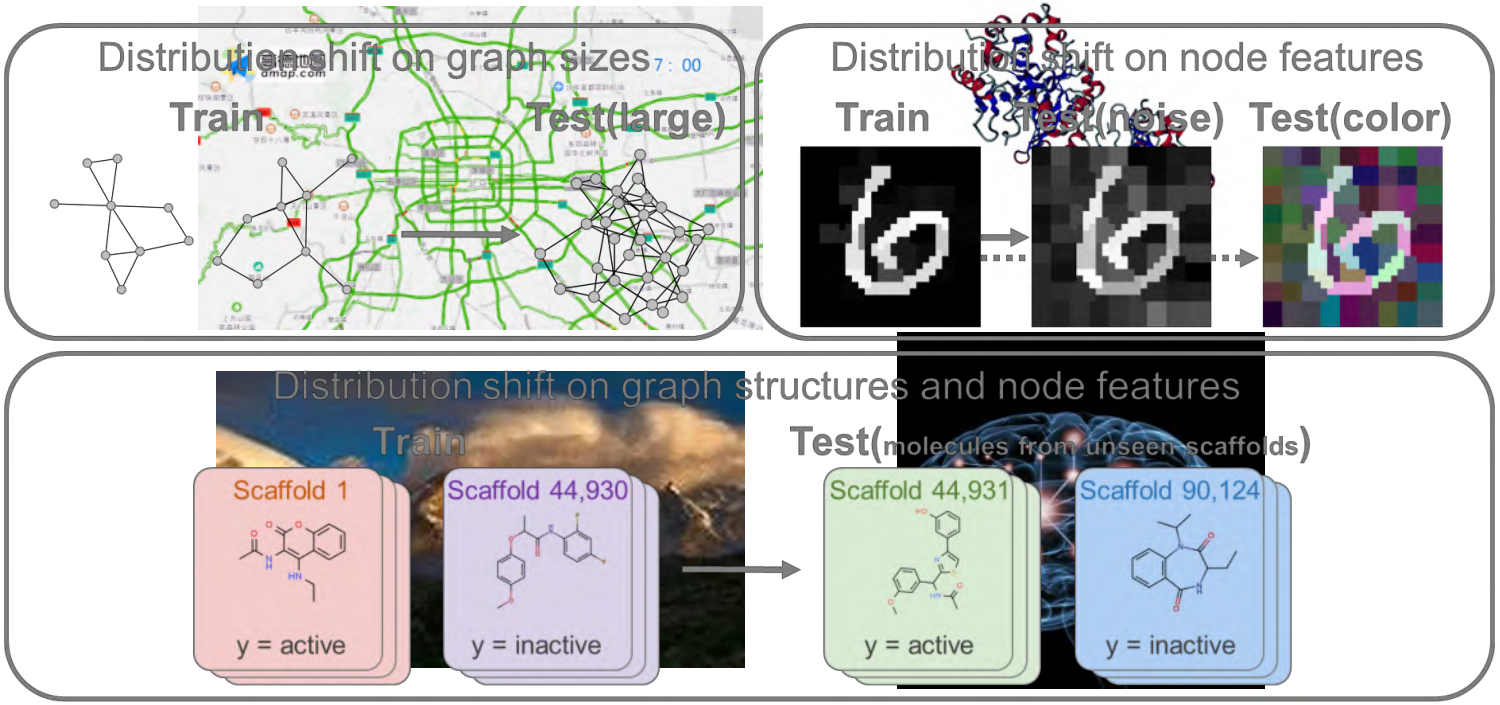


  - Nodes exchange messages to update representations

Existing GNNs have shown successes in many graph applications

However, real graphs are challenging…

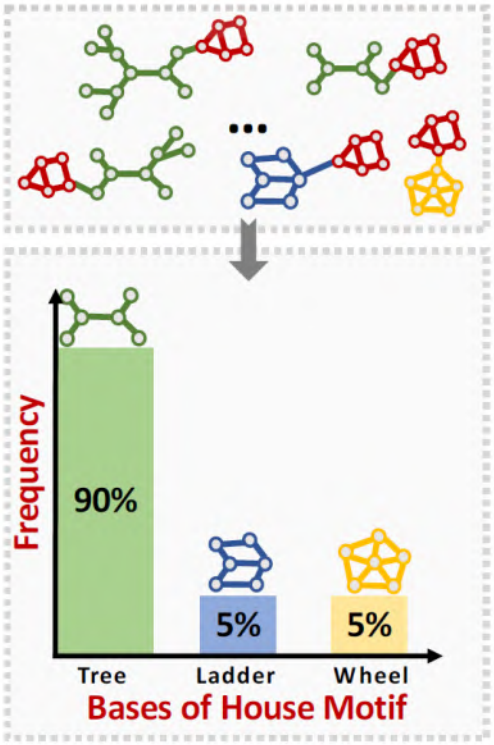
- Many real graphs exist in dynamic and open environments
  - Open: “emerging new classes, decremental/incremental features, changing data distributions, varied learning objectives, etc.” (Zhi-hua Zhou)
  - I.e., distribution shifts naturally exist in graph data

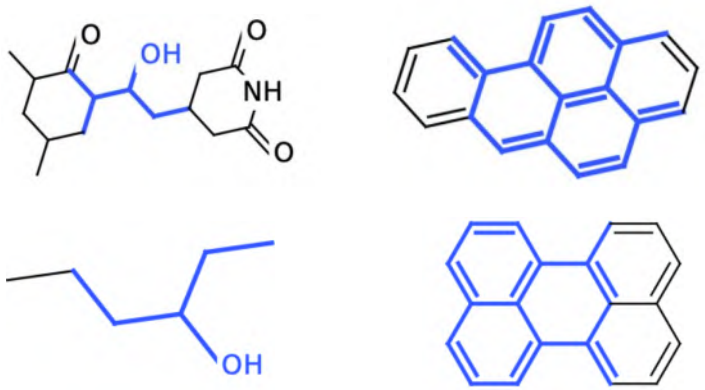


Out-of-distribution generalized GNNs are critically needed!

Main Challenge for Handling Distribution Shifts

- Why existing GNNs fail to achieve OOD generalization?
- Our answer: **spurious correlations**
  - GNNs tend to exploit statistical correlations in the training set
  - But spurious correlations cannot generalize under distribution shifts

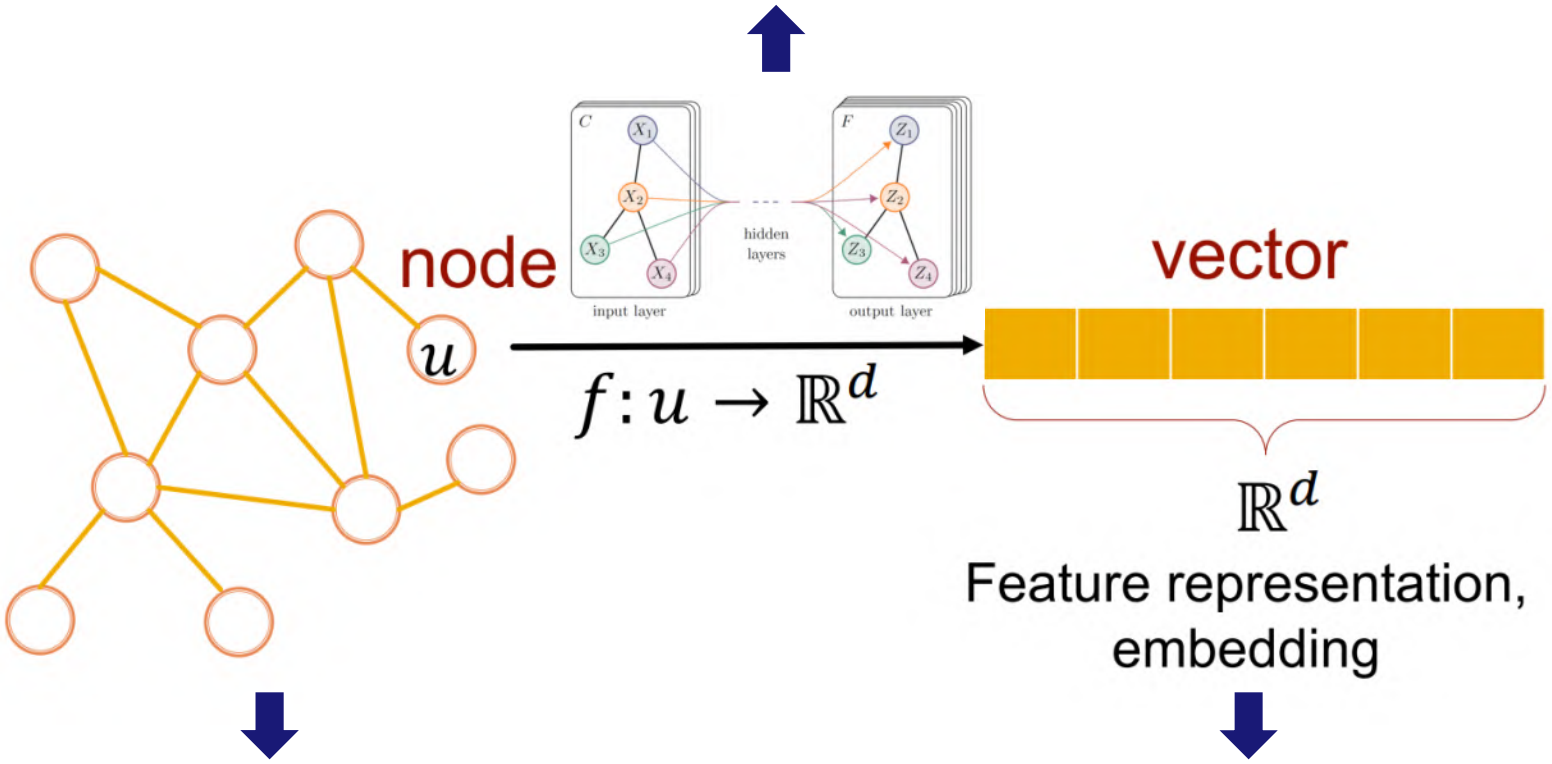




Picture credit: Duvenaud et al., NeurIPS 2015; Ying-Xin Wu, et al., ICLR 2022

Handling Distribution Shifts

How to handle distribution shifts in GNN architectures?



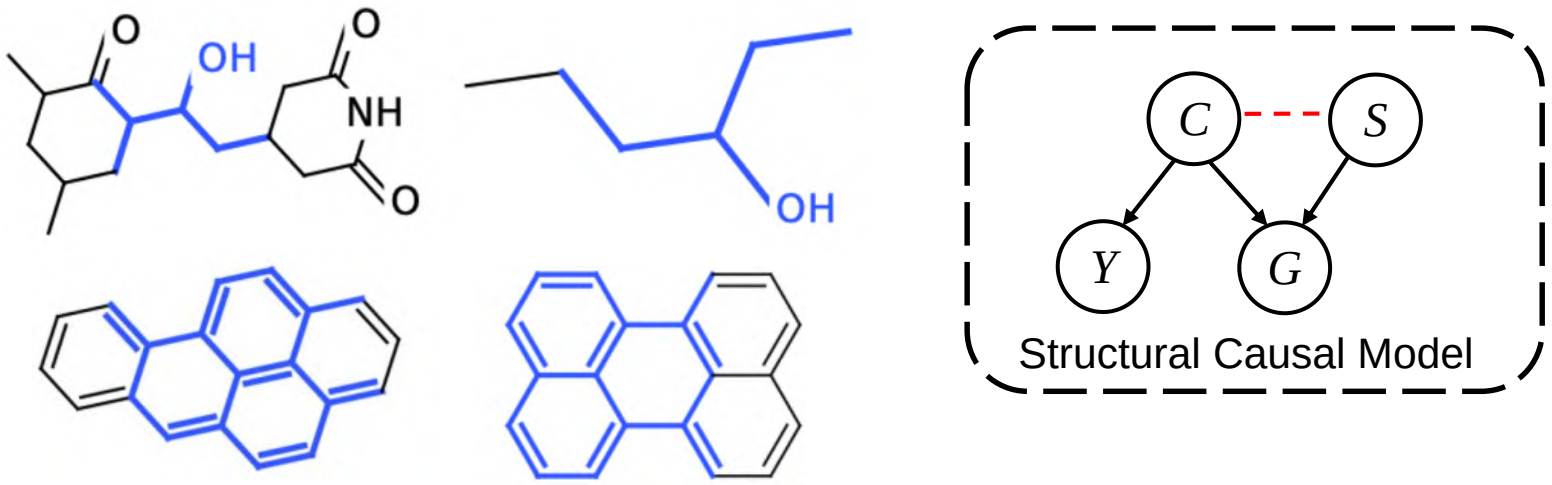
How to handle distribution shifts in the topology space?

How to handle distribution shifts in the vector space?

Picture credit: Kipf and Welling (GCN, ICLR 2017), Jure Leskovec (CS224w, Stanford Univ.)

Out-of-Distribution Generalized GNN (OOD-GNN)

- How to get rid of spurious correlations in node representations?
  - Main idea: decorrelations
  - Remove the statistical dependence of truly predictive (causal) information and spurious (non-causal) information by sampling reweighting, i.e., assign each sample (graph) a weight)
  - More theoretical backgrounds: direct confounder balancing



OOD-GNN: Out-of-Distribution Generalized Graph Neural Network. ***TKDE, 2022.***

OOD-GNN: HSIC

- In practice: encourage to eliminate statistical dependence of all dimensions
  - Since we do not know which ones are causal and spurious
- To get rid of spurious correlations, we expect $\mathbf{Z}_{*i} \perp\!\!\!\perp \mathbf{Z}_{*j}, \forall i, j \in [1, d], i \neq j$
- We adopt Hilbert-Schmidt Independence Criterion (HSIC) measured as:

Proposition 1. *Assume* $\mathbb{E}[k_{\mathbf{Z}_{*i}}(\mathbf{Z}_{*i}, \mathbf{Z}_{*i})] < \infty$ *and* $\mathbb{E}[k_{\mathbf{Z}_{*j}}(\mathbf{Z}_{*j}, \mathbf{Z}_{*j})] < \infty$, *and* $k_{\mathbf{Z}_{*i}} k_{\mathbf{Z}_{*j}}$ *is a characteristic kernel, then*

$$\mathrm{HSIC}(\mathbf{Z}_{*i}, \mathbf{Z}_{*j}) = 0 \Leftrightarrow \mathbf{Z}_{*i} \perp\!\!\!\perp \mathbf{Z}_{*j}.$$

- However, calculating HSIC is intractable. We adopt a practical version as:

$$\min \ \|\widehat{C}_{\mathbf{Z}_{*i}, \mathbf{Z}_{*j}}\|_{\mathrm{F}}^2$$

$$\widehat{C}_{\mathbf{Z}_{*i}, \mathbf{Z}_{*j}} = \frac{1}{N^{tr}-1} \sum_{n=1}^{N^{tr}} \left[\left(f(\mathbf{Z}_{ni}) - \frac{1}{N^{tr}} \sum_{m=1}^{N^{tr}} f(\mathbf{Z}_{mi}) \right)^{\top} \cdot \left(g(\mathbf{Z}_{nj}) - \frac{1}{N^{tr}} \sum_{m=1}^{N^{tr}} g(\mathbf{Z}_{mj}) \right) \right]$$

where $f(\cdot)$and $g(\cdot)$ are the random Fourier features function:

$$f(\mathbf{Z}_{*i}) := (f_1(\mathbf{Z}_{*i}), f_2(\mathbf{Z}_{*i}), \ldots, f_Q(\mathbf{Z}_{*i})),$$
$$g(\mathbf{Z}_{*j}) := (g_1(\mathbf{Z}_{*j}), g_2(\mathbf{Z}_{*j}), \ldots, g_Q(\mathbf{Z}_{*j})),$$

$$f_q(\mathbf{Z}_{*i}), g_q(\mathbf{Z}_{*j}) \in \mathcal{H}_{\mathrm{RFF}}, \forall q \in [1, Q]. \ \mathcal{H}_{\mathrm{RFF}} = \{h : x \to \sqrt{2}\cos(wx + \phi) | w \sim \mathcal{N}(0, 1), \phi \sim \mathrm{Uniform}(0, 2\pi)\}$$

