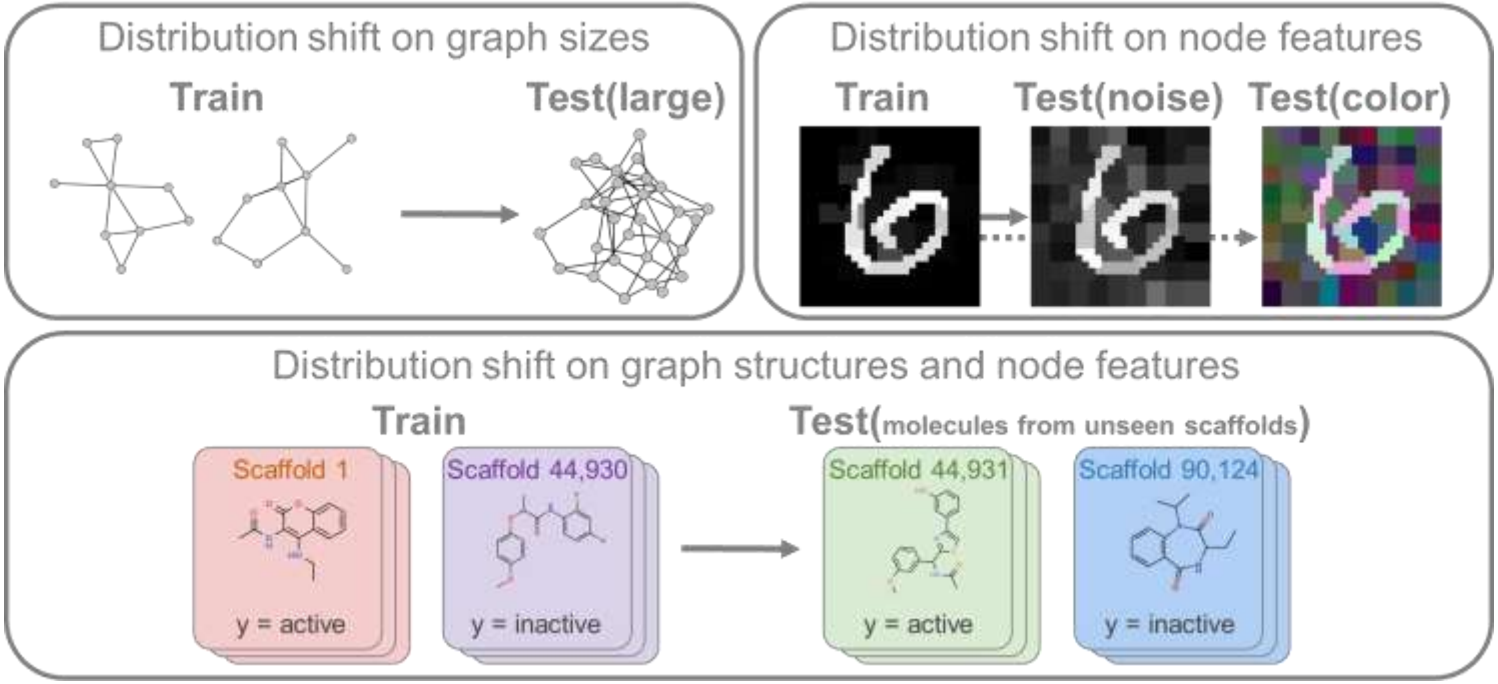
Invariance-guided Graph Representation Learning

(Joint work with Haoyang Li, Yijian Qin, Zeyang Zhang, Haonan Yuan, Yang Yao, Jie Cai, Peiwen Li, Tianyin Liao)

Graph Distribution Shifts

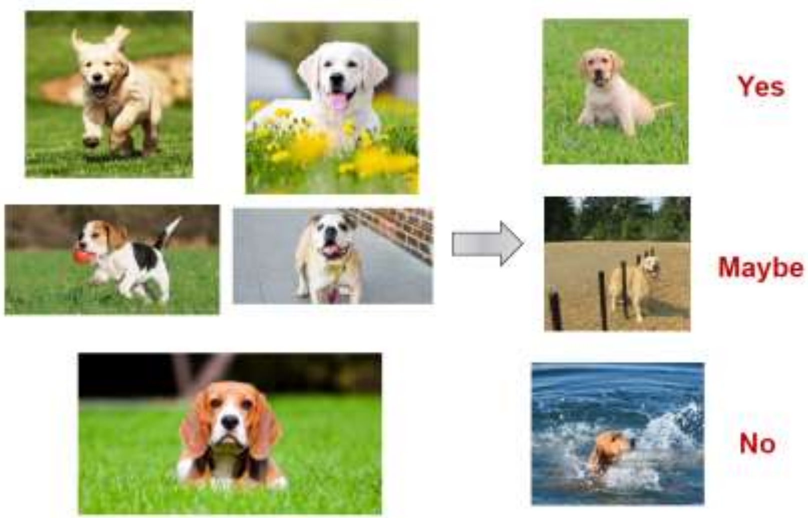
- In dynamic and open environments, distribution shifts naturally occur



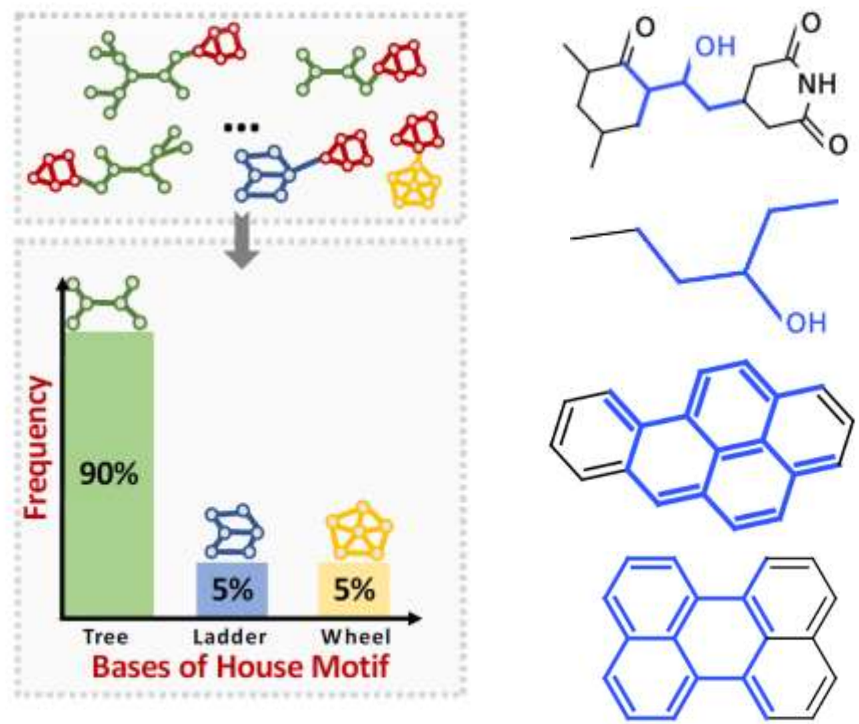
Graph distribution shifts become a major factor for distortion

Main Challenge for Handling Distribution Shifts

- Why existing GRL fail to handle distribution shifts and achieve OOD generalization?
- Answer: **spurious correlations**
  - GRL tends to exploit statistical correlations in the training set
  - But spurious correlations cannot generalize under distribution shifts



computer vision

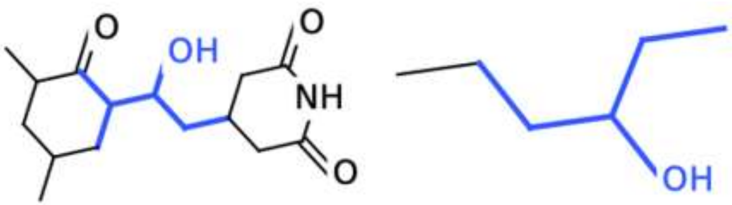


graphs

Picture credit: Convolutional Networks on Graphs for Learning Molecular Fingerprints, NeurIPS 2015; Discovering Invariant Rationales for Graph Neural Networks, ICLR 2022

Invariance-guided Graph Representation Learning

- How to get rid of spurious correlations?
- 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

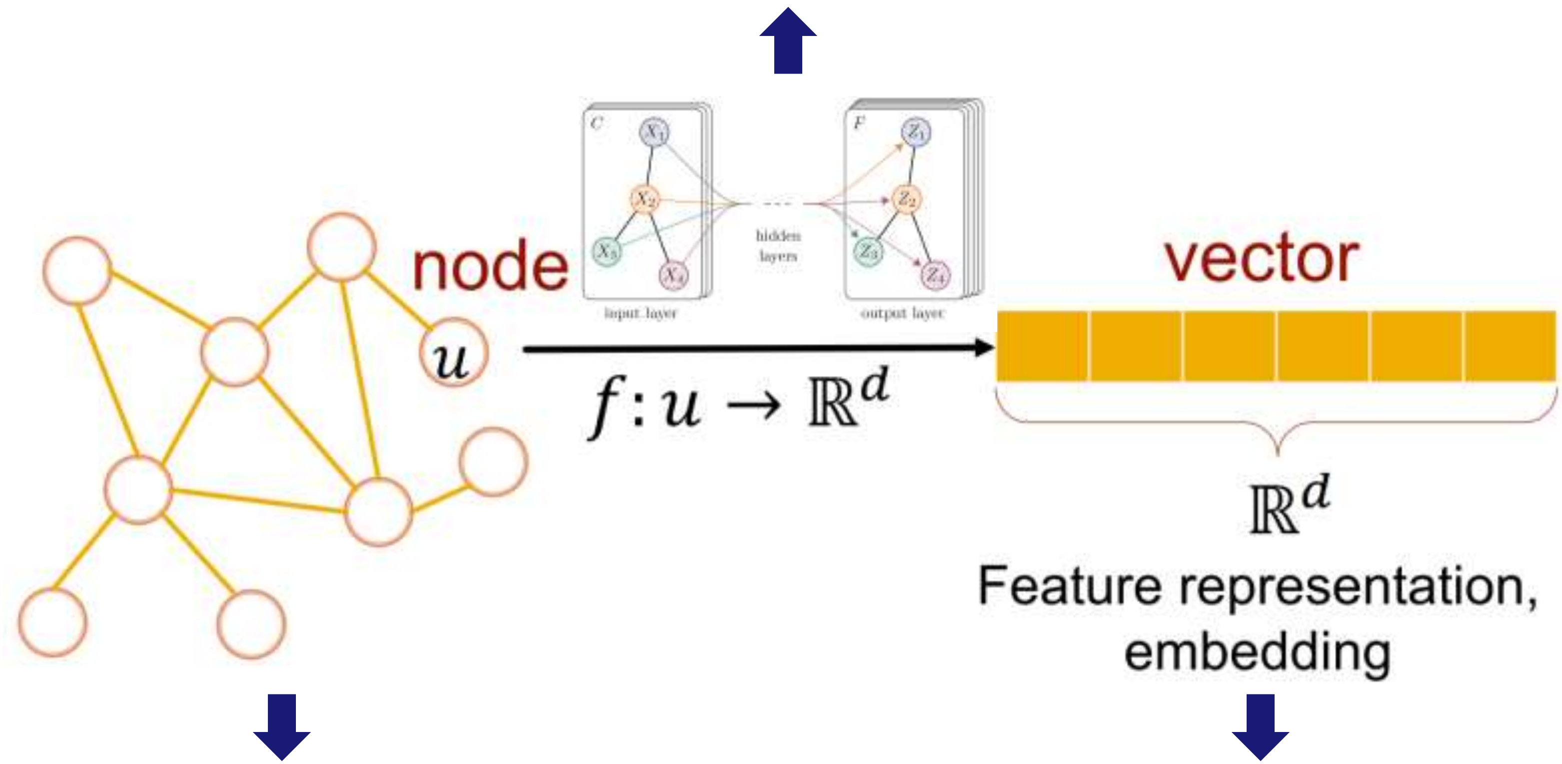
For Dog Classification

Invariance

Variance

Finding Invariance

How to find invariance in GNN architectures?