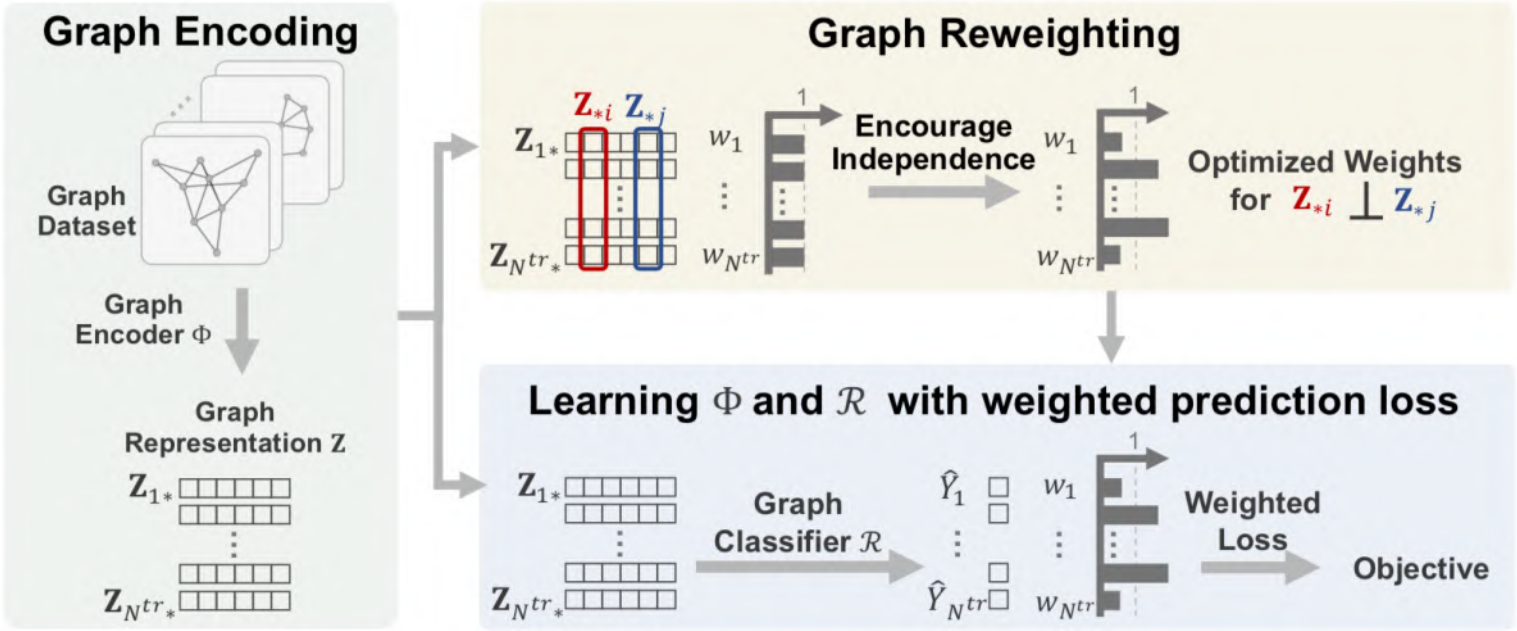
OOD-GNN: Out-of-Distribution Generalized Graph Neural Network. ***TKDE, 2022.***

OOD-GNN: Optimization

- Optimization objectives: jointly optimize weights

$$\Phi^*, \mathcal{R}^* = \operatorname{argmin}_{\Phi,\mathcal{R}} \sum_{n=1}^{N^{tr}} w_n \ell\left(\mathcal{R} \circ \Phi\left(G_n\right), \mathbf{Y}_n\right),$$

$$\mathbf{W}^* = \operatorname{argmin}_{\mathbf{W}} \sum_{1 \le i < j \le d} \|\widehat{C}^{\mathbf{W}}_{\mathbf{Z}_{*i}, \mathbf{Z}_{*j}}\|_{\mathrm{F}}^2,$$



OOD-GNN: Out-of-Distribution Generalized Graph Neural Network. ***TKDE, 2022.***

OOD-GNN: Experiments

- Setup: 14 graph datasets, various kinds of domains/shifts
- Results:

	TRIANGLES		MNIST-75SP		
	Train	Test(large)	Train	Test(noise)	Test(color)
GCN	28.3±0.6	21.3±1.9	51.7±1.0	26.5±1.4	27.0±1.3
GCN-virtual	32.4±0.6	17.0±1.8	55.1±2.3	26.0±1.5	26.1±1.8
GIN	34.7±0.7	22.2±1.9	67.6±0.8	27.9±2.5	34.3±4.4
GIN-virtual	34.2±0.6	17.6±1.7	66.7±0.9	25.7±2.9	33.4±1.2
FactorGCN	10.6±1.6	4.2±0.9	46.7±1.2	19.7±1.4	24.8±1.3
Γ_{GIN}	23.5±3.4	18.0±2.0	47.5±1.3	23.7±1.4	28.3±1.8
PNA	**43.7±3.6**	16.8±2.4	**83.0±0.9**	22.8±7.3	29.2±6.3
TopKPool	28.3±0.3	22.0±0.2	61.0±3.7	17.0±1.0	16.9±1.5
SAGPool	26.7±1.0	23.7±0.7	60.2±1.3	19.6±3.4	20.1±3.7
OOD-GNN	29.9±0.7	**25.1±0.8**	63.2±1.1	**31.5±0.9**	**38.5±1.5**

	COLLAB_{35}	PROTEINS_{25}	$\text{D\&D}_{200}$	$\text{D\&D}_{300}$
# Train/Test graphs	500/4500	500/613	462/716	500/678
#Nodes Train	32-35	4-25	30-200	30-300
#Nodes Test	32-492	6-620	201-5748	30-5748
GCN	65.9±3.4	75.1±2.2	29.2±8.2	71.9±3.6
GCN-virtual	61.5±1.6	70.4±3.7	41.6±8.0	71.6±4.4
GIN	55.5±4.9	74.0±2.7	43.0±8.3	67.8±4.3
GIN-virtual	54.8±2.7	66.0±7.5	46.7±4.5	72.1±4.3
FactorGCN	51.0±1.3	63.5±4.8	42.3±3.1	55.9±1.6
Γ_{GIN}	65.2±2.3	62.6±3.6	59.6±4.9	74.2±2.8
PNA	59.6±5.5	71.4±3.4	47.3±6.8	70.1±2.1
TopKPool	52.8±1.0	64.9±3.0	34.6±5.6	69.3±3.6
SAGPool	67.0±1.7	76.2±0.7	54.3±5.0	78.4±1.1
OOD-GNN	**67.2±1.8**	**78.4±0.9**	**60.3±4.5**	**80.1±1.0**

OOD-GNN: Out-of-Distribution Generalized Graph Neural Network. ***TKDE, 2022.***

OOD-GNN: Experiments

- Setup: 14 graph datasets, various kinds of domains/shifts
- Results:

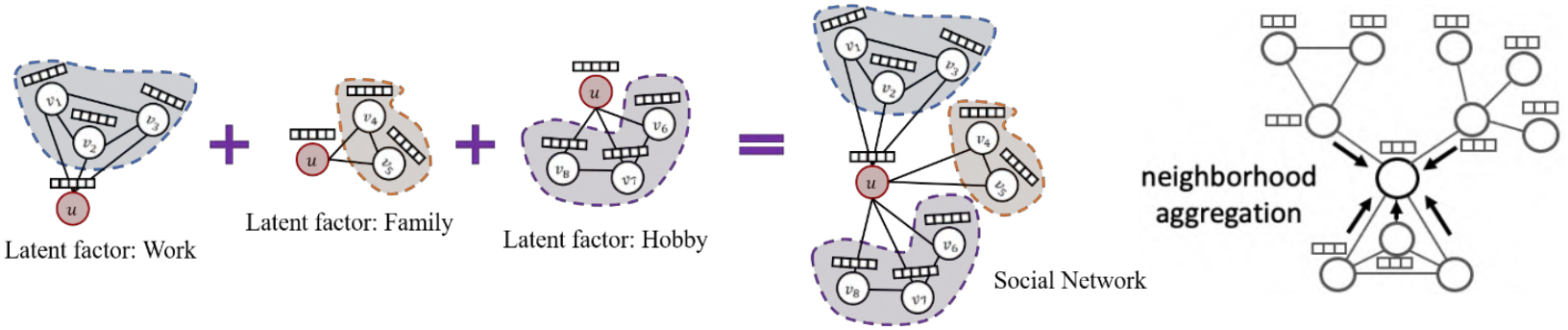
TABLE 5: Results on nine Open Graph Benchmark (OGB) datasets. We report the ROC-AUC (%) for classification tasks and RMSE for regression tasks with the standard deviation on the **test set** of all methods. None of the baseline methods is consistently competitive across all datasets, while our proposed method shows impressive performance. (↑) means that higher values indicate better results, and (↓) represents the opposite.

	TOX21	BACE	BBBP	CLINTOX	SIDER	TOXCAST	HIV	ESOL	FREESOLV
Metric	ROC-AUC (↑)							RMSE (↓)	
GCN	75.3±0.7	79.2±1.4	68.9±1.5	91.3±1.7	59.6±1.8	63.5±0.4	76.1±1.0	1.11±0.03	2.64±0.24
GCN-virtual	77.5±0.9	68.9±7.0	67.8±2.4	88.6±2.1	59.8±1.5	66.7±0.5	76.0±1.2	1.02±0.10	2.19±0.12
GIN	74.9±0.5	73.0±4.0	68.2±1.5	88.1±2.5	57.6±1.4	63.4±0.7	75.6±1.4	1.17±0.06	2.76±0.35
GIN-virtual	77.6±0.6	73.5±5.2	69.7±1.9	84.1±3.8	57.6±1.6	66.1±0.5	77.1±1.5	1.00±0.07	2.15±0.30
FactorGCN	57.8±2.1	70.0±0.6	54.1±1.1	64.2±2.1	53.3±1.7	51.2±0.8	57.1±1.5	3.39±0.15	5.69±0.32
Γ_{GIN}	52.3±1.1	54.5±0.9	51.8±1.5	52.2±1.1	51.3±1.1	50.7±0.5	51.3±0.9	4.15±0.10	7.34±0.12
PNA	71.5±0.5	77.4±2.1	66.2±1.2	81.2±2.0	59.6±1.1	60.6±0.2	79.1±1.3	0.94±0.02	2.92±0.16
TopKPool	75.6±0.9	76.9±2.4	68.6±1.1	86.9±1.1	60.6±1.5	64.7±0.1	76.7±1.1	1.17±0.03	2.08±0.10
SAGPool	74.7±3.1	76.6±1.0	69.3±2.1	88.7±1.0	61.3±1.3	64.8±0.2	77.7±1.3	1.22±0.05	2.28±0.12
OOD-GNN	**78.4±0.8**	**81.3±1.2**	**70.1±1.0**	**91.4±1.3**	**64.0±1.3**	**68.7±0.3**	**79.5±0.9**	**0.88±0.05**	**1.81±0.14**

OOD-GNN: Out-of-Distribution Generalized Graph Neural Network. ***TKDE, 2022.***

Independence-promoted Disentangled Graph Contrastive Learning (IDGCL)

- Except the reweighting, OOD-GNN performs like a normal GNN
  - The formation of a graph is typically driven by many entangled latent factors



→ Can we disentangle latent factors in the message passing?

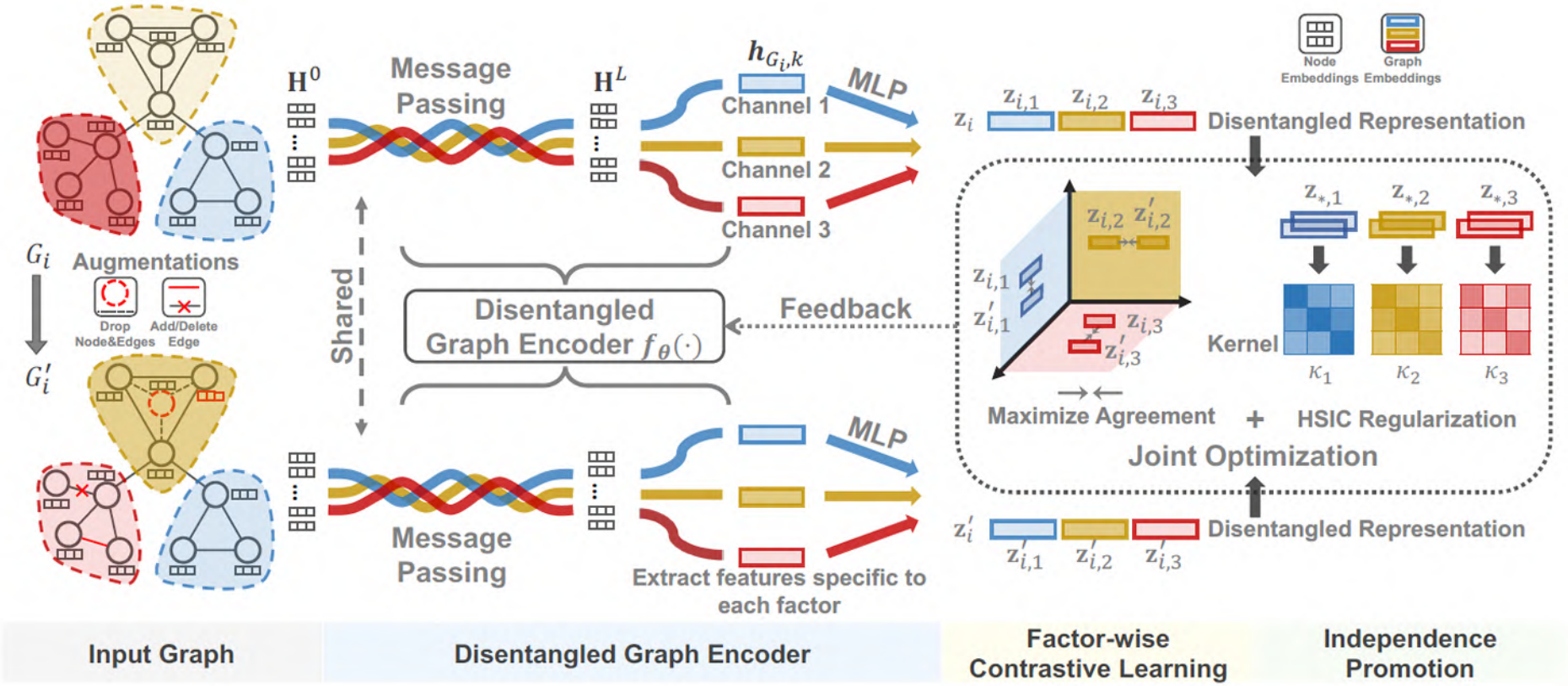
  - The graph labels can be extremely scarce for many graph datasets/scenarios

→ Can we design self-supervised learning frameworks?

Disentangled Graph Contrastive Learning with Independence Promotion. ***TKDE, 2022.***

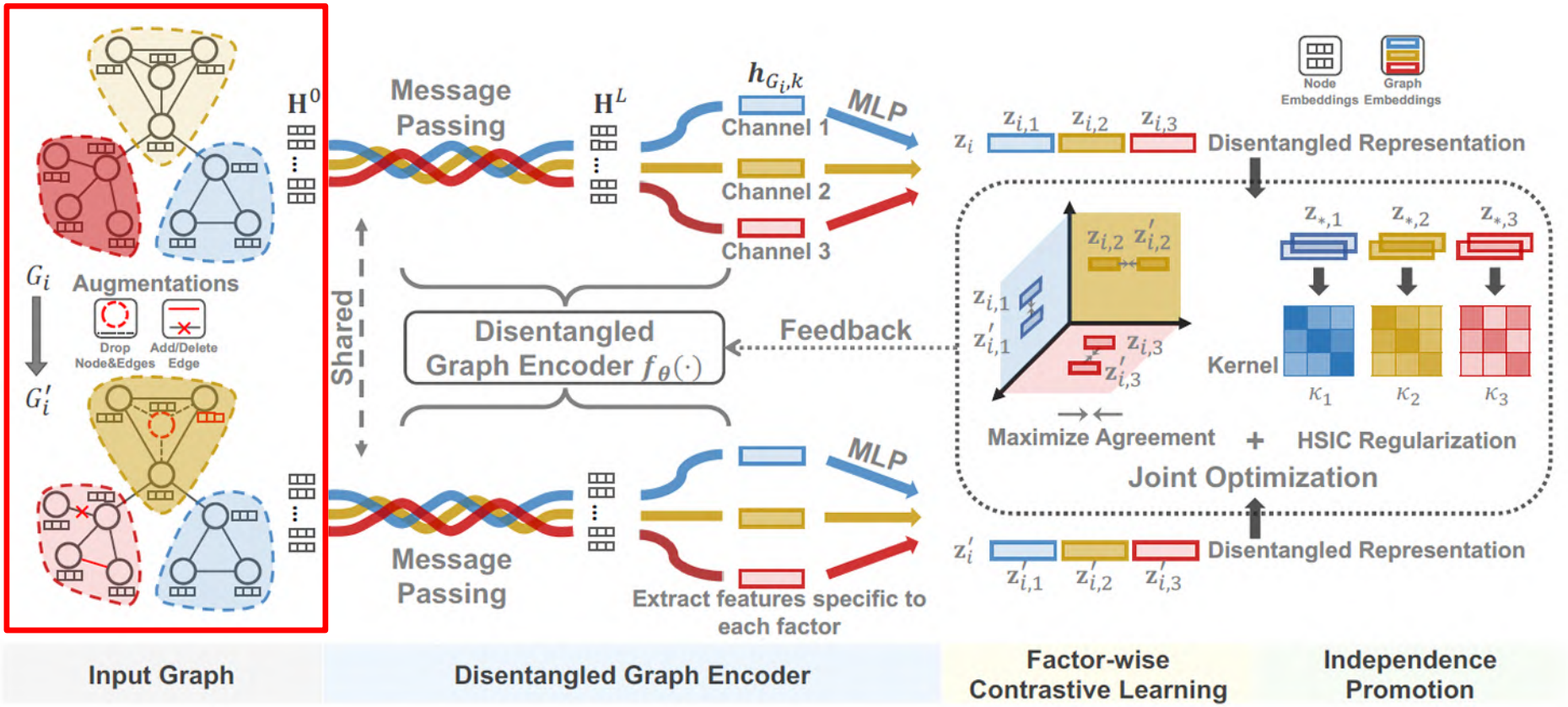
IDGCL: Method

- **Key idea**: disentangled graph encoder + factor-wise contrastive learning + HSIC
  - Each channel for one disentangled factor



Disentangled Graph Contrastive Learning with Independence Promotion. ***TKDE, 2022.***