How to find invariance in the topology space?

How to find invariance in the vector space?

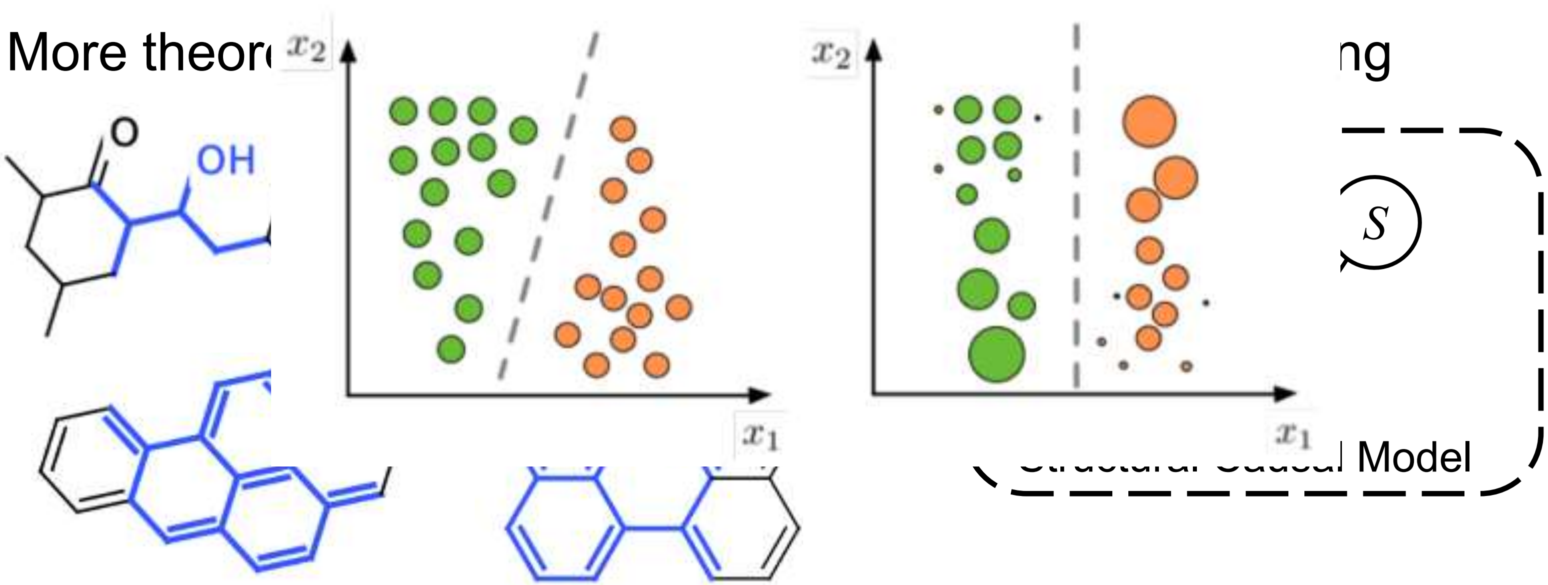
Picture credit: Kipf and Welling (GCN, ICLR 2017), Jure Leskovec (CS224w, Stanford Univ.)

Graph Invariant Learning in the Vector Space

- OOD-GNN (IEEE TKDE’22)
- StableGNN (TPAMI’23)
- IDGCL (IEEE TKDE’22)
- OOD-GCL (ICML’24)

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 theor… …ng



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:

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

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

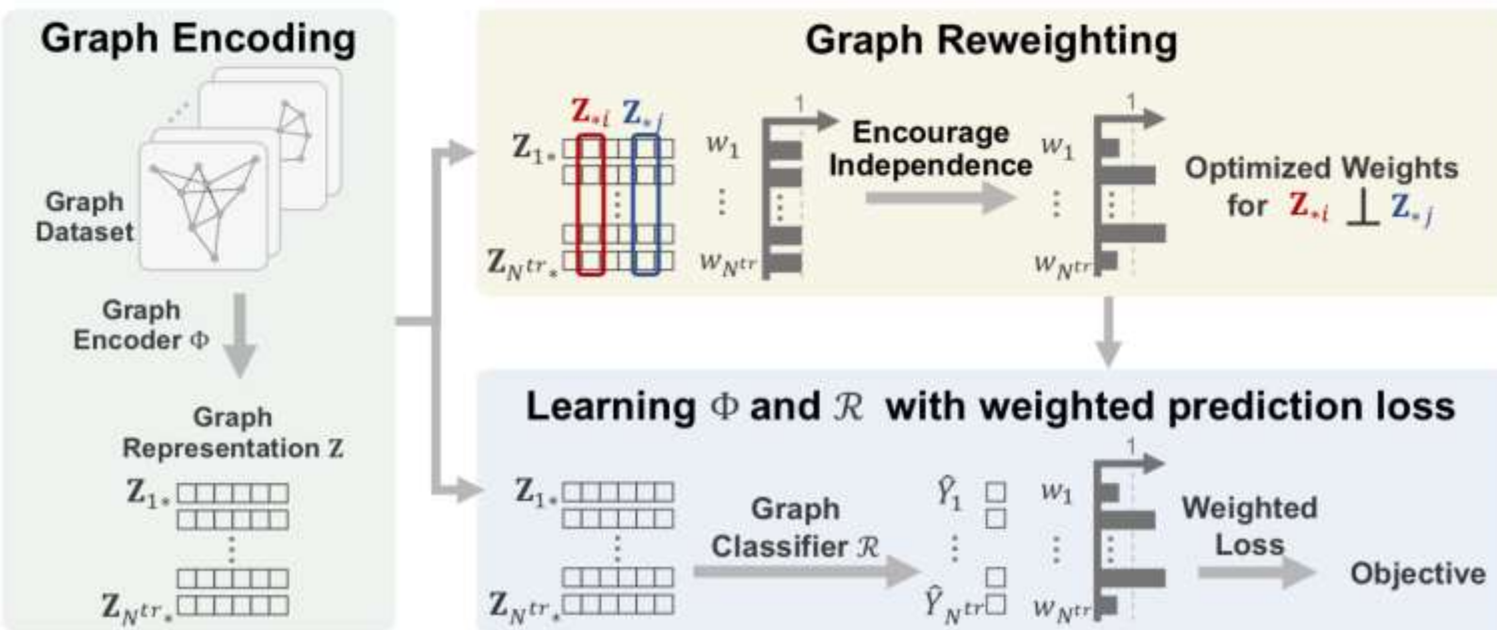
OOD-GNN: Out-of-Distribution Generalized Graph Neural Network. ***TKDE, 2022.***

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:

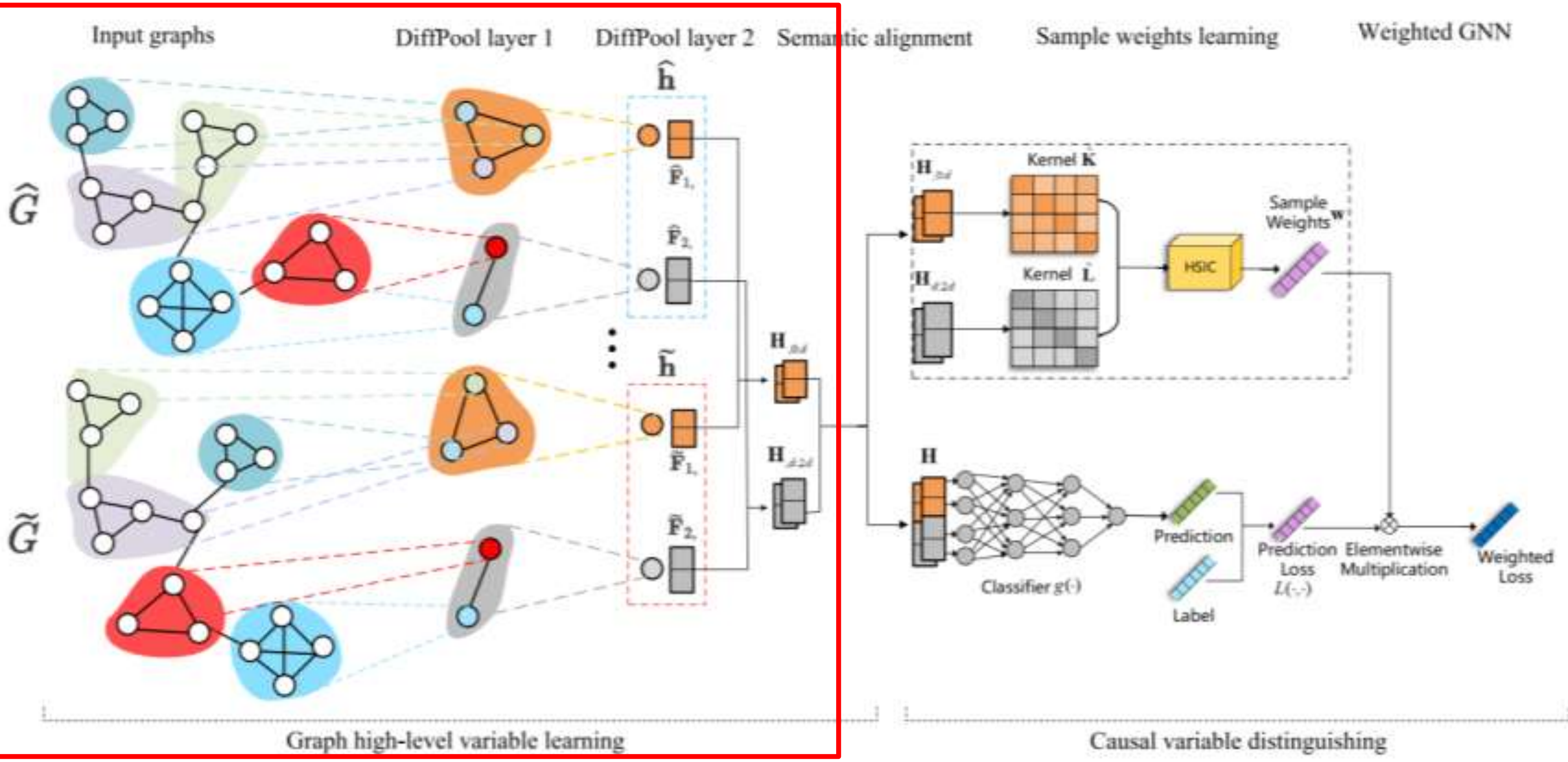
TABLE 5: Results on nine Open Graph Benchmark (OGB) datasets. We report the ROC-AUC (%) for classification tasks and RMSE for regression tasks with the standard deviation on the **test set** of all methods. None of the baseline methods is consistently competitive across all datasets, while our proposed 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.***