IDGCL: Method

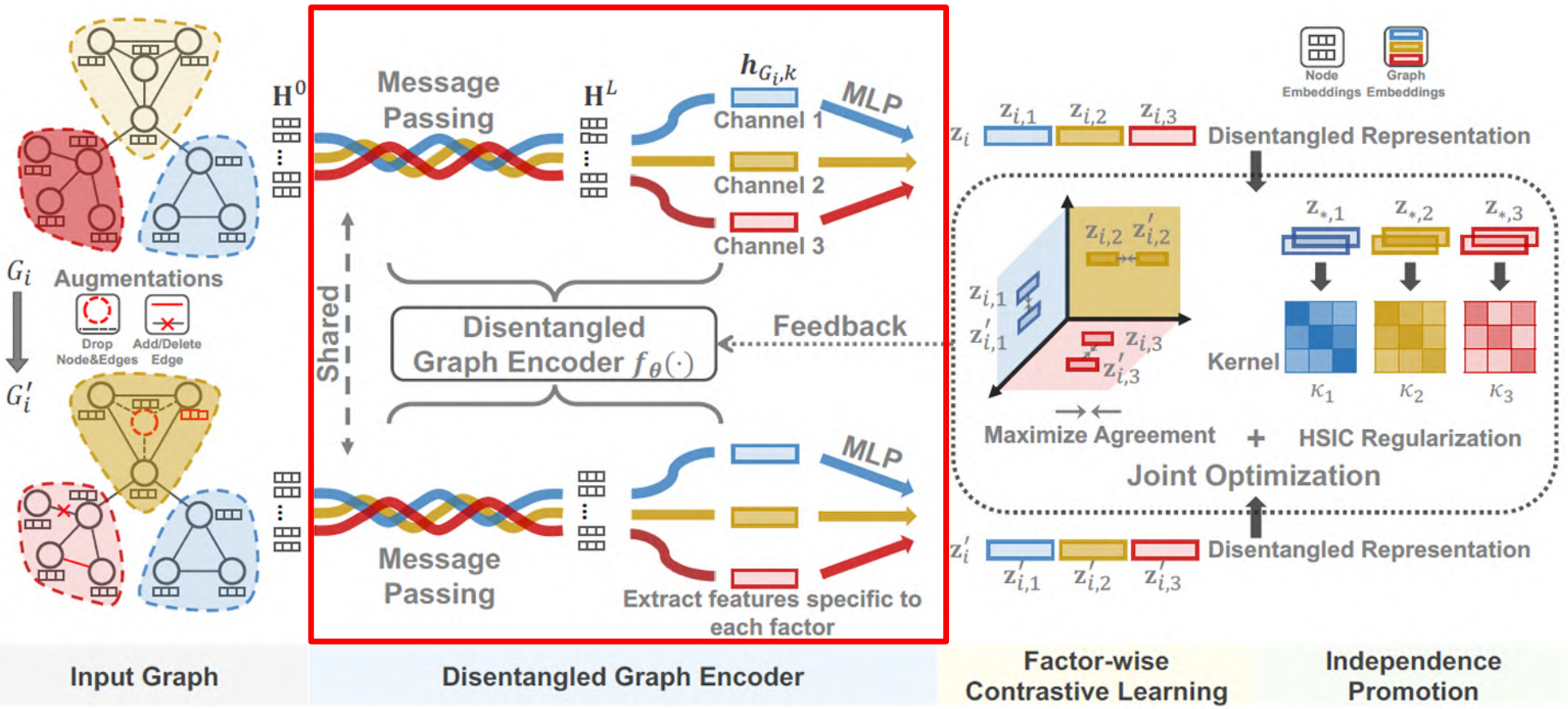


- Graph Augmentation
  - Four types of strategies: node dropping, edge perturbation, attribute masking, subgraph sampling
  - Reflect diverse aspects behind graphs, can be directly extended
  - Self-supervised loss:

$$p_\theta(y_i|x_i) = \frac{\exp \phi(\mathrm{v}_i, \mathrm{v}'_{y_i})}{\sum_{j=1}^{N} \exp \phi(\mathrm{v}_i, \mathrm{v}'_{y_j})}$$

Disentangled Graph Contrastive Learning with Independence Promotion. ***TKDE, 2022.***

IDGCL: Method



- Factor-wise message-passing
  - First, a shared GNN for a few layers
  - Then learn K GNNs with independent parameters
  - Each channel only captures one hidden factor

$$\mathbf{H}_k^{L+1} = \mathrm{GNN}_k(\mathbf{H}^L, A)$$
$$h_{G_i,k} = \mathrm{READOUT}_k(\{\mathbf{H}_k^{L+1}\})$$
$$\mathbf{z}_{i,k} = \mathrm{MLP}_k(h_{G_i,k}).$$

Disentangled Graph Contrastive Learning with Independence Promotion. ***TKDE, 2022.***

IDGCL: Method

- Factor-wise contrastive learning
  - Consider multiple latent factors

$$p_\theta(y_i|G_i) = \mathbb{E}_{p_\theta(k|G_i)}\left[p_\theta(y_i|G_i,k)\right]$$

  - Infer latent factors by K prototypes:

$$p_\theta(k|G_i) = \frac{\exp\phi(\mathbf{z}_{i,k},\mathbf{c}_k)}{\sum_{k=1}^{K}\exp\phi(\mathbf{z}_{i,k},\mathbf{c}_k)}$$

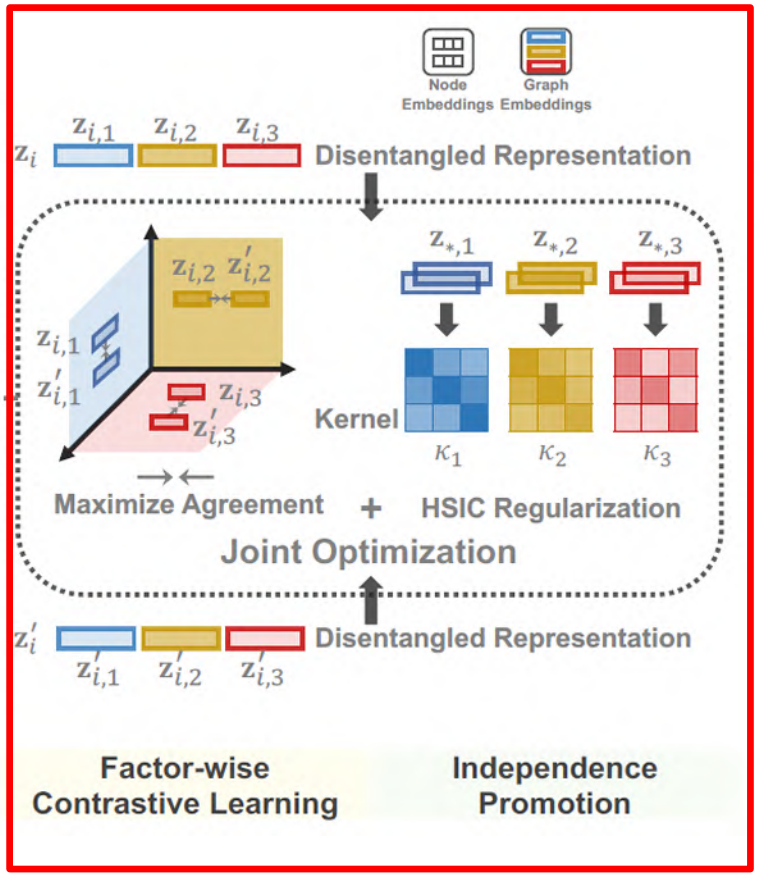
  - Subtask under each latent factor:

$$p_\theta(y_i|G_i,k) = \frac{\exp\phi(\mathbf{z}_{i,k},\mathbf{z}'_{y_i,k})}{\sum_{j=1}^{N}\exp\phi(\mathbf{z}_{i,k},\mathbf{z}'_{y_j,k})}$$

- Statistical Independence regularizer:

$$\mathcal{L}_{reg} = \sum_{1\le k_A<k_B\le K}\mathrm{HSIC}(\mathbf{z}_{*,k_A},\mathbf{z}_{*,k_B})$$

- Overall objective function: $\min_\theta \mathcal{L}(\theta,\mathcal{B}) + \lambda\mathcal{L}_{reg}$



Disentangled Graph Contrastive Learning with Independence Promotion. ***TKDE, 2022.***

IDGCL: Optimization

- Optimization
  - Maximize the joint probability $\prod_{i=1}^{N} p(y_i|G_i)$
  - Step 1: infer the posterior probability of latent factors with Bayes' theorem:

$$\theta^* = \arg\max_{\theta} \sum_{i=1}^{N} \log p_\theta(y_i|G_i) = \arg\max_{\theta} \sum_{i=1}^{N} \log \mathbb{E}_{p_\theta(k|G_i)} \left[p_\theta(y_i|G_i,k)\right].$$

  - Step 2: approximate the posterior probability with a variational distribution:

$$p_\theta(k|G_i,y_i) = \frac{p_\theta(k|G_i)p_\theta(y_i|G_i,k)}{\sum_{k=1}^{K} p_\theta(k|G_i)p_\theta(y_i|G_i,k)} \qquad p_\theta(y_i|G_i,k) = \frac{\exp \phi(\mathbf{z}_{i,k}, \mathbf{z}'_{y_i,k})}{\sum_{j=1}^{N} \exp \phi(\mathbf{z}_{i,k}, \mathbf{z}'_{y_j,k})}$$

  - Step 3: optimize the evidence lower bound (ELBO) of the log-likelihood

$$q_\theta(k|G_i,y_i) = \frac{p_\theta(k|G_i)\hat{p}_\theta(y_i|G_i,k)}{\sum_{k=1}^{K} p_\theta(k|G_i)\hat{p}_\theta(y_i|G_i,k)} \qquad \hat{p}_\theta(y_i|G_i,k) = \frac{\exp \phi(\mathbf{z}_{i,k}, \mathbf{z}'_{i,k})}{\sum_{j\in\mathcal{B},j\neq i}^{|\mathcal{B}|} \exp \phi(\mathbf{z}_{i,k}, \mathbf{z}'_{j,k})}$$

Theorem 1. *The log likelihood function of each graph* $\log p_\theta(y_i|G_i)$ *is lower bounded by the ELBO:* $\mathcal{L}(\theta,i) = \mathbb{E}_{q_\theta(k|G_i,y_i)}[\log p_\theta(y_i|G_i,k)] - D_{KL}(q_\theta(k|G_i,y_i) \parallel p_\theta(k|G_i))$.

  - Step 4: calculate the independence regularizer
  - Step 5: update parameters using gradient descends

Disentangled Graph Contrastive Learning with Independence Promotion. ***TKDE, 2022.***

IDGCL: Experiments

- Classification performance in benchmarks

TABLE 2: Graph classification accuracy (%) of our proposed method and baselines in the unsupervised learning setting. In each column, the boldfaced score denotes the best result of all the methods and the underlined score represents the best result of baselines. "–" indicates the result is not reported in the paper.

	MUTAG	PTC-MR	PROTEINS	NCI1	IMDB-B	IMDB-M	RDT-B	RDT-M5K	COLLAB
SP	85.2±2.4	58.2±2.4	75.1±0.5	73.0±0.2	55.6±0.2	38.0±0.3	64.1±0.1	39.6±0.2	–
GK	81.7±2.1	57.3±1.4	71.7±0.6	62.3±0.3	65.9±1.0	43.9±0.4	77.3±0.2	41.0±0.2	72.8±0.3
WL	80.7±3.0	58.0±0.5	72.9±0.6	80.0±0.5	72.3±3.4	47.0±0.5	68.8±0.4	46.1±0.2	–
DGK	87.4±2.7	60.1±2.6	73.3±0.8	80.3±0.5	67.0±0.6	44.6±0.5	78.0±0.4	41.3±0.2	73.1±0.3
MLG	87.9±1.6	63.3±1.5	76.1±2.0	80.8±1.3	66.6±0.3	41.2±0.0	–	–	–
node2vec	72.6±10.2	58.6±8.0	57.5±3.6	54.9±1.6	–	–	–	–	–
sub2vec	61.1±15.8	60.0±6.4	53.0±5.6	52.8±1.5	55.3±1.5	36.7±0.8	71.5±0.4	36.7±0.4	–
graph2vec	83.2±9.3	60.2±6.9	73.3±2.1	73.2±1.8	71.1±0.5	50.4±0.9	75.8±1.0	47.9±0.3	–
GVAE	87.7±0.7	61.2±1.8	–	–	70.7±0.7	49.3±0.4	87.1±0.1	52.8±0.2	–
InfoGraph	89.0±1.1	61.7±1.4	74.4±0.3	76.2±1.1	73.0±0.9	49.7±0.5	82.5±1.4	53.5±1.0	70.7±1.1
GCC	–	–	–	–	72.0	49.4	89.8	53.7	78.9
MVGRL	89.7±1.1	62.5±1.7	–	–	74.2±0.7	51.2±0.5	84.5±0.6	–	–
GraphCL	86.8±1.3	63.6±1.8	74.4±0.5	77.9±0.4	71.1±0.4	50.7±0.4	89.5±0.8	56.0±0.3	71.4±1.2
JOAO	87.7±0.8	61.1±1.7	74.1±1.1	78.4±0.5	70.8±0.3	51.0±0.5	86.4±1.5	56.0±0.3	69.3±0.3
DGCL	92.1±0.8	65.8±1.5	76.4±0.5	81.9±0.2	75.9±0.7	51.9±0.4	91.8±0.2	56.1±0.2	81.2±0.3
IDGCL	**92.5±0.6**	**66.2±1.3**	**77.1±0.2**	**82.4±0.3**	**76.1±0.2**	**52.3±0.4**	**91.9±0.3**	**56.3±0.2**	**81.3±0.3**

Disentangled Graph Contrastive Learning with Independence Promotion. ***TKDE, 2022.***

IDGCL: Experiments

- OOD graph datasets

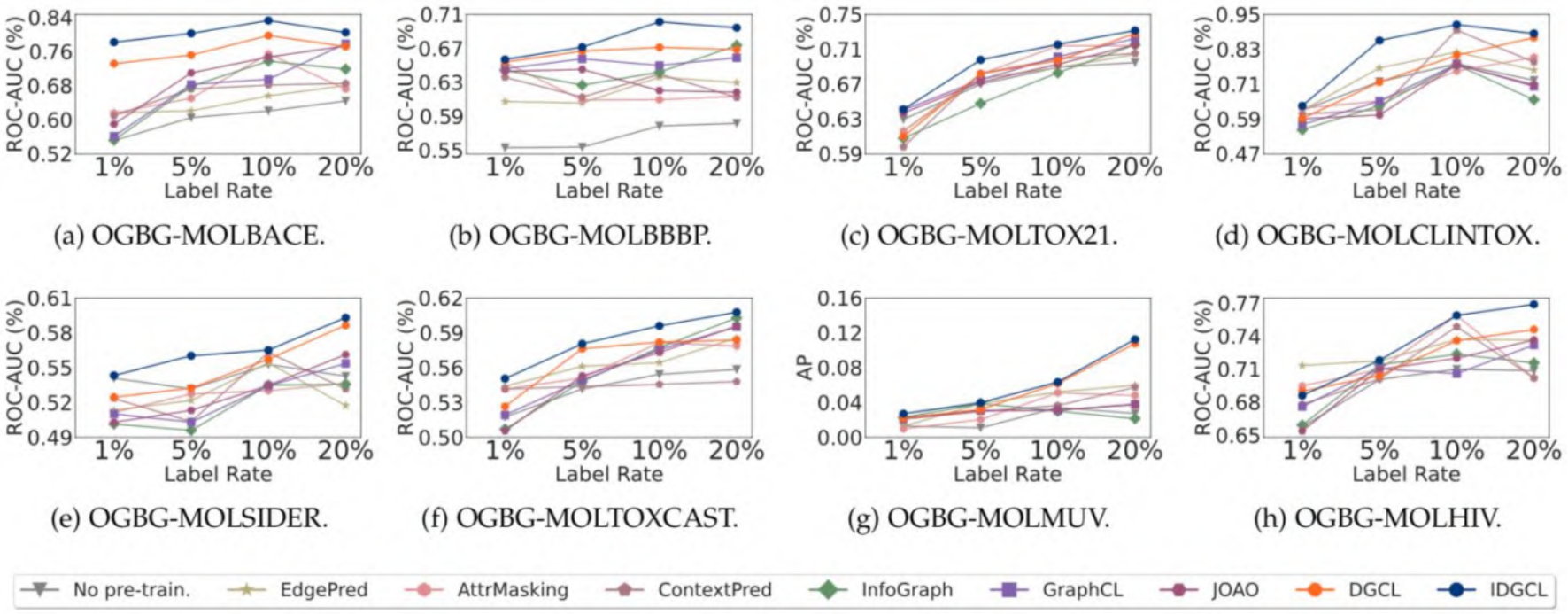
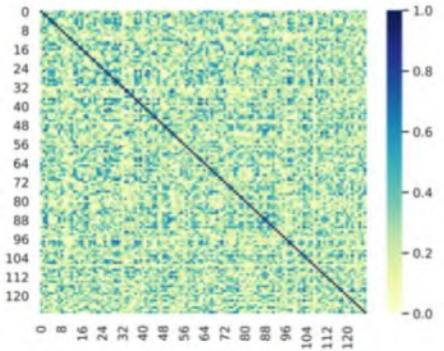


Fig. 5: Graph classification results of our proposed method and baselines in the semi-supervised learning setting.

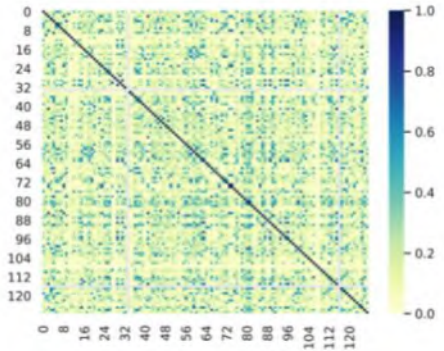
Disentangled Graph Contrastive Learning with Independence Promotion. ***TKDE, 2022.***

IDGCL: Experiments

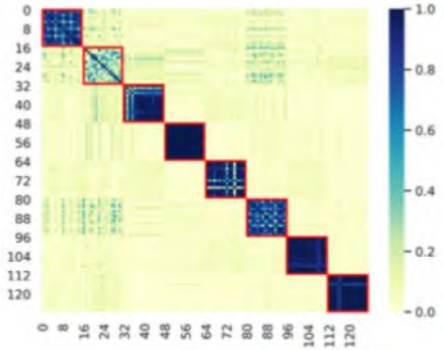
- Visualizations for representations



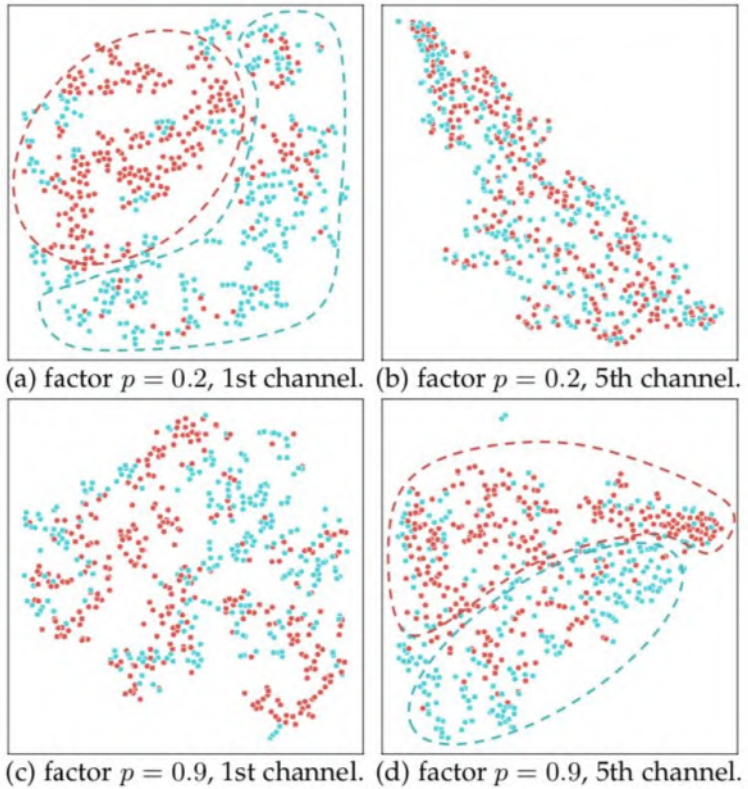
MVGRL



GraphCL



IDGCL



(a) factor $p = 0.2$, 1st channel. (b) factor $p = 0.2$, 5th channel.

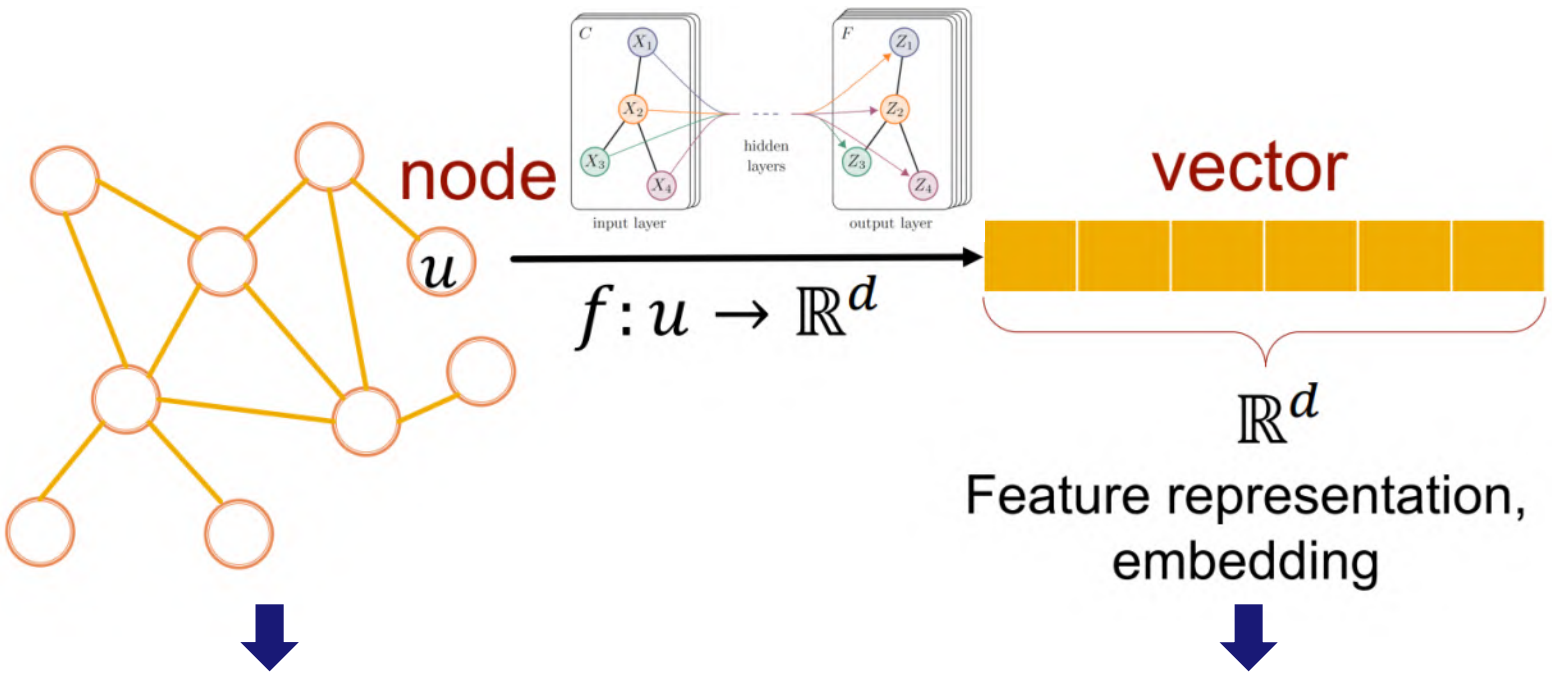
(c) factor $p = 0.9$, 1st channel. (d) factor $p = 0.9$, 5th channel.

Each channel captures one latent factor

Disentangled Graph Contrastive Learning with Independence Promotion. ***TKDE, 2022.***

Handling Distribution Shifts

How to handle distribution shifts in GNN architectures?



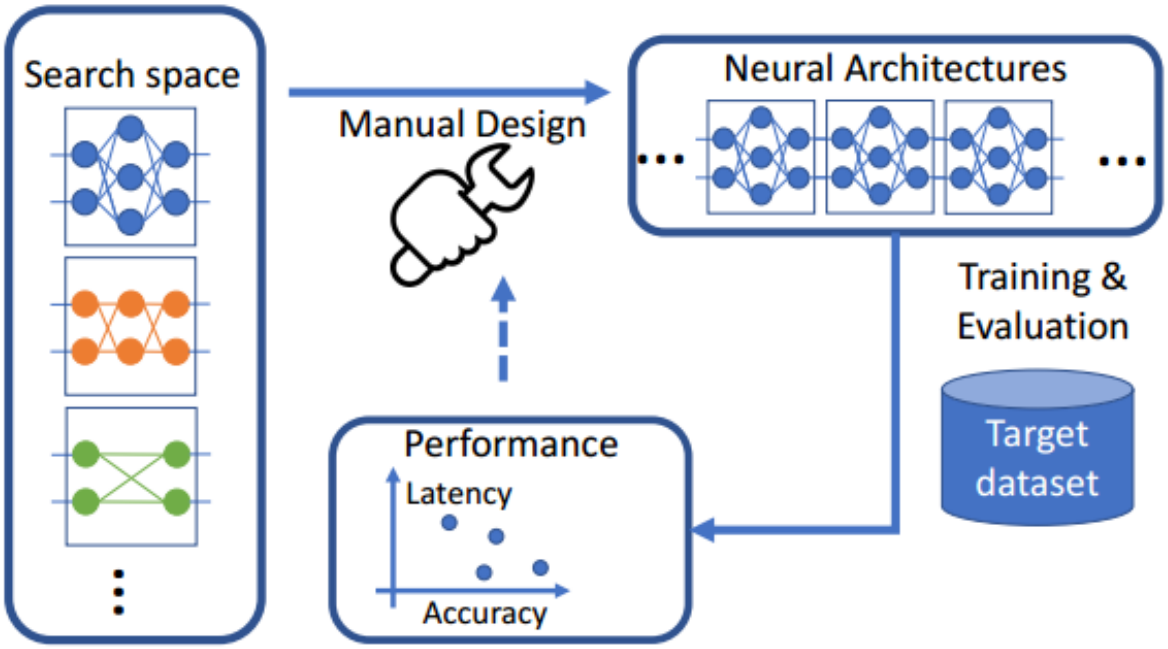
How to handle distribution shifts in the topology space?

How to handle distribution shifts in the vector space?

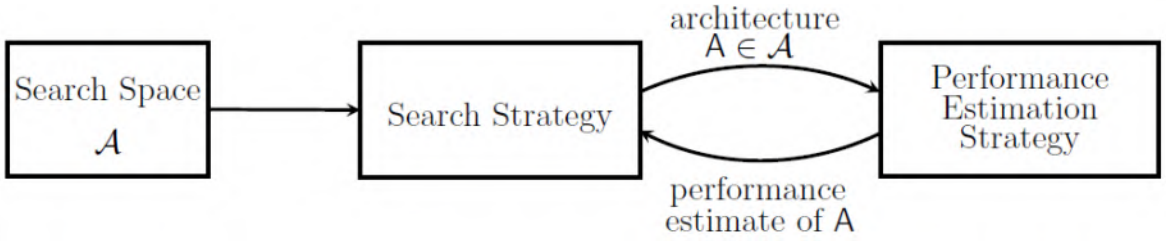
Picture credit: Kipf and Welling (GCN, ICLR 2017), Jure Leskovec (CS224w, Stanford Univ.)

Graph Neural Architecture Search (NAS)

- Goal: automatically learn the best neural architecture

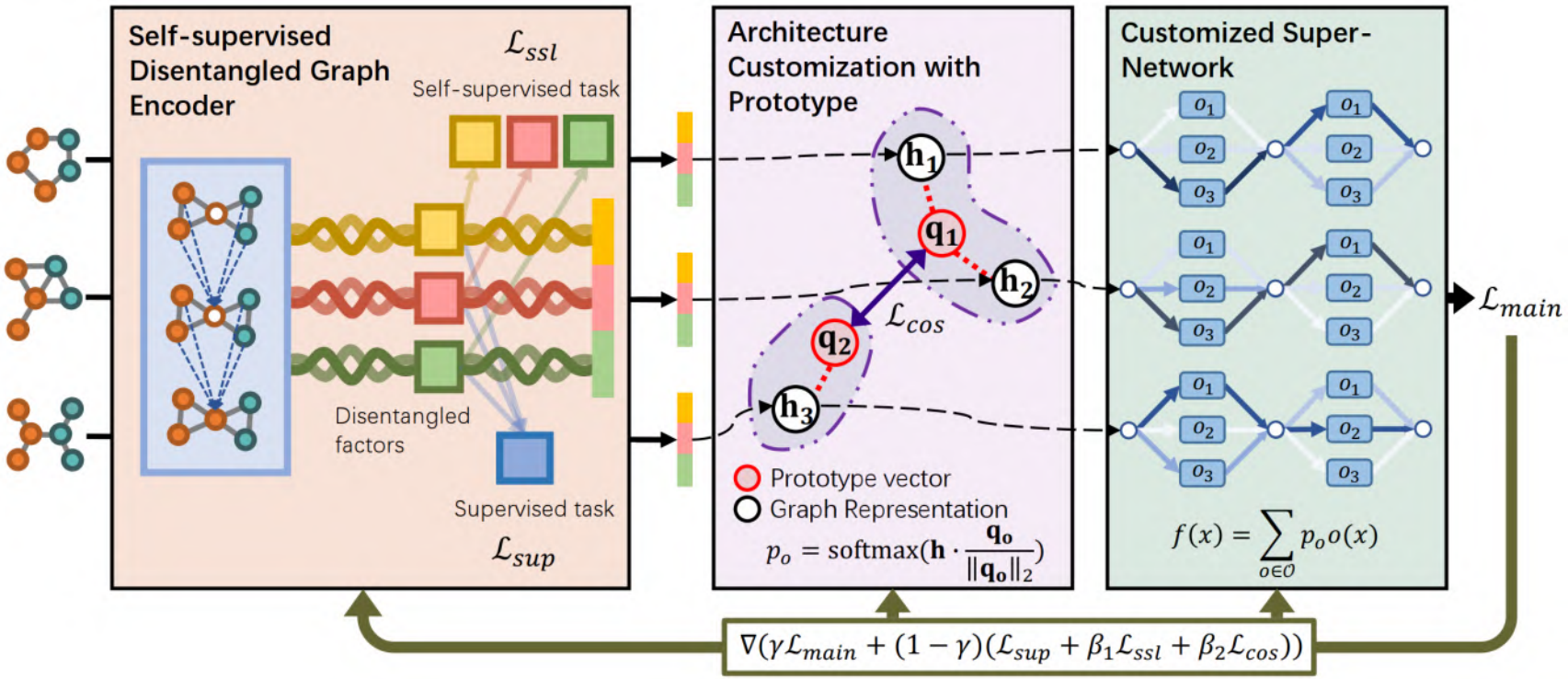


- Key designs



FBNet: Hardware-Aware Efficient ConvNet Design via Differentiable Neural Architecture Search, CVPR 2019
Neural Architecture Search A Survey, JMLR 2019

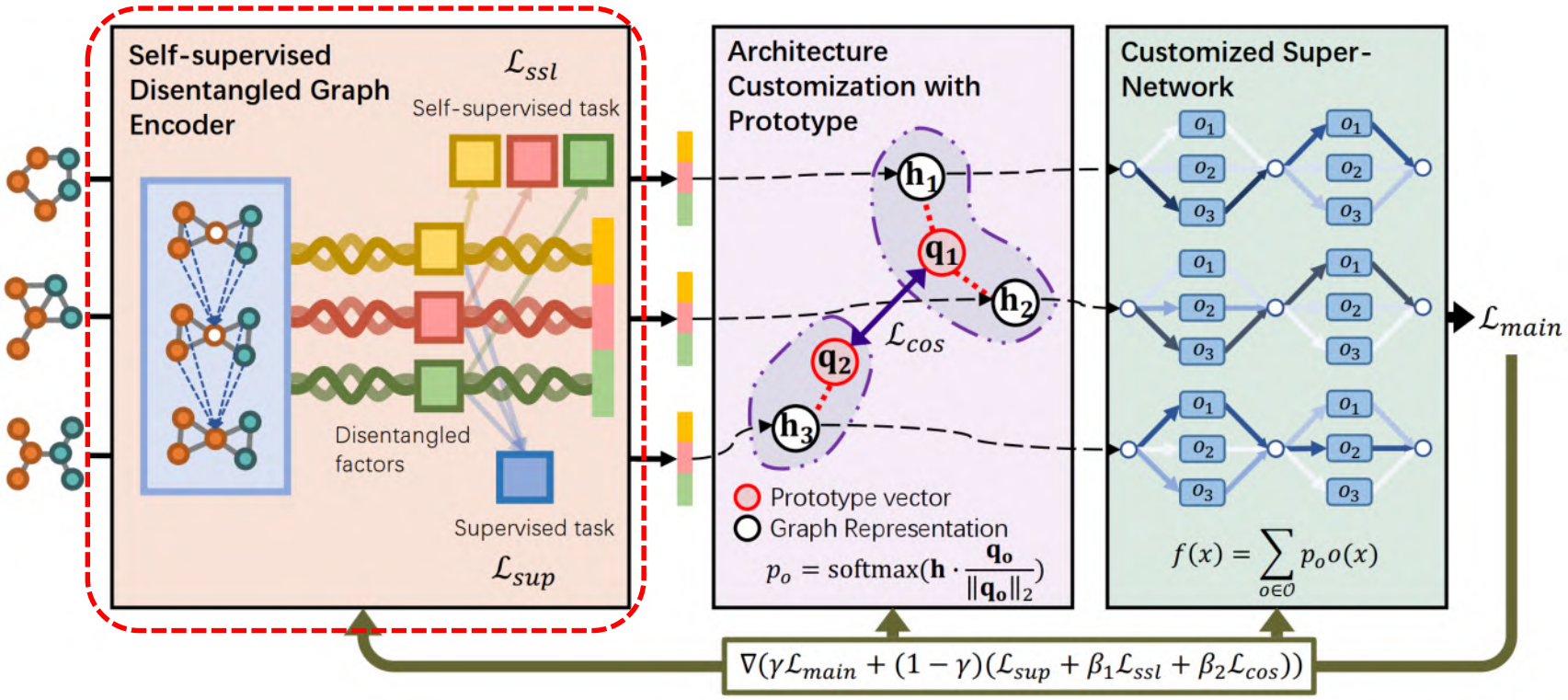
GRACES: Graph Neural Architecture Search under Distribution Shifts



Customize a unique GNN architecture for each graph instance to handle distribution shifts