StableGNN

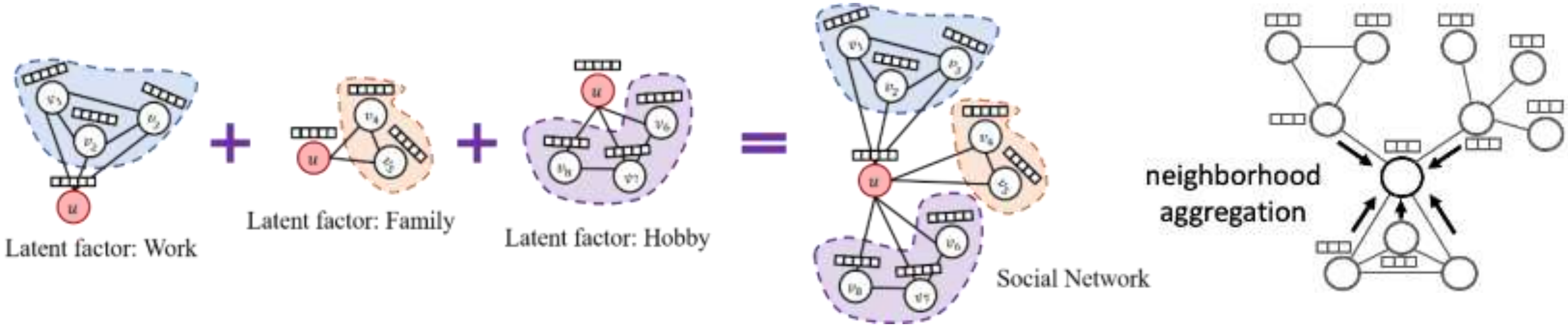
- OOD-GNN: does not delve into complicated graph structure
- Main idea: encode and remove subgraph-level spurious correlations
  - Employ graph pooling layers to learn high-level graph representation



Generalizing Graph Neural Networks on Out-of-Distribution Graphs, ***TPAMI 2023***

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?

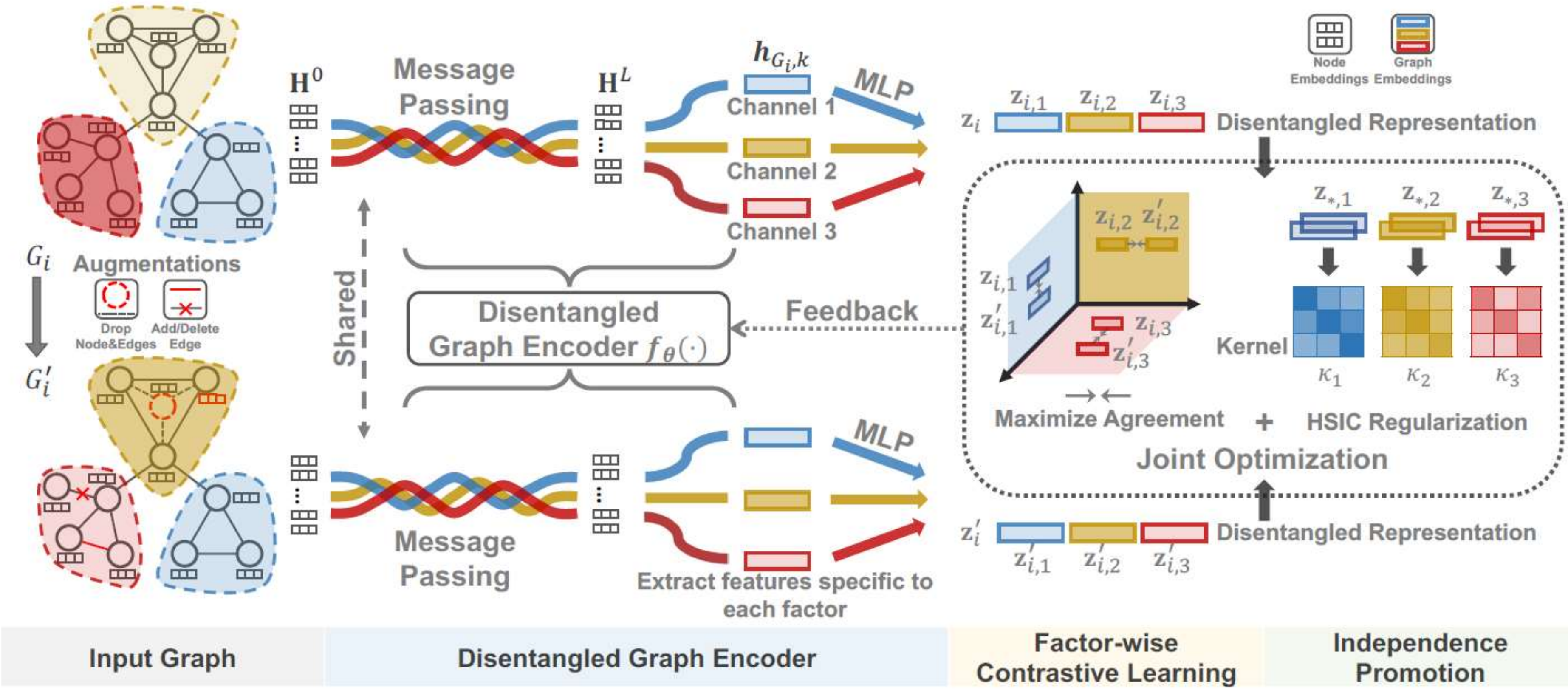
- The graph labels can be extremely scarce for many graph datasets/scenarios

→ Can we design self-supervised learning frameworks?