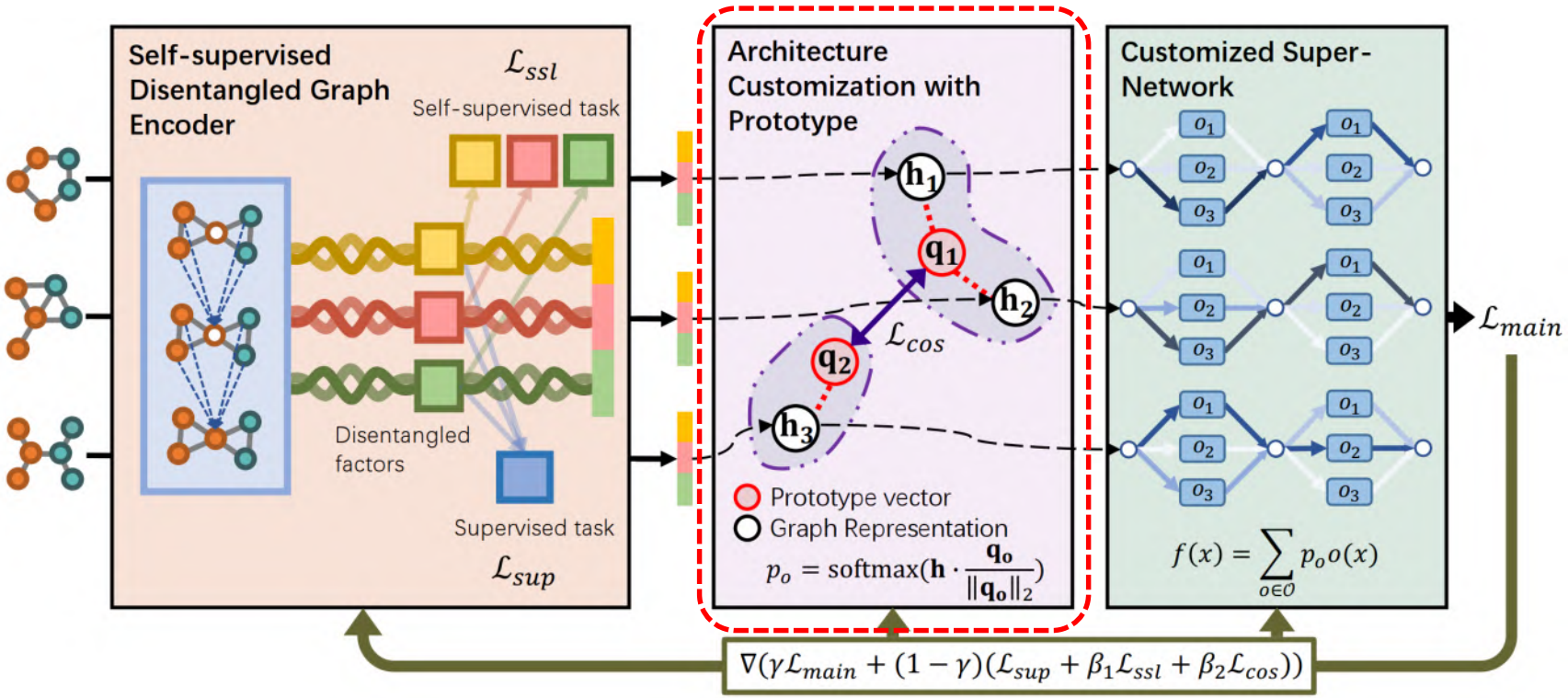
Graph Neural Architecture Search under Distribution Shifts. ***ICML, 2022.***

GRACES: Graph Encoder



- **Goal**: learn a vector representation for each graph to reflect its characteristics
- **Challenge**: preserve diverse properties of the original graph
- **Method**: self-supervised disentangled graph encoder
  - Encoder: disentangled GNN $\mathbf{H}^{(l)} = \Big\|_{k=1}^{K} \text{GNN}(\mathbf{H}_k^{(l-1)}, \mathbf{A})$
  - Supervised loss: the downstream task $\mathcal{L}_{sup} = \sum_{i=1}^{N_{tr}} \ell\left(\mathcal{C}\left(\mathbf{h}_i\right), y_i\right)$
  - Self-supervised loss: node degree as regularization $\mathcal{L}_{ssl} = \sum_{i=1}^{N_{tr}} \sum_{k=1}^{K-1} \ell_{ssl}\left(\hat{y}_{i,k}^{ssl}, y_{i,k}^{ssl}\right)$

GRACES: Architecture Customization



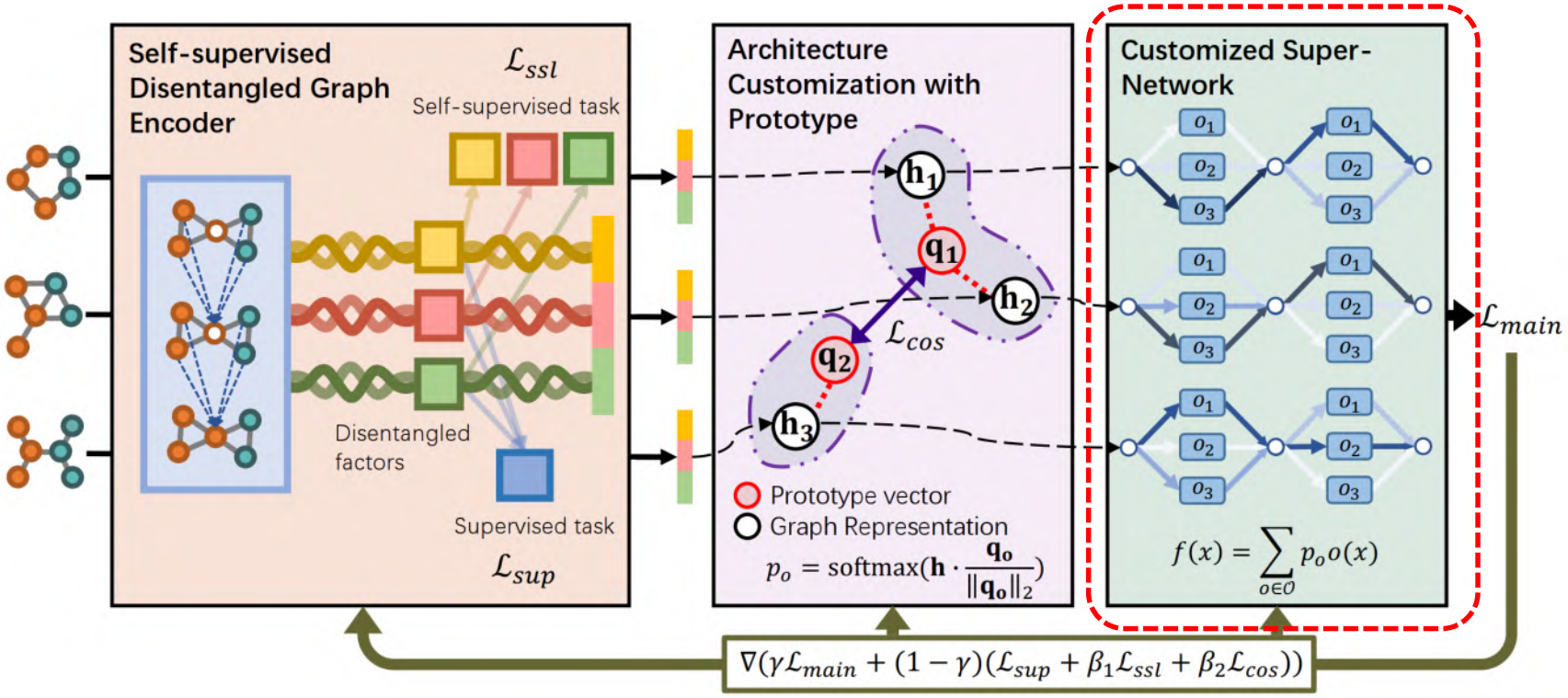
- **Goal**: customize an architecture based on the graph representation
- **Assumption:** graphs with similar characteristics need similar architectures
- **Method**: prototype based architecture customization
  - Probabilities of choosing operations:

$$\hat{p}_o^i = \mathbf{h} \cdot \frac{\mathbf{q}_o^i}{\|\mathbf{q}_o^i\|_2}, p_o^i = \frac{\exp\left(\hat{p}_o^i\right)}{\sum_{o' \in \mathcal{O}} \exp\left(\hat{p}_{o'}^i\right)},$$

  - Regularizer to avoid mode collapse:

$$\mathcal{L}_{cos} = \sum_i \sum_{o,o' \in \mathcal{O}, o \neq o'} \frac{\mathbf{q}_o^i \cdot \mathbf{q}_{o'}^i}{\|\mathbf{q}_o^i\|_2 \left\|.\mathbf{q}_{o'}^i\right\|_2}$$

GRACES: Learning Architecture Parameters



- **Goal**: learn parameters for the customized architectures
- **Method**: customized super-network $f^i(\mathbf{x}) = \sum_{o \in \mathcal{O}} p_o^i o(\mathbf{x})$
- **Loss functions:**

$$\mathcal{L} = \gamma \mathcal{L}_{main} + (1-\gamma)\mathcal{L}_{reg}$$

$$\mathcal{L}_{reg} = \mathcal{L}_{sup} + \beta_1 \mathcal{L}_{ssl} + \beta_2 \mathcal{L}_{cos}$$

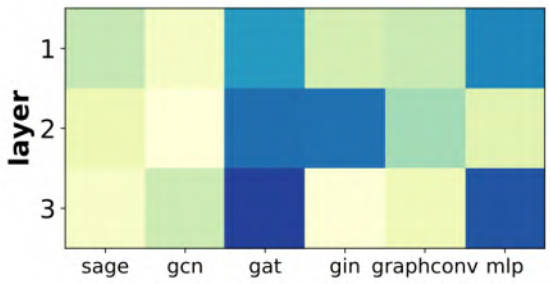
GRACES: Experiments

Synthetic OOD graph datasets

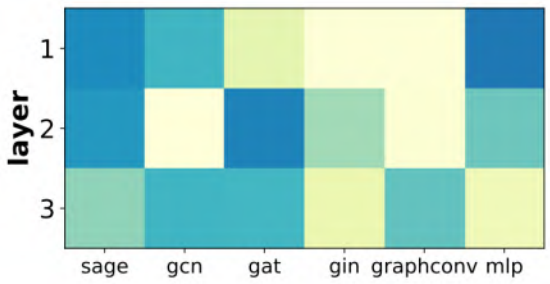
bias	$b = 0.7$	$b = 0.8$	$b = 0.9$
GCN	$48.39_{\pm 1.69}$	$41.55_{\pm 3.88}$	$39.13_{\pm 1.76}$
GAT	$50.75_{\pm 4.89}$	$42.48_{\pm 2.46}$	$40.10_{\pm 5.19}$
GIN	$36.83_{\pm 5.49}$	$34.83_{\pm 3.10}$	$37.45_{\pm 3.59}$
SAGE	$46.66_{\pm 2.51}$	$44.50_{\pm 5.79}$	$44.79_{\pm 4.83}$
GraphConv	$47.29_{\pm 1.95}$	$44.67_{\pm 5.88}$	$44.82_{\pm 4.84}$
MLP	$48.27_{\pm 1.27}$	$46.73_{\pm 3.48}$	$46.41_{\pm 2.34}$
ASAP	$54.07_{\pm 13.85}$	$48.32_{\pm 12.72}$	$43.52_{\pm 8.41}$
DIR	$50.08_{\pm 3.46}$	$48.22_{\pm 6.27}$	$43.11_{\pm 5.43}$
random	$45.92_{\pm 4.29}$	$51.72_{\pm 5.38}$	$45.89_{\pm 5.09}$
DARTS	$50.63_{\pm 8.90}$	$45.41_{\pm 7.71}$	$44.44_{\pm 4.42}$
GNAS	$55.18_{\pm 18.62}$	$51.64_{\pm 19.22}$	$37.56_{\pm 5.43}$
PAS	$52.15_{\pm 4.35}$	$43.12_{\pm 5.95}$	$39.84_{\pm 1.67}$
GRACES	$\mathbf{65.72}_{\pm \mathbf{17.47}}$	$\mathbf{59.57}_{\pm \mathbf{17.37}}$	$\mathbf{50.94}_{\pm \mathbf{8.14}}$

Real-world OOD graph datasets

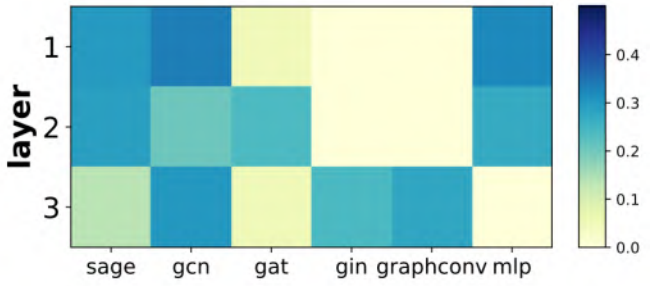
dataset	hiv	sider	bace
GCN	$75.99_{\pm 1.19}$	$59.84_{\pm 1.54}$	$68.93_{\pm 6.95}$
GAT	$76.80_{\pm 0.58}$	$57.40_{\pm 2.01}$	$75.34_{\pm 2.36}$
GIN	$77.07_{\pm 1.49}$	$57.57_{\pm 1.56}$	$73.46_{\pm 5.24}$
SAGE	$75.58_{\pm 1.40}$	$56.36_{\pm 1.32}$	$74.85_{\pm 2.74}$
GraphConv	$74.46_{\pm 0.86}$	$56.09_{\pm 1.06}$	$78.87_{\pm 1.74}$
MLP	$70.88_{\pm 0.83}$	$58.16_{\pm 1.41}$	$71.60_{\pm 2.30}$
ASAP	$73.81_{\pm 1.17}$	$55.77_{\pm 1.18}$	$71.55_{\pm 2.74}$
DIR	$77.05_{\pm 0.57}$	$57.34_{\pm 0.36}$	$76.03_{\pm 2.20}$
DARTS	$74.04_{\pm 1.75}$	$60.64_{\pm 1.37}$	$76.71_{\pm 1.83}$
PAS	$71.19_{\pm 2.28}$	$59.31_{\pm 1.48}$	$76.59_{\pm 1.87}$
GRACES	$\mathbf{77.31}_{\pm \mathbf{1.00}}$	$\mathbf{61.85}_{\pm \mathbf{2.56}}$	$\mathbf{79.46}_{\pm \mathbf{3.04}}$



(a) *Tree*-based graphs



(b) *Ladder*-based graphs

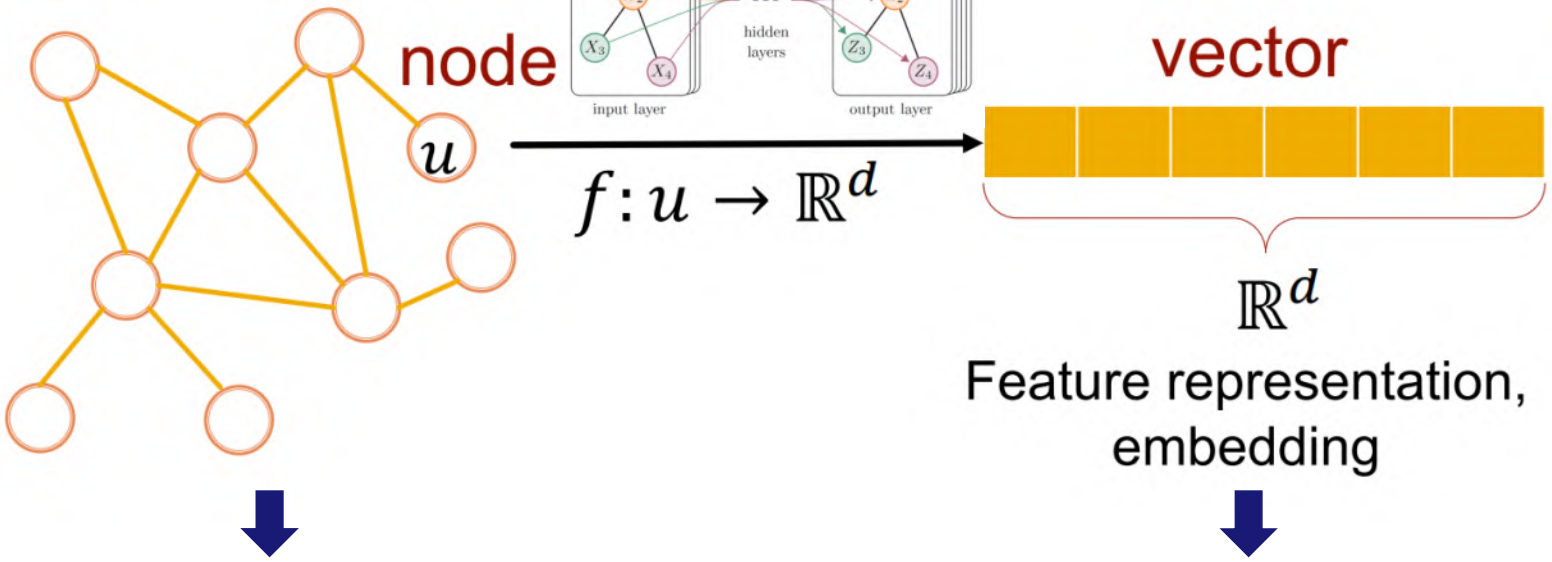


(c) *Wheel*-based graphs

Customization of architectures

Handling Distribution Shifts

How to handle distribution shifts in GNN architectures?



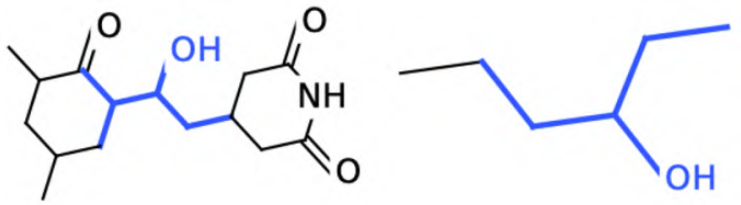
How to handle distribution shifts in the topology space?

How to handle distribution shifts in the vector space?

Picture credit: Kipf and Welling (GCN, ICLR 2017), Jure Leskovec (CS224w, Stanford Univ.)