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

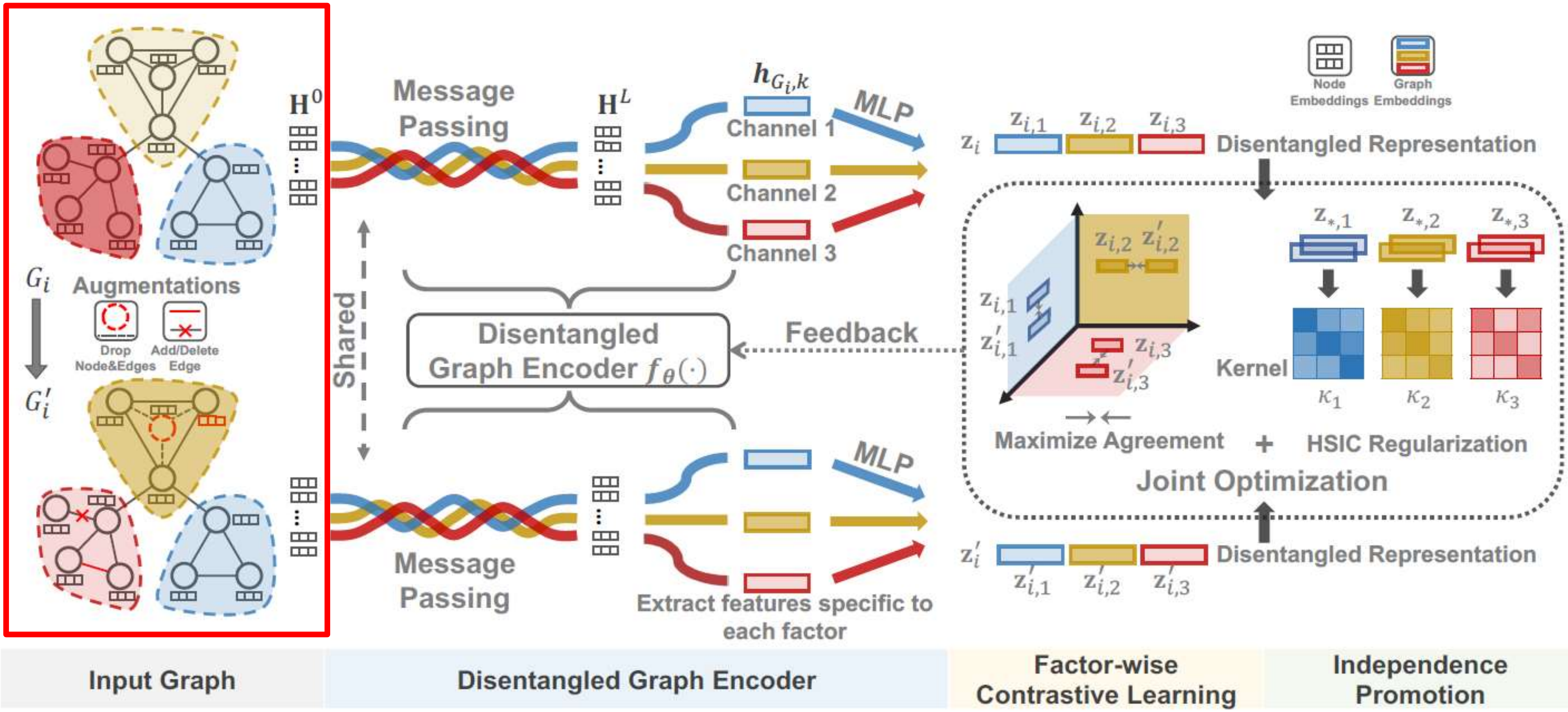
IDGCL: Method

- **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

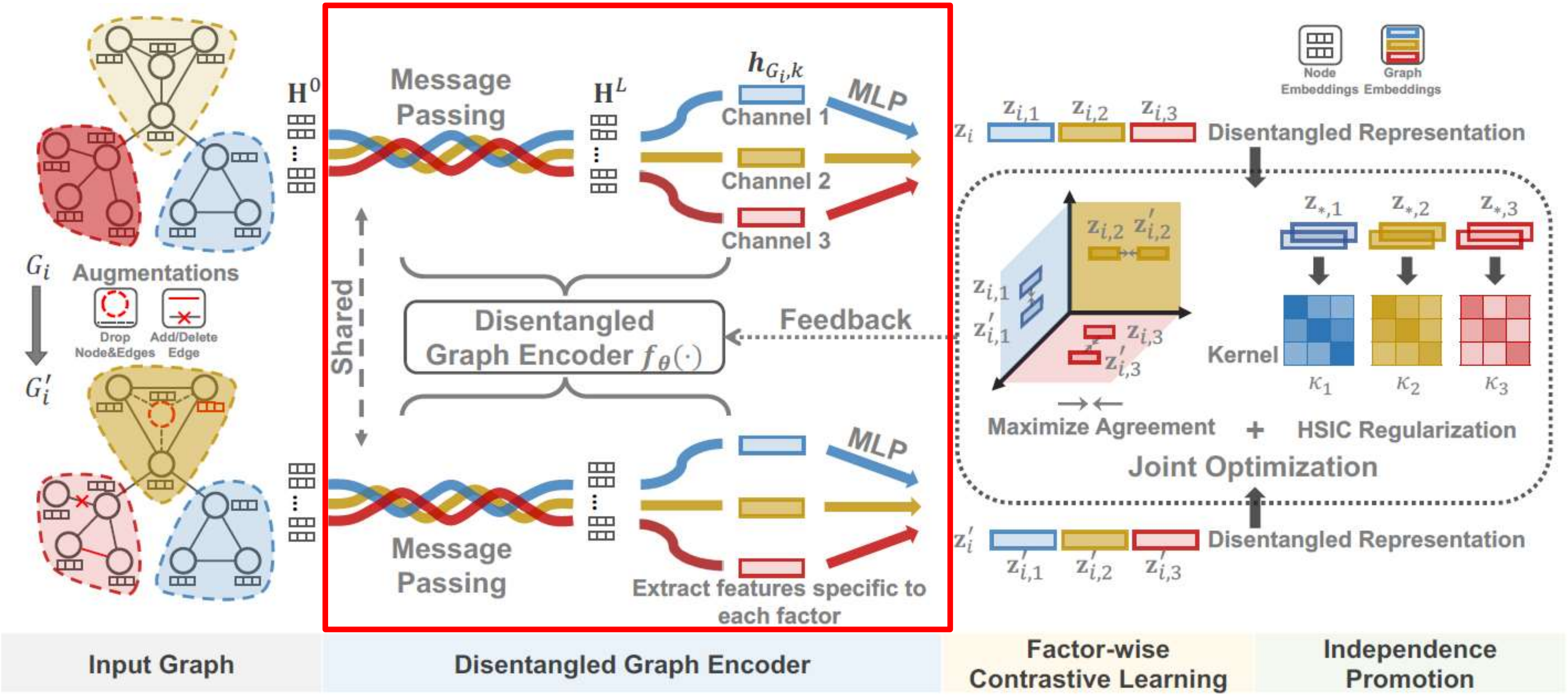


- Graph Augmentation
  - Four types of strategies: node dropping, edge perturbation, attribute masking, subgraph sampling
  - Reflect diverse aspects behind graphs, can be directly extended
  - Self-supervised loss:

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

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:

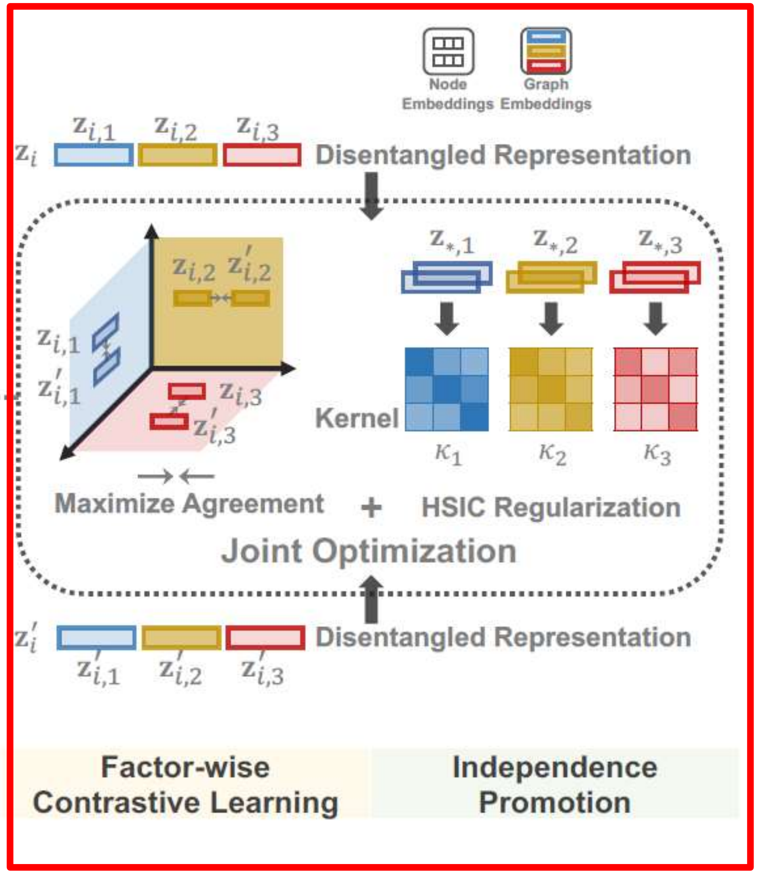
$$p_\theta(k|G_i) = \frac{\exp \phi(\mathbf{z}_{i,k}, \mathbf{c}_k)}{\sum_{k=1}^{K} \exp \phi(\mathbf{z}_{i,k}, \mathbf{c}_k)}$$

  - 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:

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

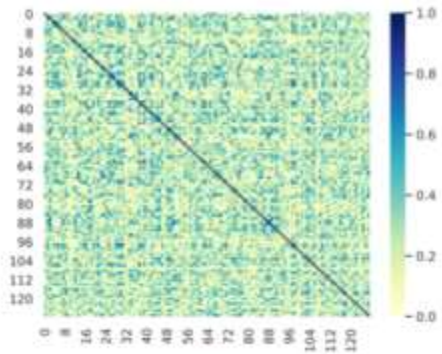


- Overall objective function: $\min_{\theta} \mathcal{L}(\theta, \mathcal{B}) + \lambda \mathcal{L}_{reg}$

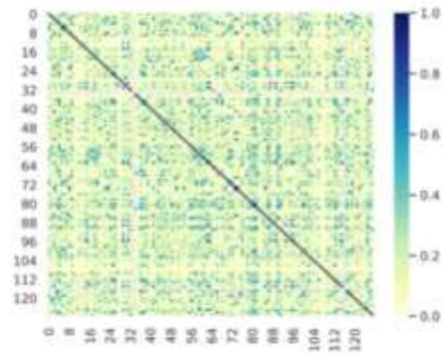
Disentangled Graph Contrastive Learning with Independence Promotion. ***TKDE, 2022.***

IDGCL: Experiments

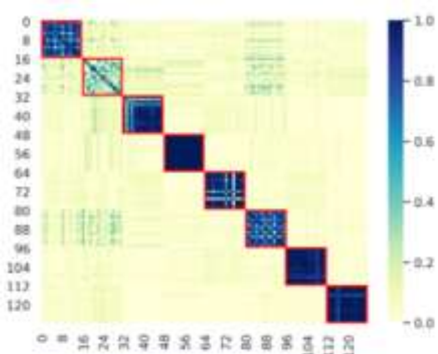
- Visualizations for representations



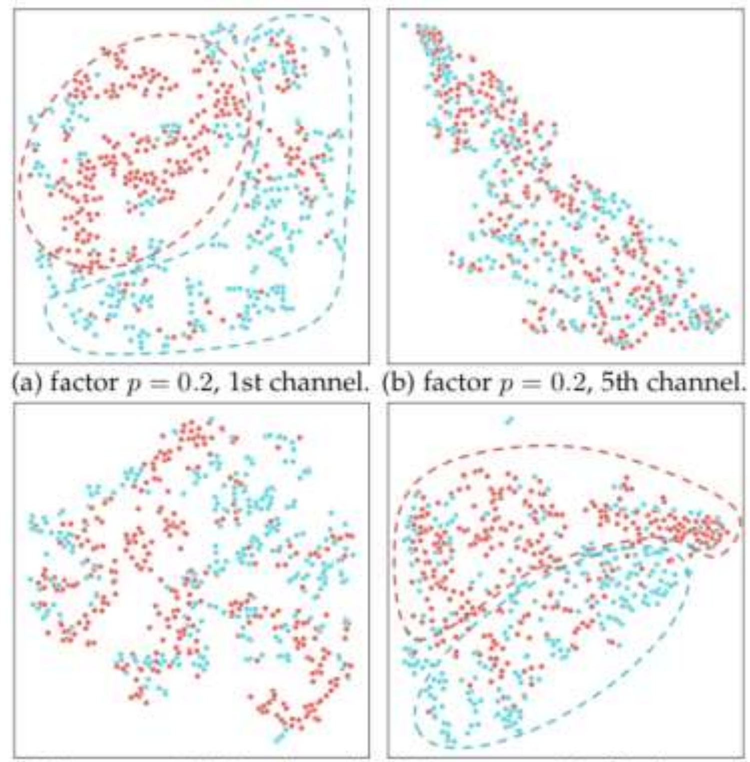
MVGRL



GraphCL



IDGCL



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

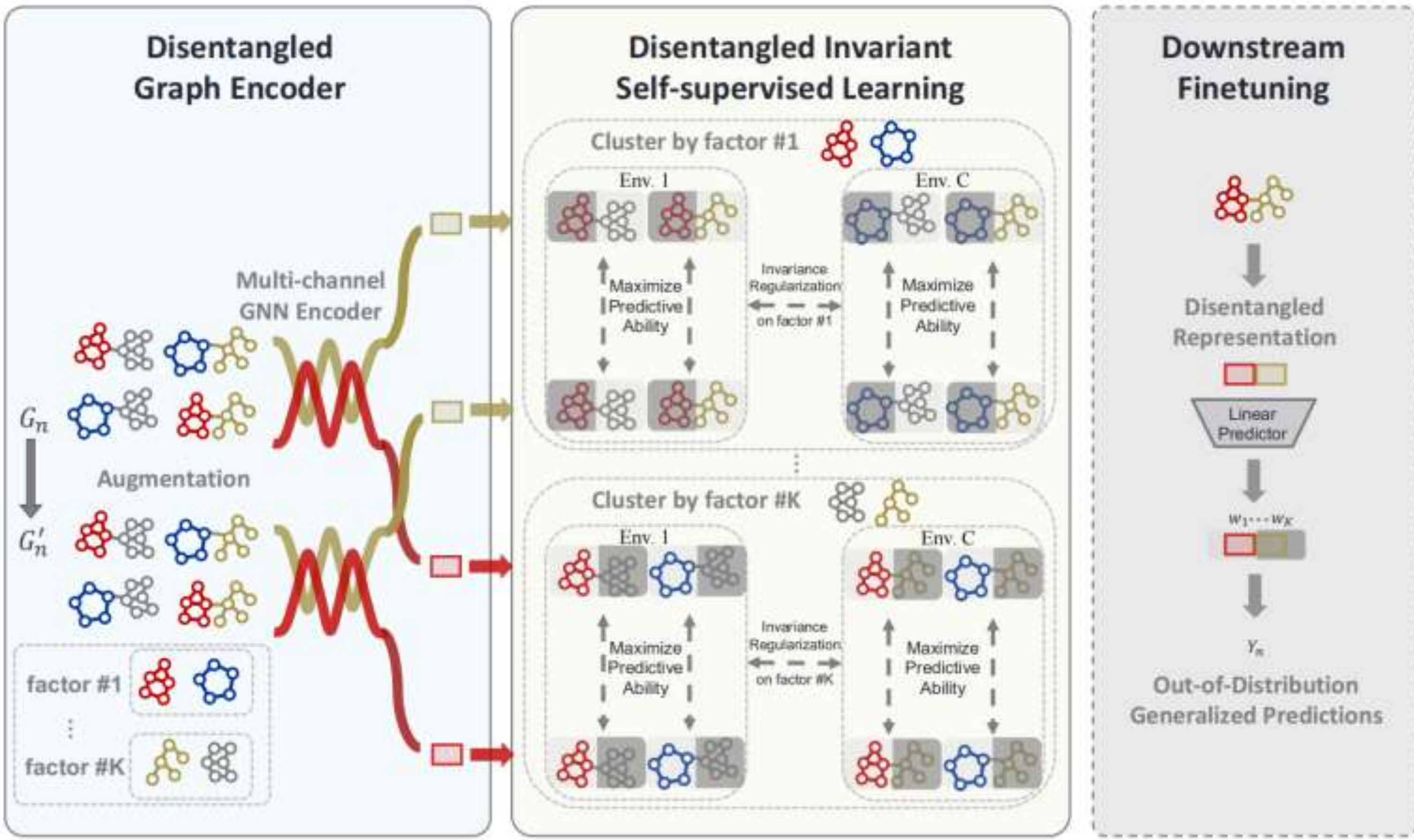
(c) factor $p = 0.9$, 1st channel. (d) factor $p = 0.9$, 5th channel.

Each channel captures one latent factor

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

OOD-GCL

- Can we move a step forward to the "pre-training, fine-tuning" paradigm?
- Main idea: learn disentangled invariant graph representation for *K* latent clusters
  - Can be used as pre-training and easily fine-tuned for downstream tasks



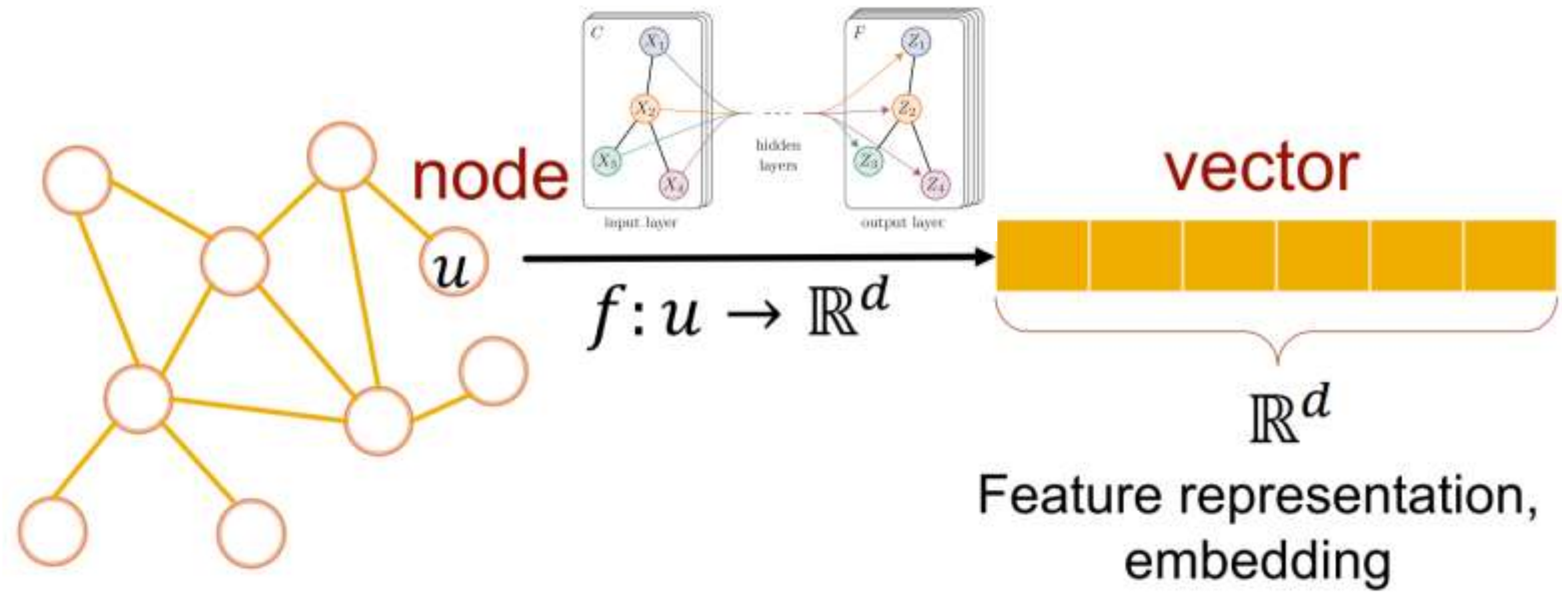
Disentangled Graph Self-supervised Learning for Out-of-Distribution Generalization. ***ICML, 2024.***

Recap: Graph Invariant Learning in the Vector Space

- OOD-GNN (IEEE TKDE’22): sample reweighting for decorrelation
- StableGNN (TPAMI’23): pooling for subgraph-level decorrelation
- IDGCL (IEEE TKDE’22): self-supervised decorrelation
- OOD-GCL (ICML’24): “pre-training, fine-tuning” decorrelation

Finding Invariance

How to find invariance in GNN **architectures**?



How to find invariance in the **topology space**?

How to find invariance in the **vector space**?

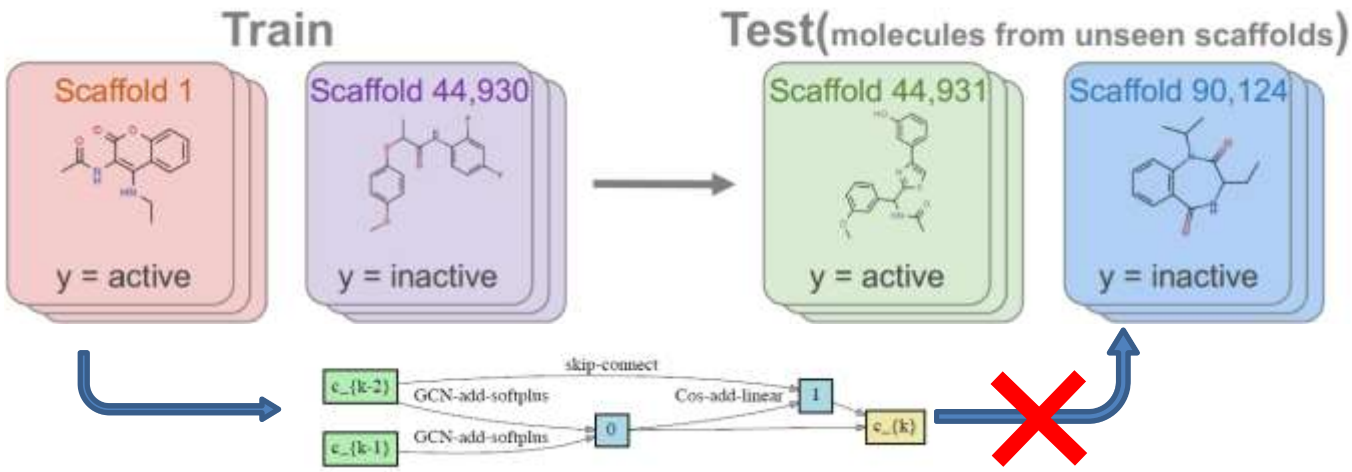
Picture credit: Kipf and Welling (GCN, ICLR 2017), Jure Leskovec (CS224w, Stanford Univ.)