Graph Invariant Learning (GIL)

- How to get rid of spurious correlations in the topology space?
- Main idea: distinguish invariant and variant subgraphs
  - Invariant: relationships with labels are stable under distribution shifts
  - Variant: the complement of invariant, e.g., environments



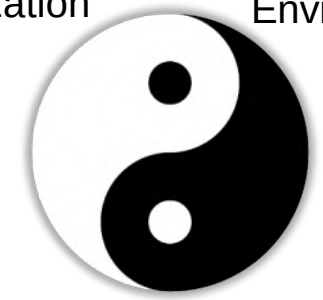
Invariance

OOD Generalization

Variance

Environment/Domain

Information

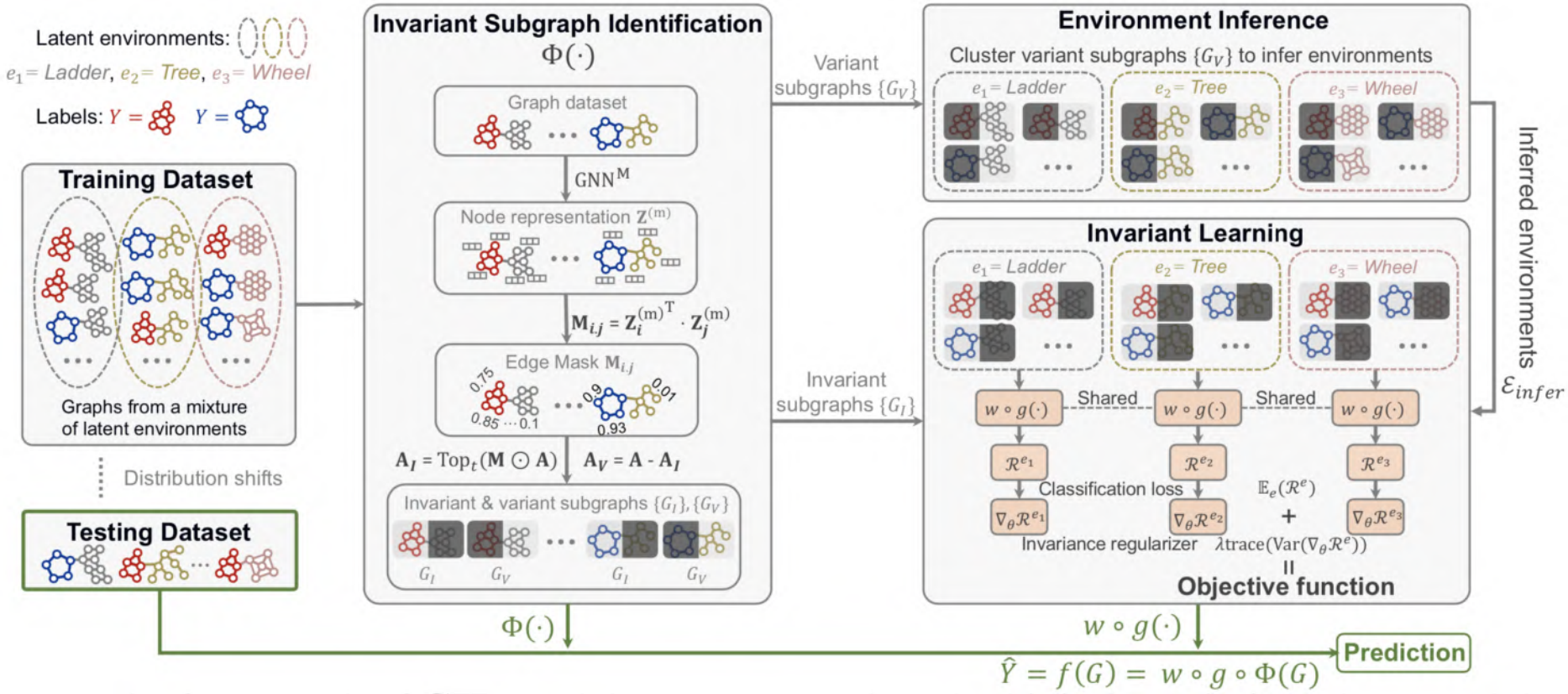


- Challenge:
  - There is no labels for invariant and variant subgraphs
  - Variant and invariant subgraphs are highly entangled

Learning Invariant Graph Representations under Distribution Shifts. ***NeurIPS, 2022.***

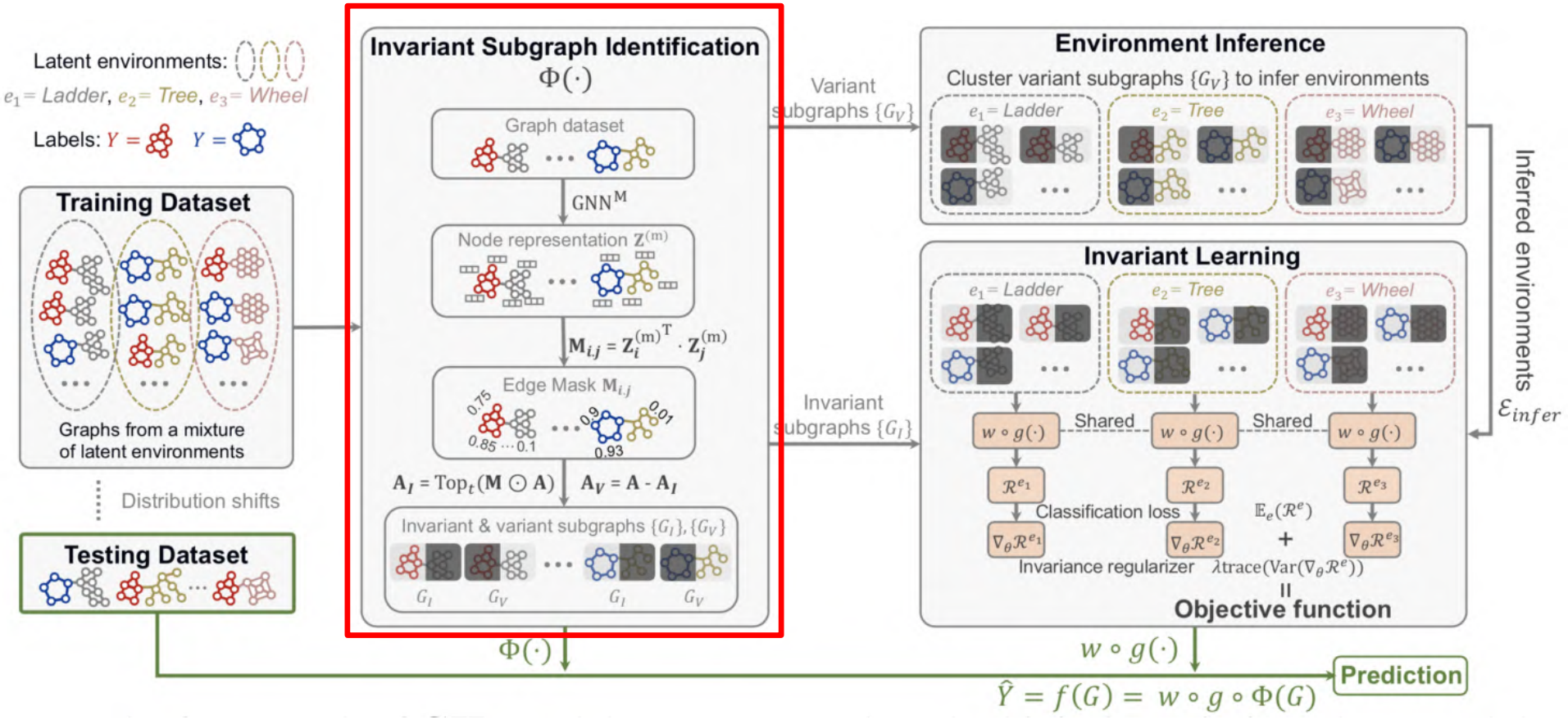
GIL: Method

- **Key Idea**: mutual promotion of invariant learning and environment (variant) inference
  - Invariant subgraphs: for predicting labels
  - Variant subgraphs: for providing environments



Learning Invariant Graph Representations under Distribution Shifts. ***NeurIPS, 2022.***

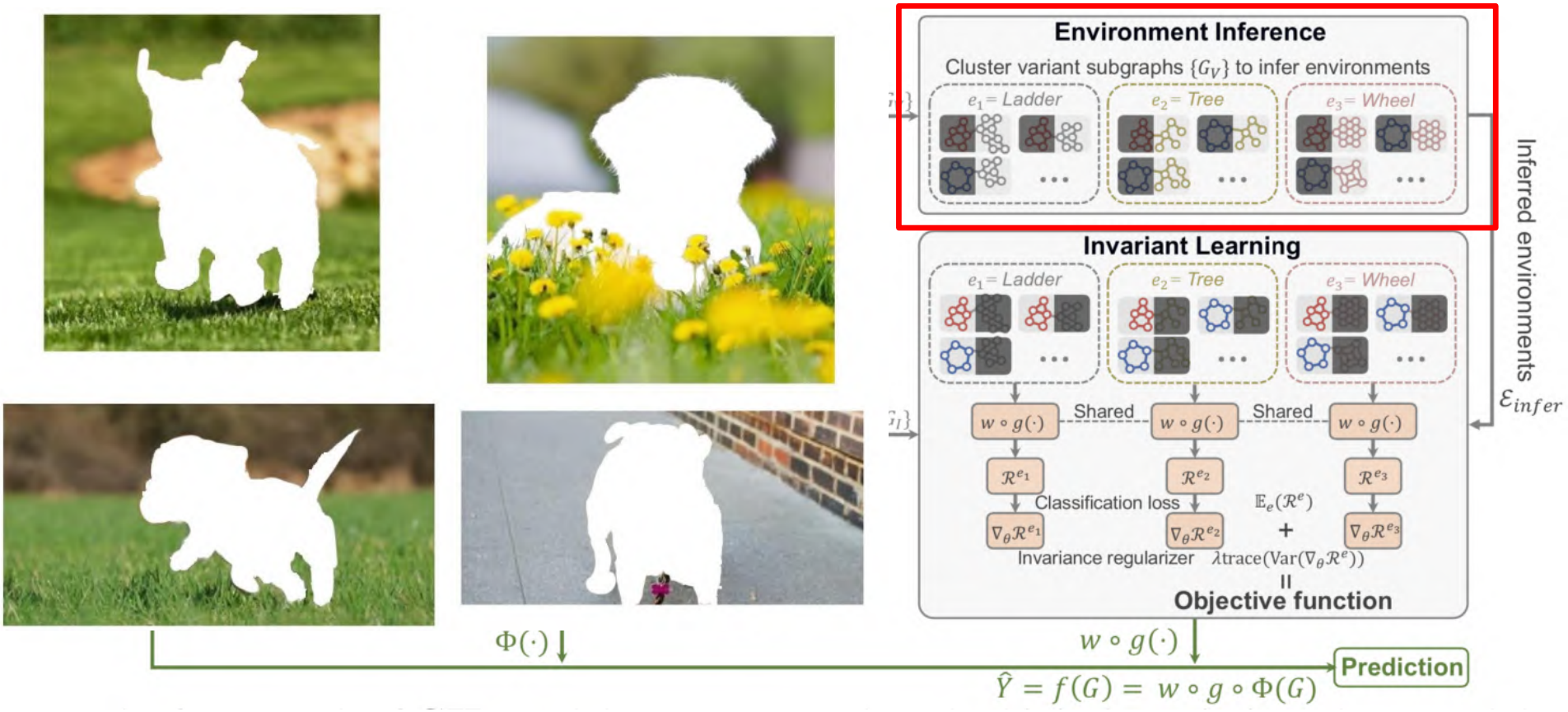
GIL: Method



- **Goal**: learn a mask to separate invariant and variant subgraphs
- **Challenge**: need to handle graphs of various sizes and be inductive
- Proposed method: GNN with top-t pooling

$$\mathbf{Z}^{(m)} = \text{GNN}^{\text{M}}(G) \quad \mathbf{M}_{i,j} = \mathbf{Z}_i^{(m)^\top} \cdot \mathbf{Z}_j^{(m)} \quad \mathbf{A}_I = \text{Top}_t(\mathbf{M} \odot \mathbf{A}), \mathbf{A}_V = \mathbf{A} - \mathbf{A}_I$$

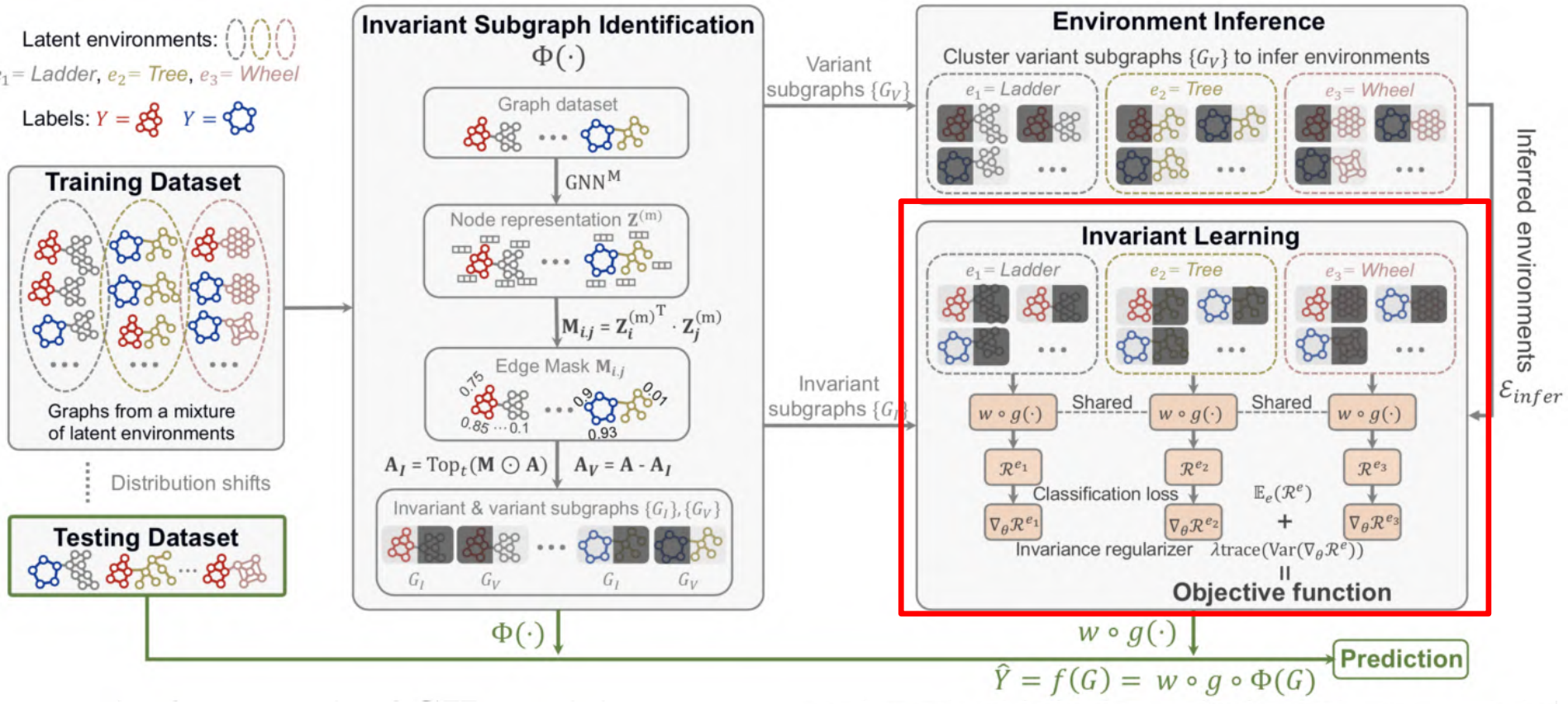
GIL: Method



- Assumption: the variant subgraphs capture environment-discriminative features
- Challenge: there is no ground-truth environment labels
- Proposed method: cluster variant subgraphs infer environments, e.g., k-means

$$\mathcal{E}_{infer} = \text{k-means}(\mathbf{H})$$

GIL: Method



- Goal: find an invariant subgraph generator $\mathcal{I}_{\mathcal{E}} = \{\Phi(\cdot) : P^e(\mathrm{Y}|\Phi(\mathrm{G})) = P^{e'}(\mathrm{Y}|\Phi(\mathrm{G})), e, e' \in \text{supp}(\mathcal{E})\}$
- Optimization:

Theorem 3.2. *A generator $\Phi(\mathrm{G})$ is the optimal generator if and only*

$$\Phi^* = \arg\max_{\Phi \in \mathcal{I}_{\mathcal{E}}} I(\mathrm{Y}; \Phi(\mathrm{G})),$$

where $I(\cdot;\cdot)$ is the mutual information between the label and the generated subgraph.

- Invariance regularizer: $\mathbb{E}_{e \in \text{supp}(\mathcal{E}_{infer})} \mathcal{R}^e(f(\mathrm{G}), \mathrm{Y}; \theta) + \lambda \text{trace}(\text{Var}_{\mathcal{E}_{infer}}(\nabla_\theta \mathcal{R}^e))$

GIL: Theory

- We prove that the maximal invariant subgraph generator can achieve OOD optimal

Theorem 4.1. *Let Φ^* be the optimal invariant subgraph generator in Assumption 3.1 and denote the complement as $\mathrm{G}\backslash\Phi^*(\mathrm{G})$, i.e., the corresponding variant subgraph. Then, we can obtain the optimal predictor under distribution shifts, i.e., the solution to Problem 1, as follows:*

$$\arg\min_{w,g} w \circ g \circ \Phi^*(\mathrm{G}) = \arg\min_{f} \sup_{e\in\mathrm{supp}(\mathcal{E})} \mathcal{R}(f|e), \tag{10}$$

  - Several assumptions:

$$(1)\ \Phi^*(\mathrm{G}) \perp \mathrm{G}\backslash\Phi^*(\mathrm{G})$$

$$(2)\ \forall\Phi \in \mathcal{I}_{\mathcal{E}},\ \exists\ e' \in \mathrm{supp}(\mathcal{E}) \text{ such that } P^{e'}(\Phi(\mathrm{G})) = P^{e}(\Phi(\mathrm{G}))$$
$$\text{and } P^{e'}(\mathrm{G},\mathrm{Y}) = P^{e'}(\Phi(\mathrm{G}),\mathrm{Y})P^{e'}(\mathrm{G}\backslash\Phi(\mathrm{G}))$$

- We prove that GIL maintains permutation invariance.

Theorem 4.2. *Our proposed **GIL** model is permutation-invariant if $\mathrm{GNN}^{\mathbf{M}}$ and $\mathrm{GNN}^{\mathbf{I}}$ are permutation-equivariant and $\mathrm{READOUT}^{\mathbf{I}}$ is permutation-invariant.*

- We show that the time complexity of GIL is on par with the existing GNNs
  - Time Complexity**:** $O(|E|d + |V|d^2)$, $|E|$ and $|V|$ are the edge and node number.

Learning Invariant Graph Representations under Distribution Shifts. ***NeurIPS, 2022.***

GIL: Experiments

- OOD Generalization on synthetic datasets (Spurious-Motif)
  - Each graph includes one invariant motif (i.e., label) and variant motif (i.e., spurious part).
  - r controls the bias strength; $|r_{test} - r_{train}|$ is the distribution shift strength.

Unbiased Test Set

	Scenario 1: $r_{test} = 1/3$					
r_{train}	$r = 1/3$	$r = 0.5$	$r = 0.6$	$r = 0.7$	$r = 0.8$	$r = 0.9$
ERM	53.60±3.79	51.24±4.13	47.04±7.01	38.80±3.72	37.84±3.01	37.44±2.15
Attention	54.31±3.98	53.24±3.56	42.52±6.20	35.20±1.05	34.48±1.18	33.88±1.01
Top-k Pool	54.68±2.71	53.12±5.58	44.56±4.57	37.44±2.04	35.24±2.28	34.28±4.11
SAG Pool	54.08±3.66	52.60±3.52	44.68±5.25	37.68±4.03	34.28±1.82	32.72±1.83
ASAP	54.00±4.21	51.92±3.81	45.12±1.98	36.28±0.86	34.24±2.02	34.40±3.15
GroupDRO	53.20±4.91	51.40±4.35	48.32±5.35	39.12±4.27	38.40±2.76	37.64±1.69
IRM	52.00±2.34	50.60±3.54	47.84±6.95	38.80±3.72	39.84±3.21	39.00±3.98
V-REx	53.16±3.25	46.04±6.11	45.36±3.66	40.24±3.86	39.48±3.00	39.12±3.48
DIR	52.96±5.06	52.08±1.93	50.12±2.76	49.84±2.46	45.20±1.11	41.24±4.73
GIL	**55.44**±3.11	**54.56**±3.02	**53.60**±4.82	**53.12**±2.18	**51.24**±3.88	**46.04**±3.51

Biased Test Set

	Scenario 2: $r_{test} = 0.2$					
r_{train}	$r = 1/3$	$r = 0.5$	$r = 0.6$	$r = 0.7$	$r = 0.8$	$r = 0.9$
ERM	48.48±4.53	41.72±4.81	36.92±6.93	35.72±8.33	28.80±3.91	19.60±1.66
Attention	44.04±4.33	31.64±0.67	25.72±5.34	24.80±4.06	23.20±3.60	18.04±2.88
Top-k Pool	45.68±5.16	34.20±4.34	31.00±2.89	30.64±3.59	29.16±2.18	27.56±3.91
SAG Pool	44.36±6.09	38.64±3.02	31.36±4.40	32.84±1.86	28.72±3.11	26.60±5.37
ASAP	49.88±4.90	34.52±4.35	27.00±2.61	27.20±2.53	27.96±3.89	22.88±4.33
GroupDRO	52.68±4.04	43.68±4.05	31.92±6.84	34.36±8.41	28.88±5.14	20.32±1.64
IRM	50.24±6.73	41.60±4.75	35.24±5.35	34.92±8.03	29.44±5.47	21.84±3.57
V-REx	50.56±2.83	37.16±6.24	34.52±3.00	29.72±4.58	27.32±3.18	24.04±6.08
DIR	50.68±5.20	49.96±1.75	45.44±6.00	40.56±2.36	39.92±4.53	32.52±4.59
GIL	**54.80**±3.93	**52.48**±4.41	**50.08**±5.47	**47.44**±2.87	**46.36**±3.80	**35.80**±5.03

Learning Invariant Graph Representations under Distribution Shifts. ***NeurIPS, 2022.***

GIL: Experiments

- OOD Generalization on real-world datasets

	MNIST-75sp	Graph-SST2	MOLSIDER	MOLHIV
ERM	14.94±3.27	81.44±0.59	57.57±1.56	76.20±1.14
Attention	16.44±3.78	81.57±0.71	56.99±0.54	75.84±1.33
Top-k Pool	15.02±3.08	79.78±1.35	60.63±1.52	73.01±1.65
SAG Pool	19.34±1.73	80.24±1.72	61.29±1.31	73.26±0.84
ASAP	15.14±3.58	81.57±0.84	55.77±1.34	73.81±1.17
GroupDRO	15.72±4.35	81.29±1.44	56.31±1.15	75.44±2.70
IRM	18.74±2.43	81.01±1.13	57.10±0.92	74.46±2.74
V-REx	18.40±1.12	81.76±0.08	57.76±0.78	75.62±0.79
DIR	17.38±3.52	83.29±0.53	57.74±1.63	77.05±0.57
GSAT	20.12±1.35	82.95±0.58	60.82±1.36	76.47±1.53
GIL	**21.94±0.38**	**83.44±0.37**	**63.50±0.57**	**79.08±0.54**

- ogbg-molhiv

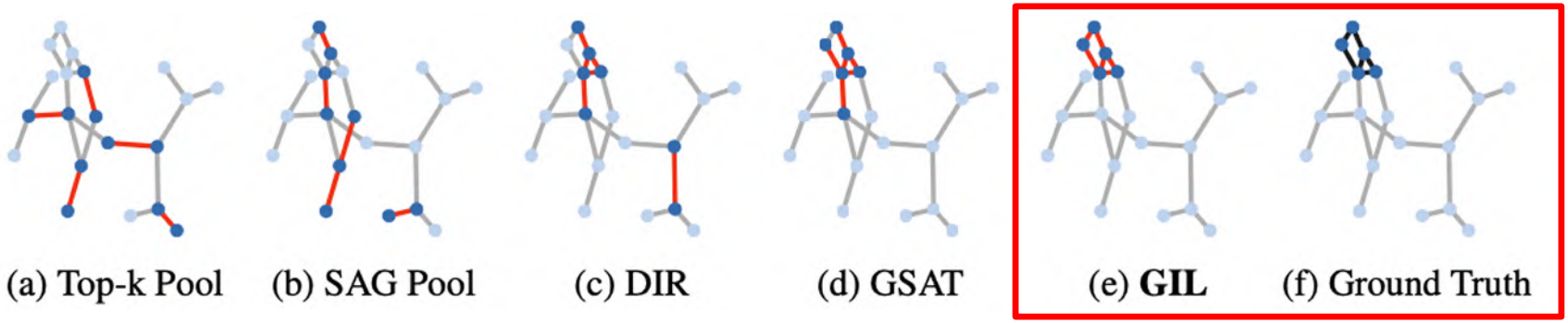
CIN (Rank #8)	**GIL (CIN Backbone)**	HIG (Rank #2)	PAS+FPs (Rank #1)	**GIL (HIG Backbone)**
80.94±0.57	**81.15±0.46**	84.03±0.21	84.20±0.15	**84.23±0.25**

Compatible with various backbone GNNs
and a new SOTA on OGB leaderboard!

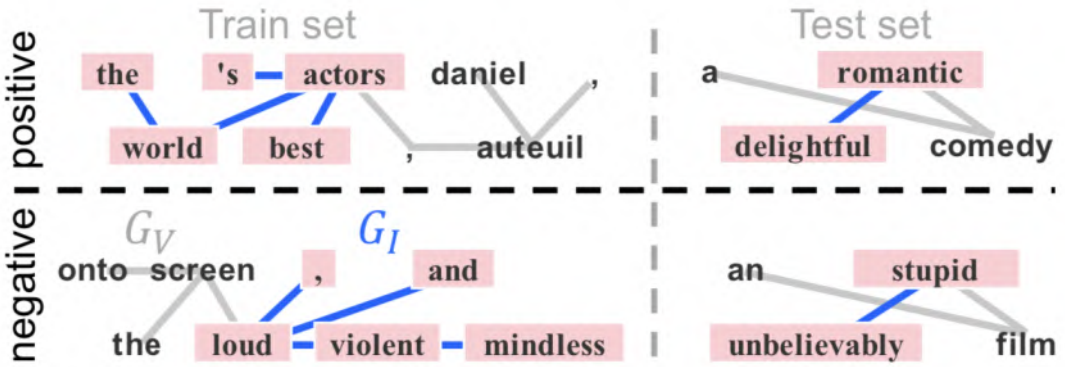
Learning Invariant Graph Representations under Distribution Shifts. ***NeurIPS, 2022.***

GIL: Experiments

- Showcase on Spurious-Motif datasets



- Showcases on Graph-SST2 (human-understandable)

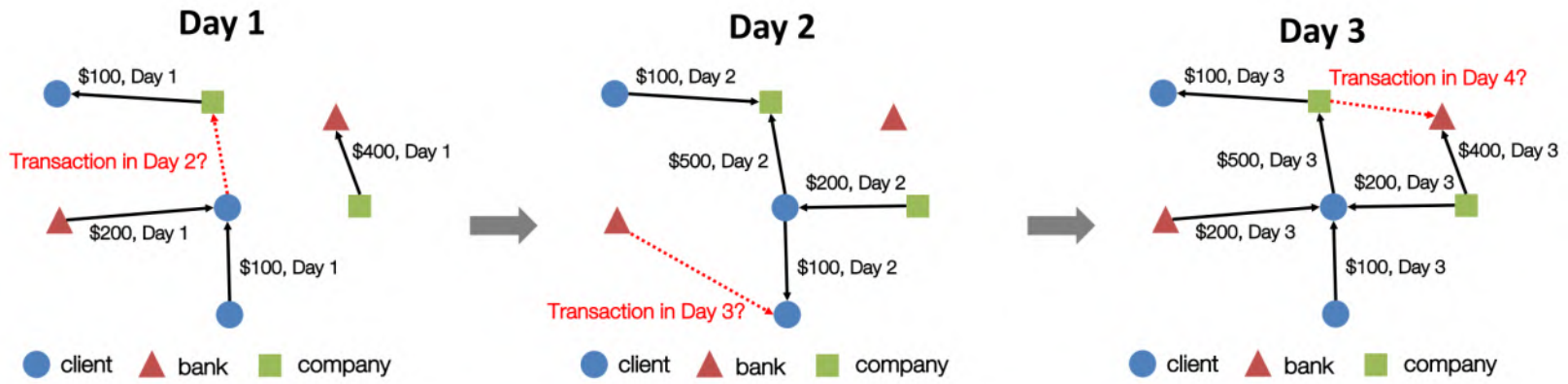


Capture the subgraphs with positive/negative semantics

Learning Invariant Graph Representations under Distribution Shifts. ***NeurIPS, 2022.***

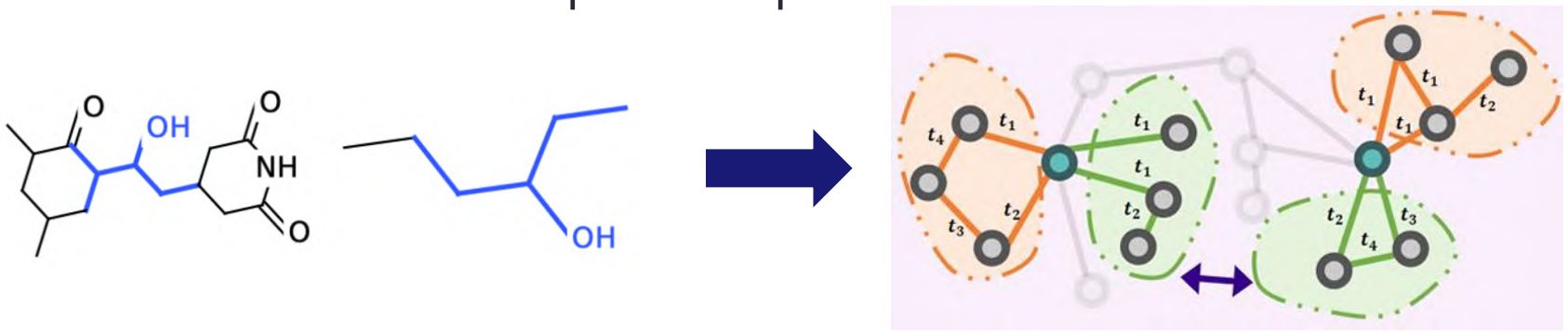
Disentangled Intervention based Dynamic Graph Attention Network (DIDA)

- Many graphs are dynamic in nature



Picture credit: ROLAND: graph learning framework for dynamic graphs, KDD 2022

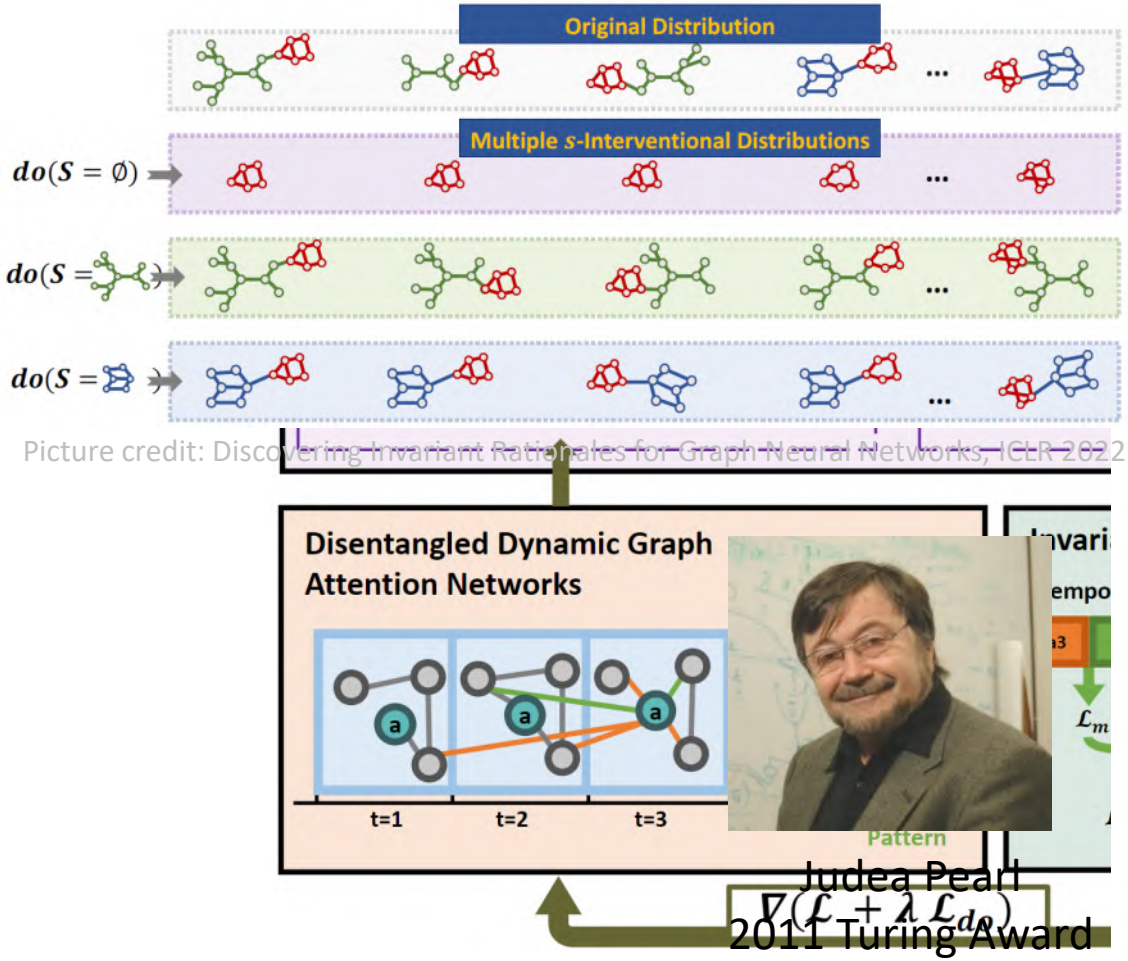
- Distribution shifts can be spatio-temporal