Graph Invariant Learning in the Architecture Space

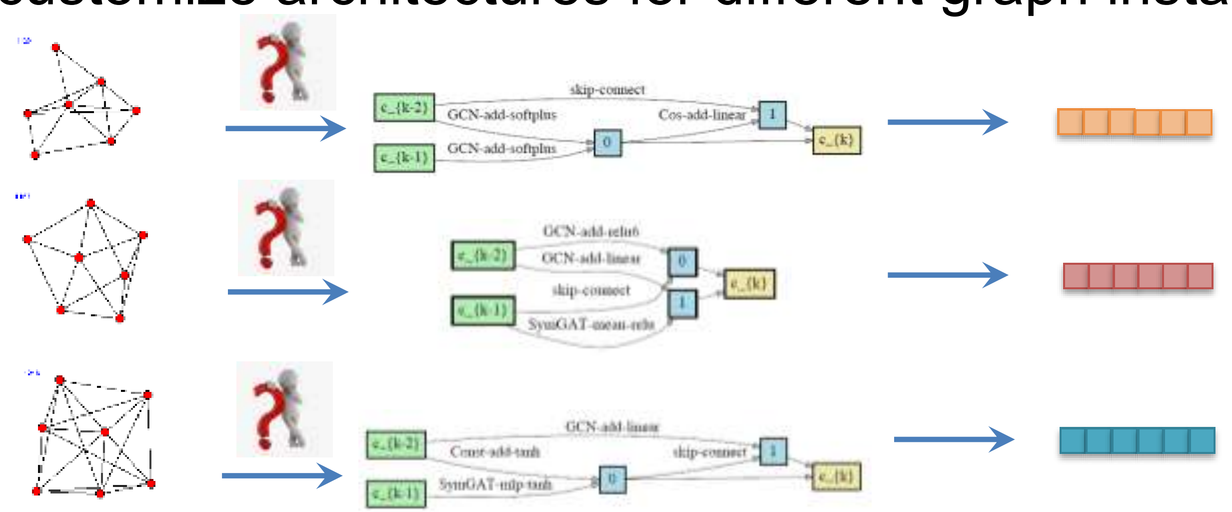
- GRACES (ICML’22)
- OMGNAS (AAAI’24)
- DCGAS (AAAI’24)
- CARNAS (KDD’25)

Challenge of Fixed Architectures

- A fixed architecture on the training data may fail to generalize

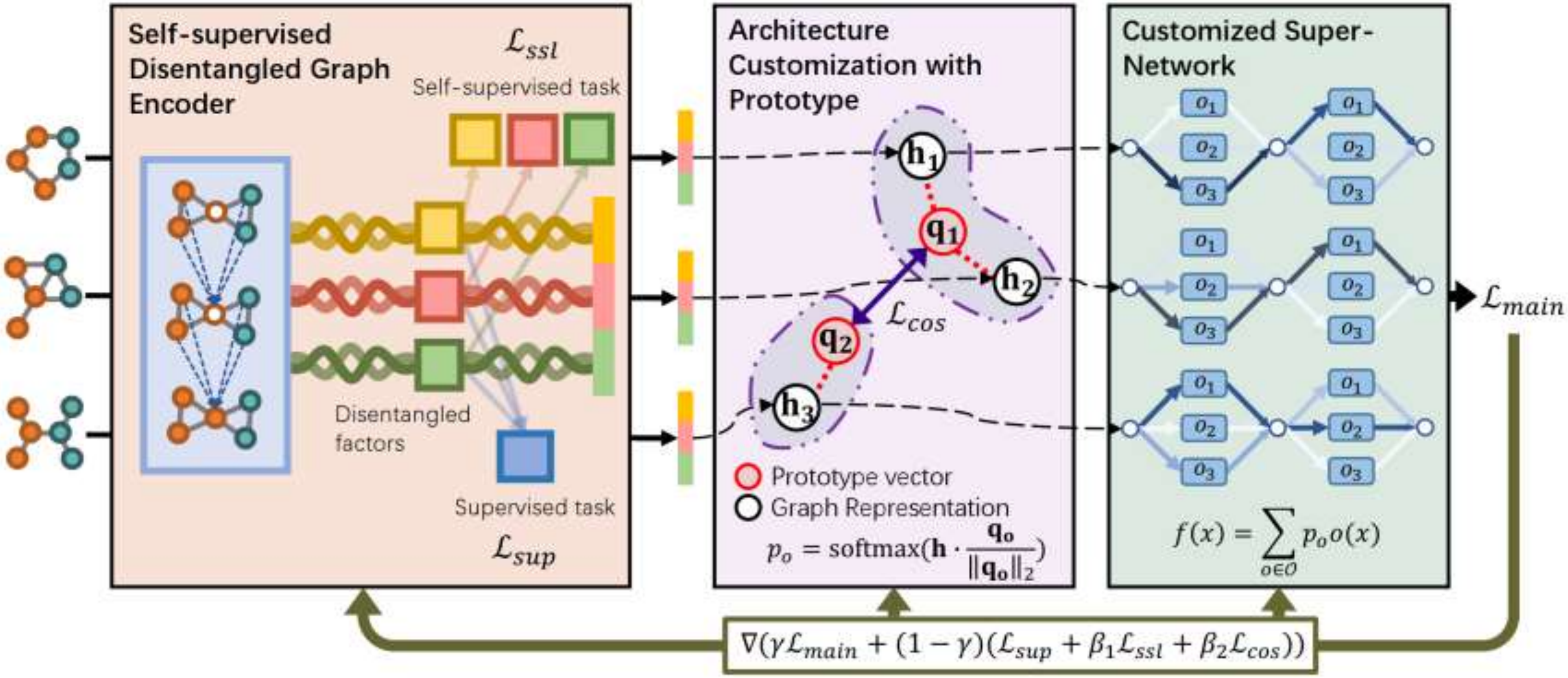


- Solution: customize architectures for different graph instances



GRACES: Graph Neural Architecture Search under Distribution Shifts

- Main idea: customize a unique GNN architecture for each graph instance to handle distribution shifts



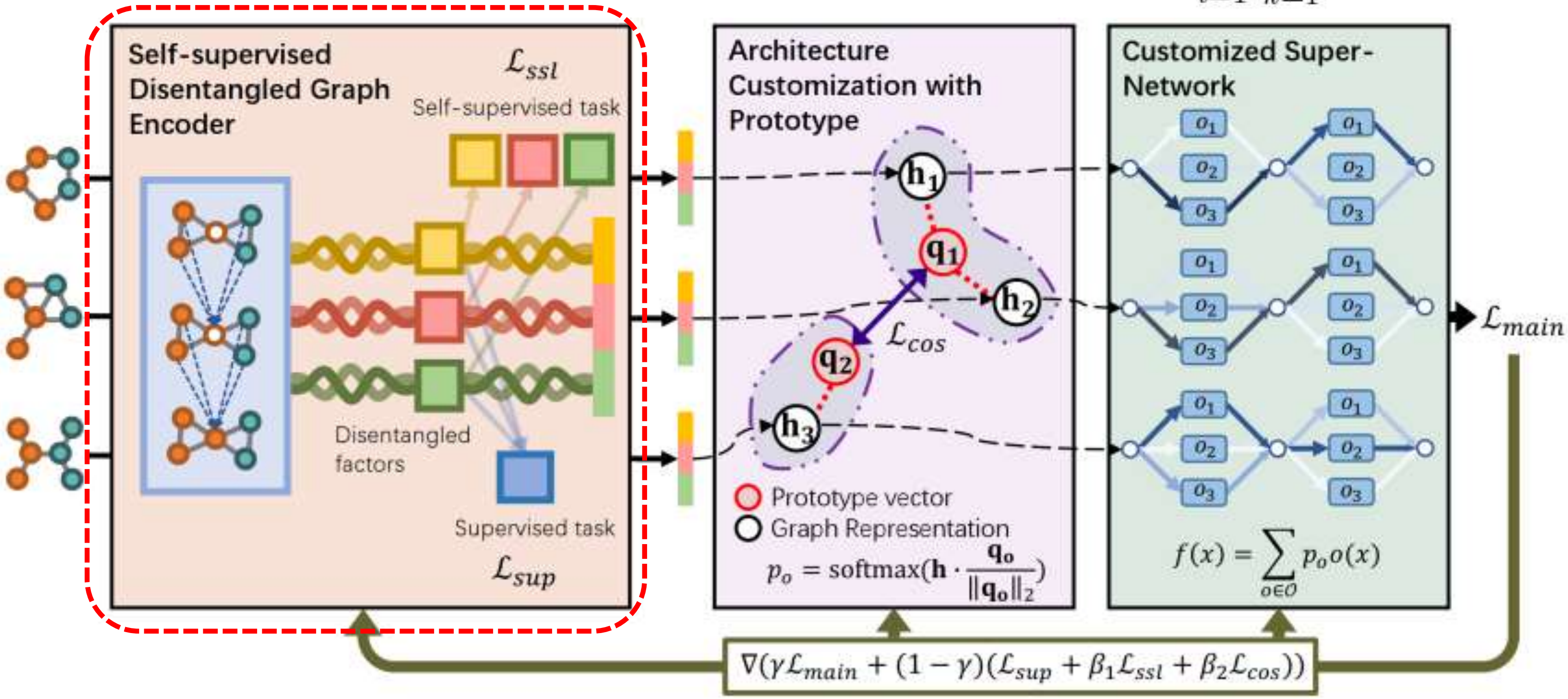
Graph Neural Architecture Search under Distribution Shifts. ***ICML, 2022.***

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
  - Supervised loss: the downstream task
  - Self-supervised loss: node degree as regularization

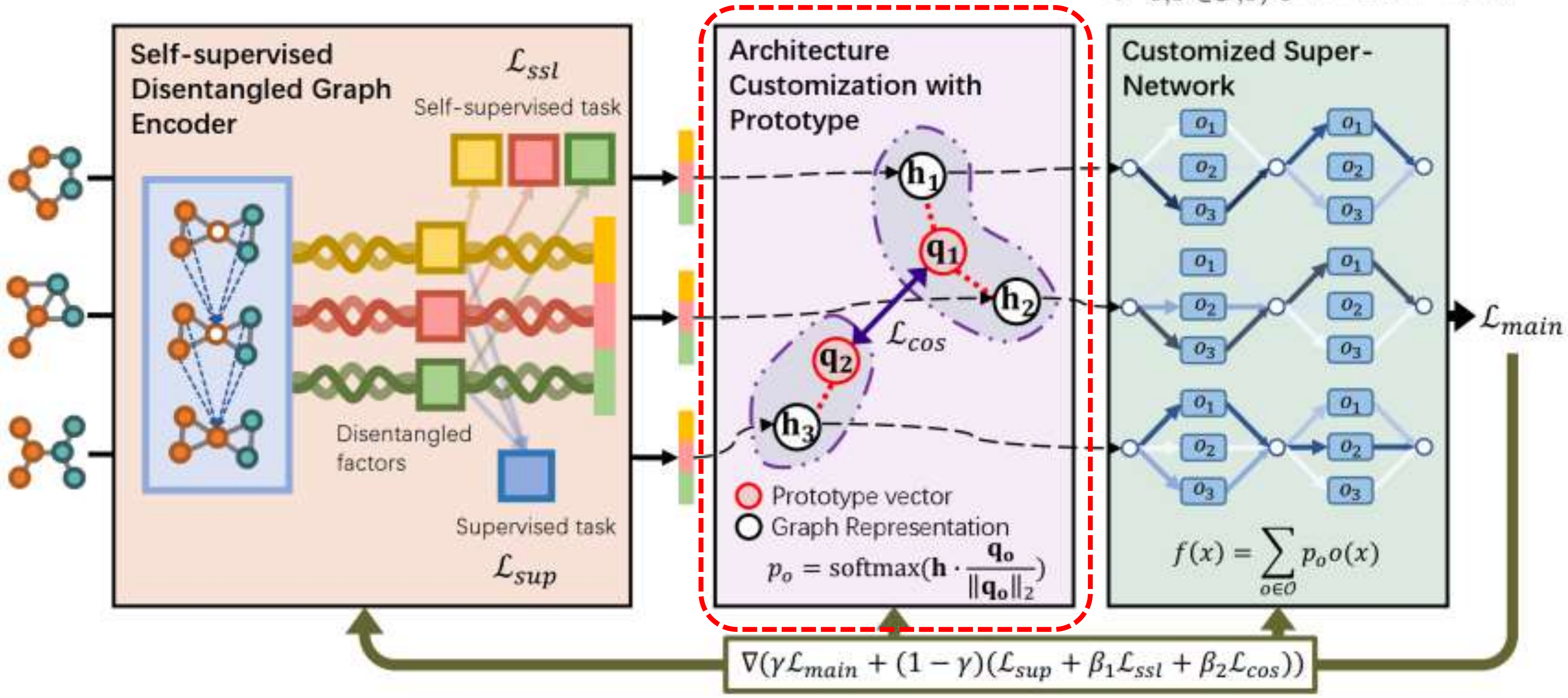
$$\mathbf{H}^{(l)} = \Big\Vert_{k=1}^{K} \mathrm{GNN}(\mathbf{H}_k^{(l-1)}, \mathbf{A}) \quad \mathcal{L}_{sup} = \sum_{i=1}^{N_{tr}} \ell\left(\mathcal{C}\left(\mathbf{h}_i\right), y_i\right)$$