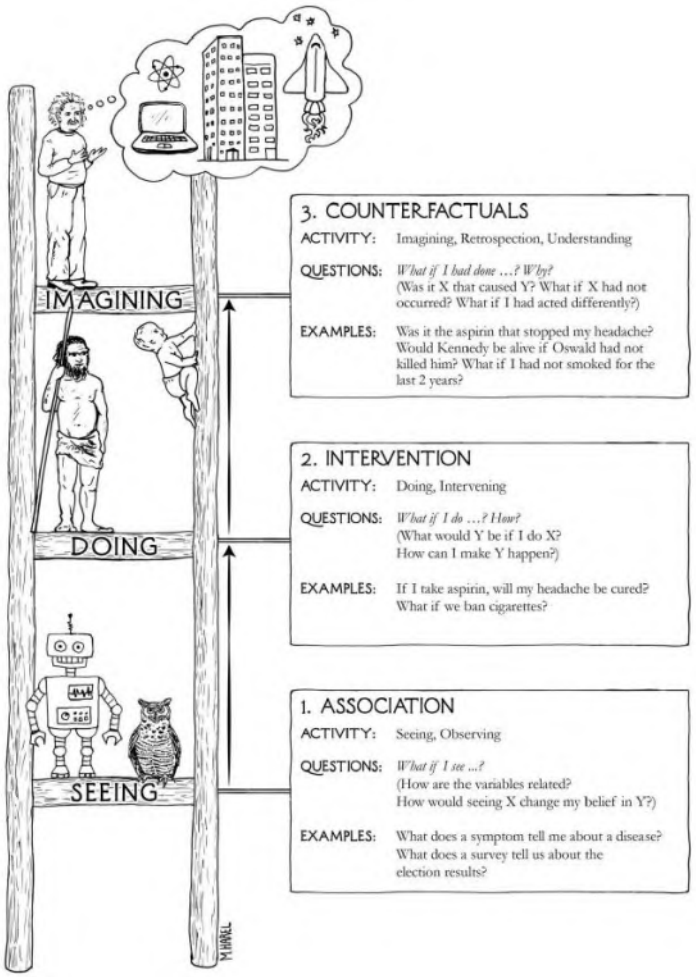
Dynamic Graph Neural Networks Under Spatio-Temporal Distribution Shift. ***NeurIPS, 2022.***

DIDA: Method

- **Key Idea**: finding invariant/variant spatial-temporal patterns and apply intervention
  - Intervention: from causal theory to get rid of spurious correlation



Picture credit: Discovering Invariant Rationales for Graph Neural Networks, ICLR 2022



Judea Pearl
2011 Turing Award

Dynamic Graph Neural Networks Under Spatio-Temporal Distribution Shift. ***NeurIPS, 2022.***

DIDA: Method

- Goal: separate invariant and variant spatial-temporal subgraphs
- Proposed method: disentangled dynamic graph attention network
  - First calculate masks

$$\mathbf{q}_u^t = \mathbf{W}_q(\mathbf{h}_u^t || \mathrm{TE}(t)), \mathbf{k}_v^{t'} = \mathbf{W}_k(\mathbf{h}_v^{t'} || \mathrm{TE}(t')), \mathbf{v}_v^{t'} = \mathbf{W}_v(\mathbf{h}_v^{t'} || \mathrm{TE}(t'))$$

$$\mathbf{m}_I = \mathrm{Softmax}(\frac{\mathbf{q} \cdot \mathbf{k}^I}{\sqrt{d}}), \mathbf{m}_V = \mathrm{Softmax}(-\frac{\mathbf{q} \cdot \mathbf{k}^I}{\sqrt{d}})$$

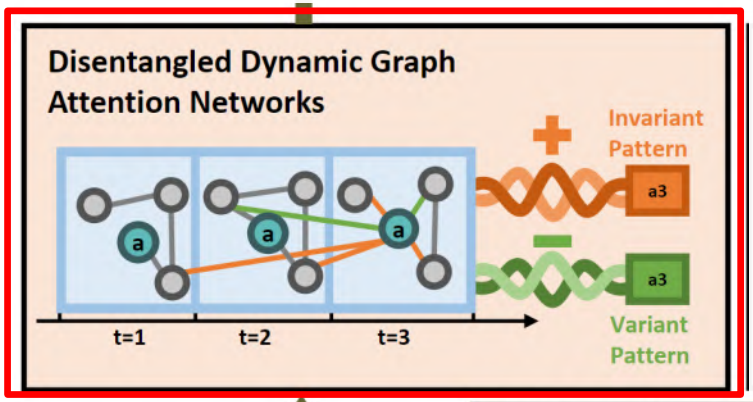
  - Then calculate message-passing

$$\mathbf{z}_I^t(u) = \mathrm{Agg}_I(\mathbf{m}_I, \mathbf{v} \odot \mathbf{m}_f)$$

$$\mathbf{z}_V^t(u) = \mathrm{Agg}_V(\mathbf{m}_V, \mathbf{v})$$

  - Updating node representation

$$\mathbf{h}_u^t \leftarrow \mathbf{z}_I^t(u) + \mathbf{z}_V^t(u)$$



Dynamic Graph Neural Networks Under Spatio-Temporal Distribution Shift. ***NeurIPS, 2022.***

DIDA: Method

- Goal: create intervened distributions by sampling and reassembling variant patterns

$$\mathbf{z}_I^{t_1}(u), \mathbf{z}_V^{t_1}(u) \leftarrow \mathbf{z}_I^{t_1}(u), \mathbf{z}_V^{t_2}(v)$$

Dynamic Graph Neural Networks Under Spatio-Temporal Distribution Shift. ***NeurIPS, 2022.***

DIDA: Method

- Goal: focus on invariant patterns using intervened distributions
- Original objective:

$$\min_{\theta_1,\theta_2} \mathbb{E}_{(y^t,\mathcal{G}_v^{1:t})\sim p_{tr}(\mathbf{y}^t,\mathbf{G}_v^{1:t})} \mathcal{L}(f_{\theta_1}(\tilde{P}_I^t(v)), y^t)$$

$$s.t \quad \mathbf{y}^t \perp \tilde{\mathbf{P}}_V^t(v) \mid \tilde{\mathbf{P}}_I^t(v)$$

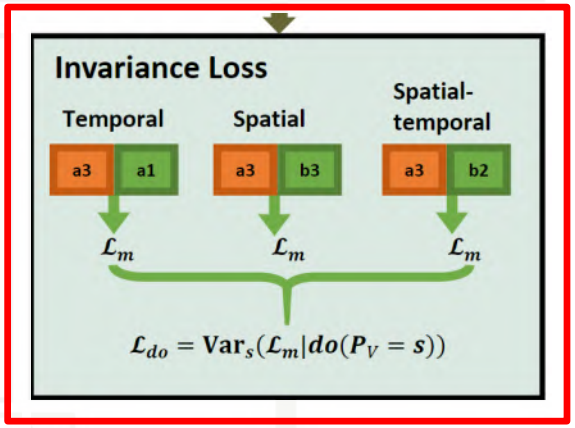
- Practical version: intervention-invariant regularization

$$\min_{\theta} \mathcal{L} + \lambda \mathcal{L}_{do}$$

$$\mathcal{L} = \ell(f(\mathbf{z}_I), \mathbf{y})$$

$$\mathcal{L}_m = \ell(g(\mathbf{z}_V, \mathbf{z}_I), \mathbf{y})$$

$$\mathcal{L}_{do} = \mathrm{Var}_{s_i \in \mathcal{S}}(\mathcal{L}_m | \mathrm{do}(\mathbf{P}_V^t = s_i))$$



Dynamic Graph Neural Networks Under Spatio-Temporal Distribution Shift. ***NeurIPS, 2022.***

DIDA: Experiments

- Synthetic datasets

Model \ $\overline{p}$	0.4		0.6		0.8	
Split	Train	Test	Train	Test	Train	Test
GCRN	69.60±1.14	72.57±0.72	74.71±0.17	72.29±0.47	75.69±0.07	67.26±0.22
EGCN	78.82±1.40	69.00±0.53	79.47±1.68	62.70±1.14	81.07±4.10	60.13±0.89
DySAT	84.71±0.80	70.24±1.26	89.77±0.32	64.01±0.19	94.02±1.29	62.19±0.39
IRM	85.20±0.07	69.40±0.09	89.48±0.22	63.97±0.37	**95.02±0.09**	62.66±0.33
VREx	84.77±0.84	70.44±1.08	89.81±0.21	63.99±0.21	94.06±1.30	62.21±0.40
GroupDRO	84.78±0.85	70.30±1.23	89.90±0.11	64.05±0.21	94.08±1.33	62.13±0.35
DIDA	**87.92±0.92**	**85.20±0.84**	**91.22±0.59**	**82.89±0.23**	92.72±2.16	**72.59±3.31**

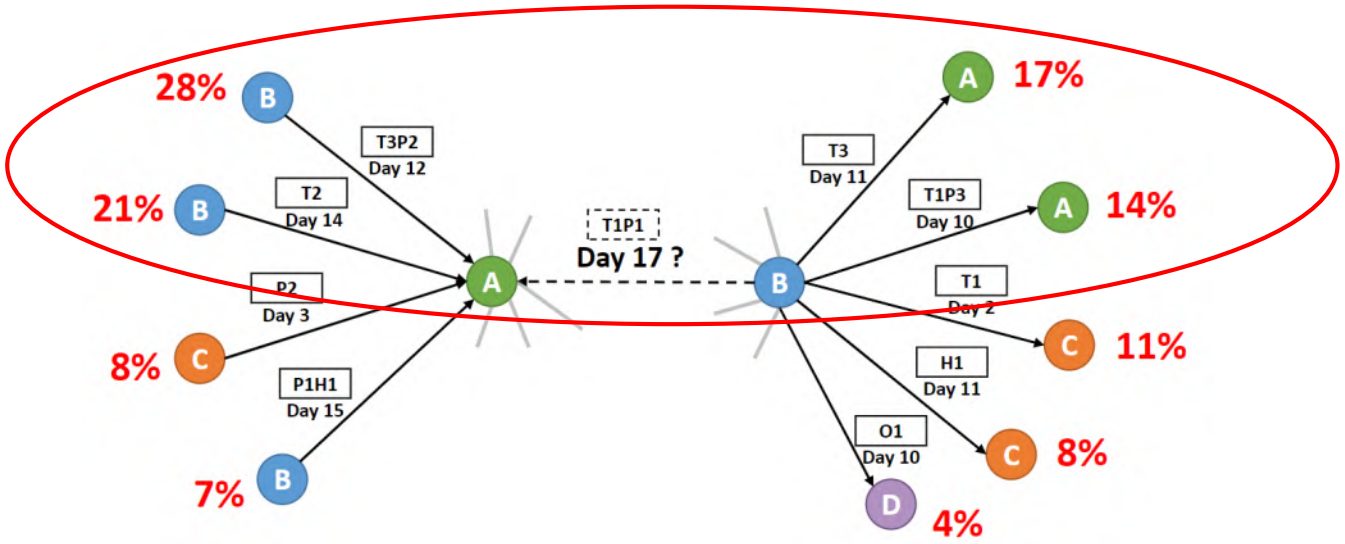
- Real-world datasets

Model	COLLAB		Yelp		Transaction	
Test Data	w/o DS	w/ DS	w/o DS	w/ DS	w/o DS	w/ DS
GAE	77.15±0.50	74.04±0.75	70.67±1.11	64.45±5.02	71.90±0.32	73.44±0.41
VGAE	86.47±0.04	74.95±1.25	76.54±0.50	65.33±1.43	79.31±0.37	75.66±0.30
GCRN	82.78±0.54	69.72±0.45	68.59±1.05	54.68±7.59	78.99±0.28	71.24±0.35
EGCN	86.62±0.95	76.15±0.91	78.21±0.03	53.82±2.06	73.22±1.11	66.49±0.97
DySAT	88.77±0.23	76.59±0.20	78.87±0.57	66.09±1.42	81.55±0.66	76.18±0.43
IRM	87.96±0.90	75.42±0.87	66.49±10.78	56.02±16.08	81.65±0.50	75.61±0.61
VREx	88.31±0.32	76.24±0.77	79.04±0.16	66.41±1.87	81.72±0.35	76.24±0.52
GroupDRO	88.76±0.12	76.33±0.29	**79.38±0.42**	66.97±0.61	81.50±0.24	75.92±0.37
DIDA	**91.97±0.05**	**81.87±0.40**	78.22±0.40	**75.92±0.90**	**83.08±0.33**	**77.61±0.59**

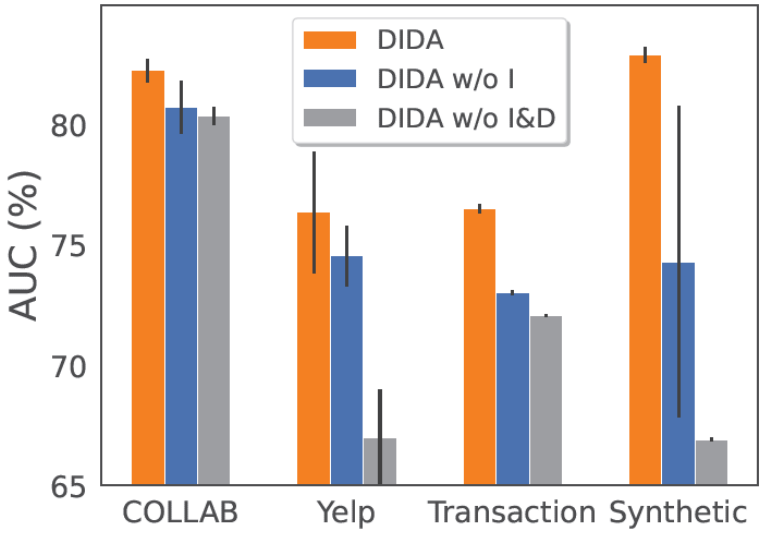
Dynamic Graph Neural Networks Under Spatio-Temporal Distribution Shift. ***NeurIPS, 2022.***

DIDA: Experiments

- Showcases:

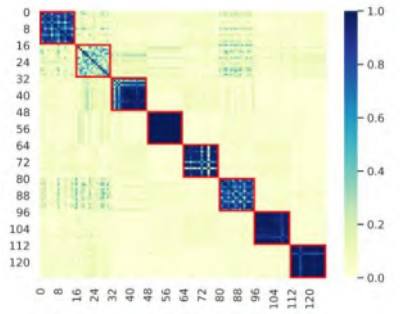


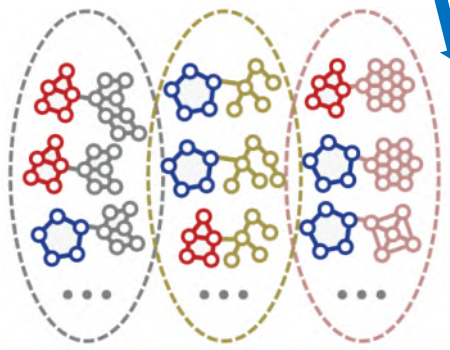
- Ablation studies:



Dynamic Graph Neural Networks Under Spatio-Temporal Distribution Shift. ***NeurIPS, 2022.***

Recap





Out-of-distribution Generalization Graph Learning

Encourage Independence and Disentanglement in representations

- OOD-GNN: Out-of-Distribution Generalized Graph Neural Network — TKDE'22
- Self-supervised
- Disentangled Graph Contrastive Learning with Independence Promotion — TKDE'22

Customize unique GNN architectures for graphs

- Graph Neural Architecture Search under Distribution Shifts — ICML'22

Finding invariant and variant graph structures

- Learning Invariant Graph Representations under Distribution Shifts — NeurIPS'22
- Dynamic Graphs
- Dynamic Graph Neural Networks under Spatio-Temporal Distribution Shift — NeurIPS'22

Survey

Out-Of-Distribution Generalization on Graphs: A Survey

Haoyang Li* , **Xin Wang*** , **Ziwei Zhang** and **Wenwu Zhu**
Tsinghua University, Beijing, China
lihy18@mails.tsinghua.edu.cn, {xin_wang, zwzhang, wwzhu}@tsinghua.edu.cn

Abstract

Graph machine learning has been extensively studied in both academia and industry. Although booming with a vast number of emerging methods and techniques, most of the literature is built on the I.I.D. hypothesis, i.e., testing and training graph data are independent and identically distributed. However, this I.I.D. hypothesis can hardly be satisfied in many real-world graph scenarios where the model performance substantially degrades when there exist distribution shifts between testing and training graph data. To solve this critical problem, out-of-distribution (OOD) generalization on graphs, which goes beyond the I.I.D. hypothesis, has made great progress and attracted ever-especially for graph neural networks (GNNs), have shown great successes in both academia and industry, illustrating their excellent capabilities in various applications, e.g., social network analysis [Qiu *et al.*, 2018], recommendation systems [Wu *et al.*, 2020], knowledge representation [Wang *et al.*, 2017], traffic forecasting [Yu *et al.*, 2018], etc.

Despite the notable success of graph machine learning approaches, the existing literature generally relies on the assumption that the testing and training graph data are independently drawn from the identical distribution, i.e., I.I.D. hypothesis. However, in real-world scenarios, such a hypothesis is difficult to be satisfied due to the uncontrollable underlying data generation mechanism [Bengio *et al.*, 2019]. In practice, there will inevitably be scenarios with distribution shifts between testing and training graphs [Li *et al.*, 2021b; Wang *et al.*, 2021b], which may cause significant perfor-

Out-Of-Distribution Generalization on Graphs: A Survey. *arXiv:2202.07987*.

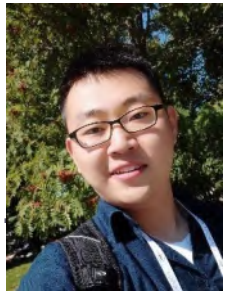
Paper collection: https://github.com/THUMNLab/awesome-graph-ood

Acknowledgements

Wenwu Zhu
Tsinghua Univ.

Peng Cui
Tsinghua Univ.

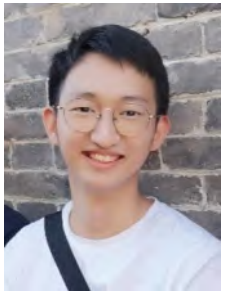
Haoyang Li
Tsinghua Univ.



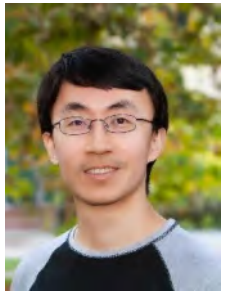
Yijian Qin
Tsinghua Univ.

Xin Wang
Tsinghua Univ.

Zeyang Zhang
Tsinghua Univ.



Pengtao Xie
UCSD



THANK YOU!

https://zw-zhang.github.io
zwzhang@tsinghua.edu.cn