$$\mathcal{L}_{ssl} = \sum_{i=1}^{N_{tr}} \sum_{k=1}^{K-1} \ell_{ssl}\left(\hat{y}_{i,k}^{ssl}, y_{i,k}^{ssl}\right)$$



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: $\hat{p}_o^i = \mathbf{h} \cdot \frac{\mathbf{q}_o^i}{\|\mathbf{q}_o^i\|_2}, p_o^i = \frac{\exp(\hat{p}_o^i)}{\sum_{o' \in \mathcal{O}} \exp(\hat{p}_{o'}^i)},$
  - Regularizer to avoid mode collapse: $\mathcal{L}_{cos} = \sum_i \sum_{o,o' \in \mathcal{O}, o \neq o'} \frac{\mathbf{q}_o^i \cdot \mathbf{q}_{o'}^i}{\|\mathbf{q}_o^i\|_2 \|\cdot \mathbf{q}_{o'}^i\|_2}$

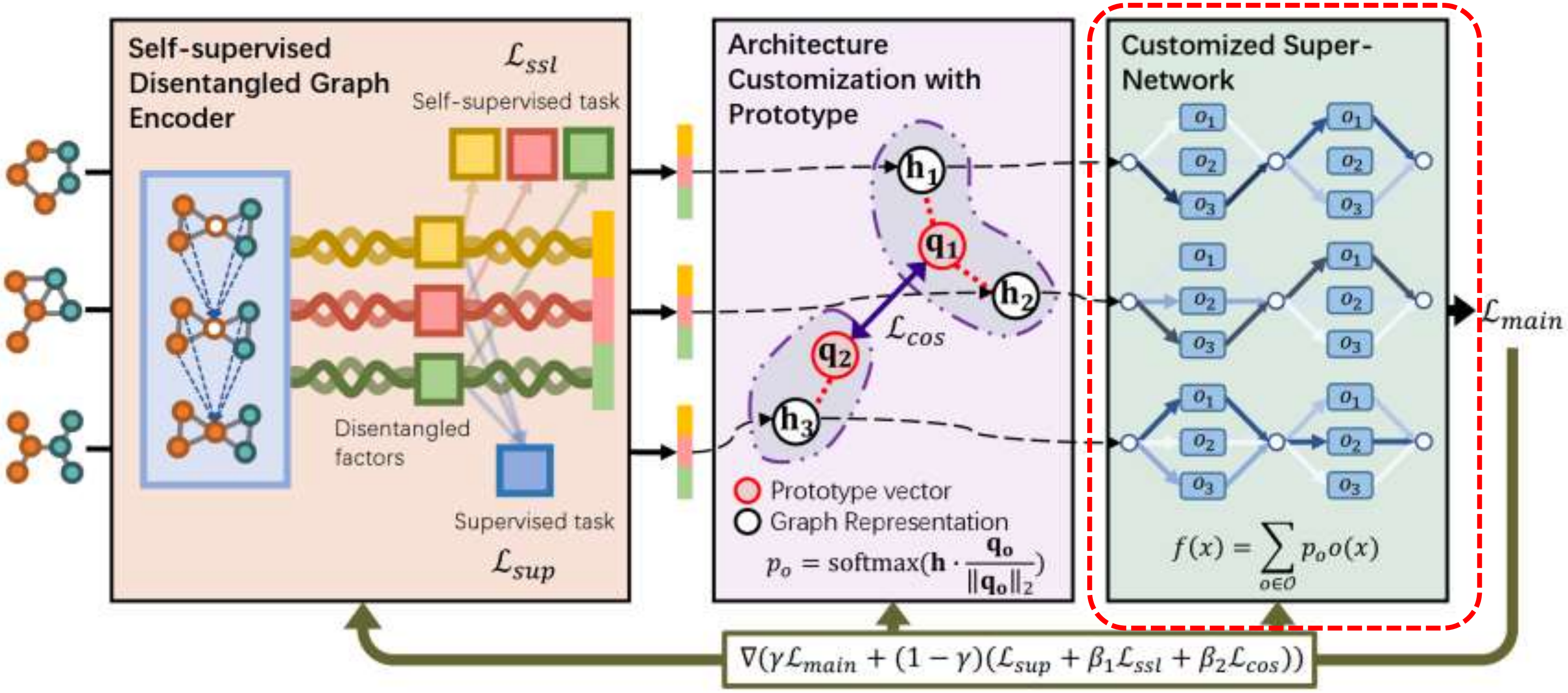


GRACES: Learning Architecture Parameters

- **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:**

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

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



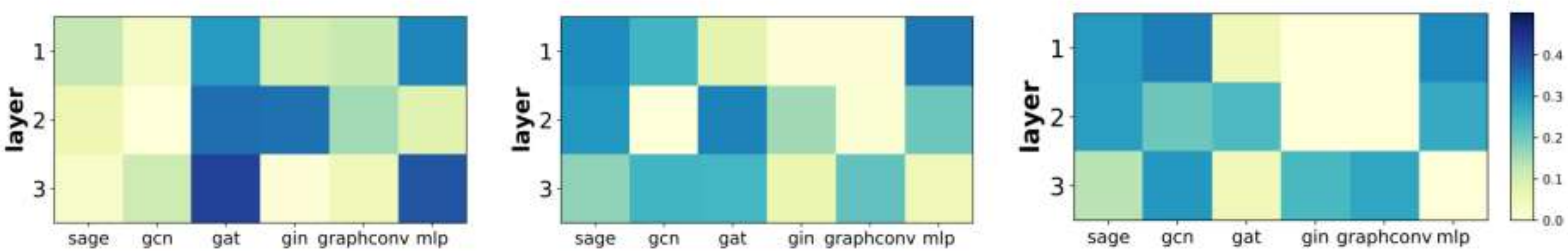
GRACES: Experiments

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	$\mathbf{65.72}_{\pm 17.47}$	$\mathbf{59.57}_{\pm 17.37}$	$\mathbf{50.94}_{\pm 8.14}$

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	$\mathbf{77.31}_{\pm 1.00}$	$\mathbf{61.85}_{\pm 2.56}$	$\mathbf{79.46}_{\pm 3.04}$

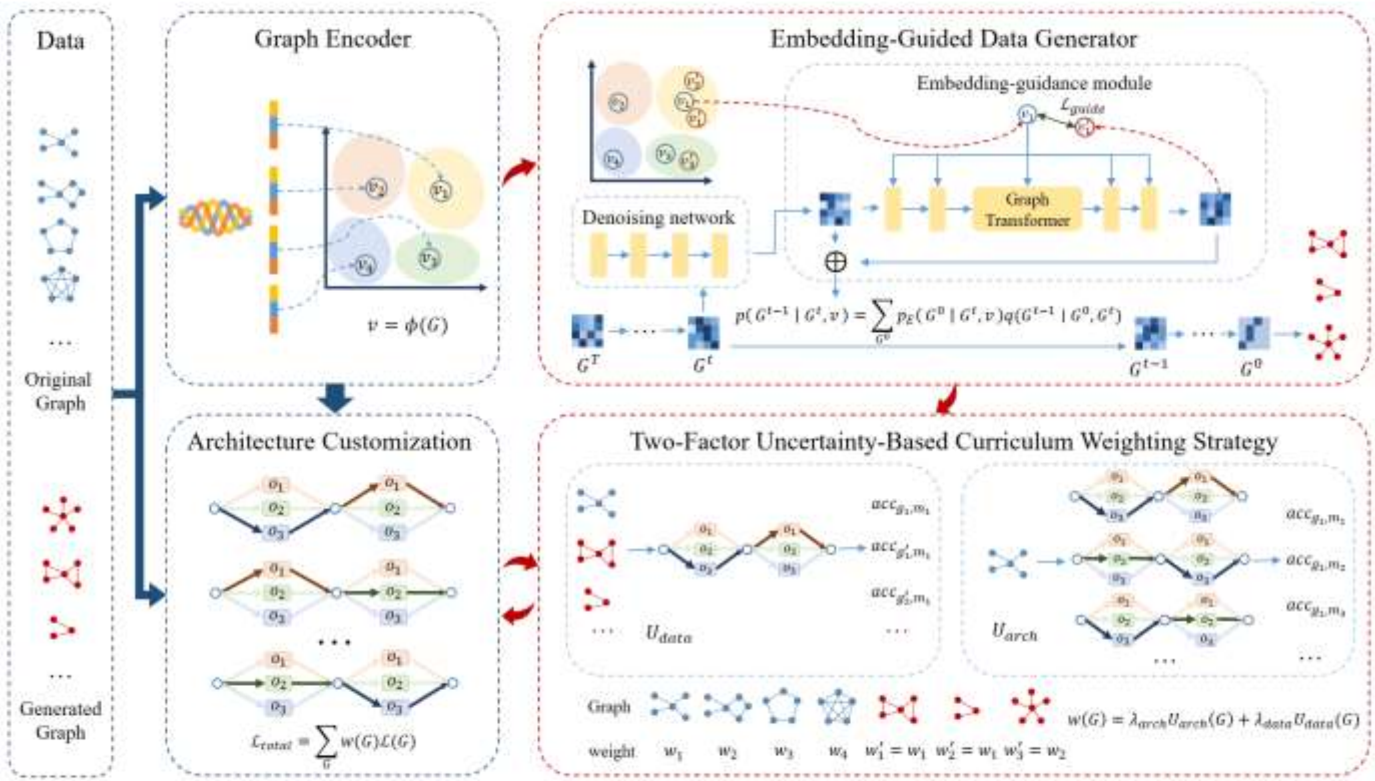


(a) *Tree*-based graphs (b) *Ladder*-based graphs (c) *Wheel*-based graphs

Customization of architectures

DCGAS: Data-Augmented Curriculum GraphNAS

- Main idea: GRACES only focus on searched architectures
  - Training data can be limited → can we augment existing graph?
  - Importance of graph is different → can we design clever learning strategy?

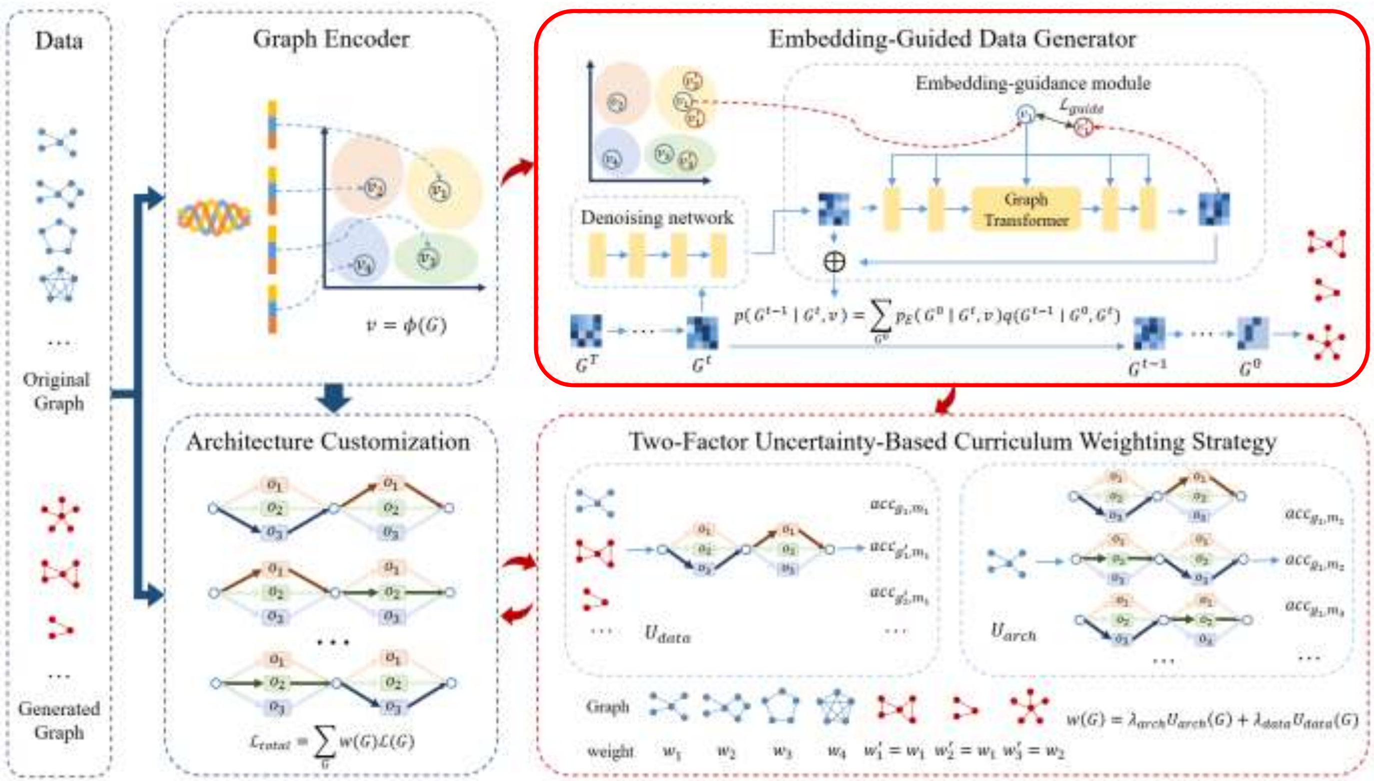


Data-Augmented Curriculum Graph Neural Architecture Search under Distribution Shifts. ***AAAI, 2024***

DCGAS: Data-Augmented Curriculum GraphNAS

- Embedding-guided data generator: generating graphs with similar structures
  - Embedding guidance + discrete graph diffusion model

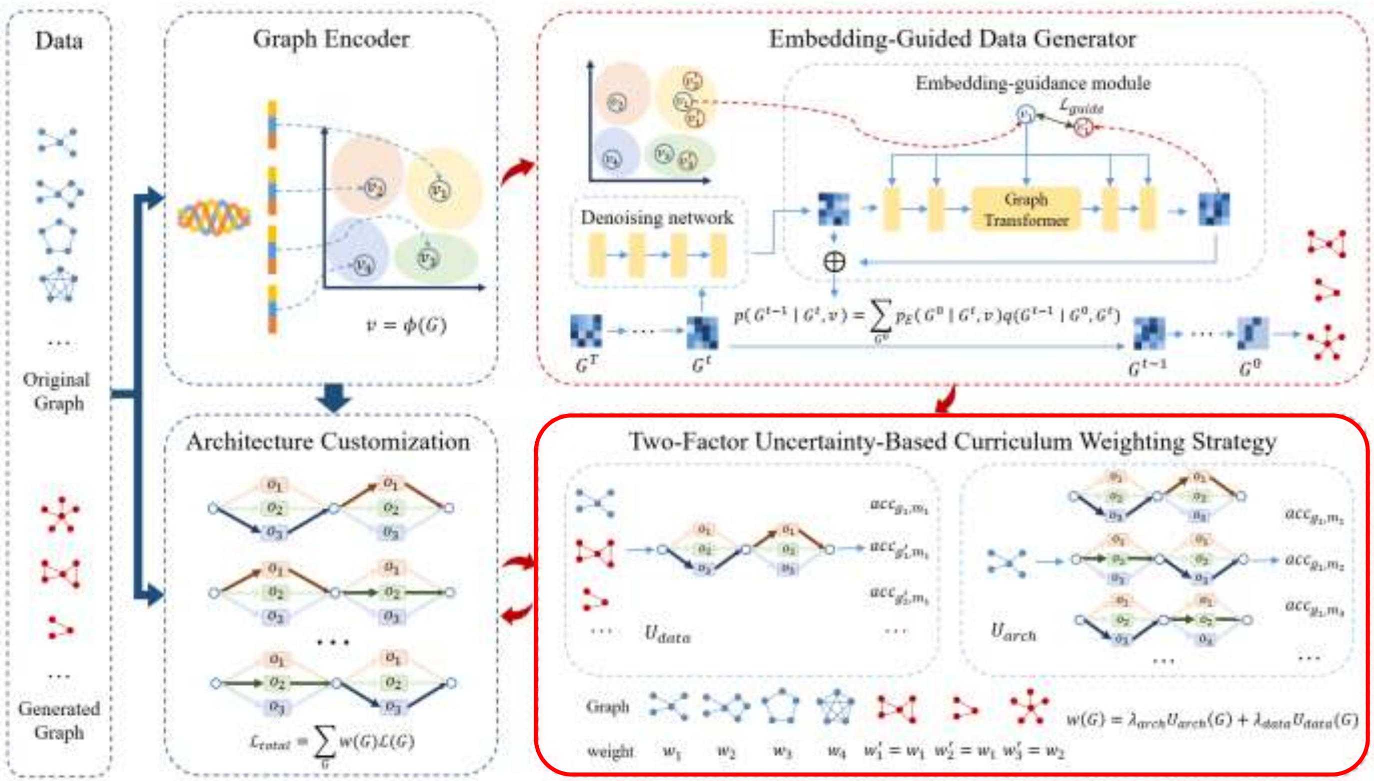
$$p(G^{t-1} \mid G^t) = \sum_{G^0} p_\theta(G^0 \mid G^t) q(G^{t-1} \mid G^0, G^t) \qquad p_{\text{guide}}(G^0 \mid G^t, v) \propto p_\theta(G^0 \mid G^t) p(v \mid G^0)$$



Data-Augmented Curriculum Graph Neural Architecture Search under Distribution Shifts. ***AAAI, 2024***

DCGAS: Data-Augmented Curriculum GraphNAS

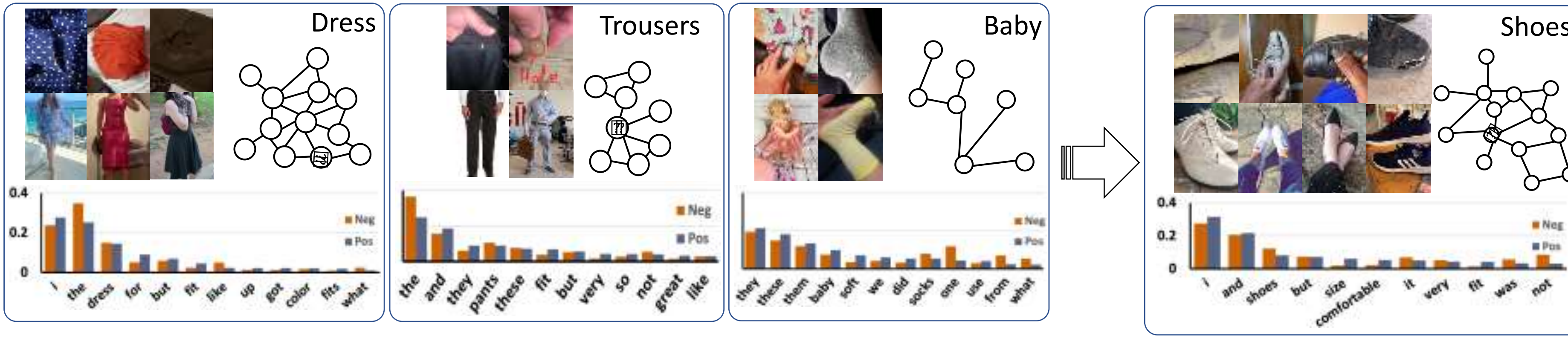
- Two-factor uncertainty-based curriculum weighting: schedule the training
  - Measures the uncertainty of the architecture performance on data
  - Higher uncertainties indicate higher weights $w(G) = \frac{\lambda_{\text{arch}} U_{\text{arch}}(G) + \lambda_{\text{data}} U_{\text{data}}(G)}{\sum_{G \in \mathcal{G}} \left(\lambda_{\text{arch}} U_{\text{arch}}(G) + \lambda_{\text{data}} U_{\text{data}}(G) \right)}$



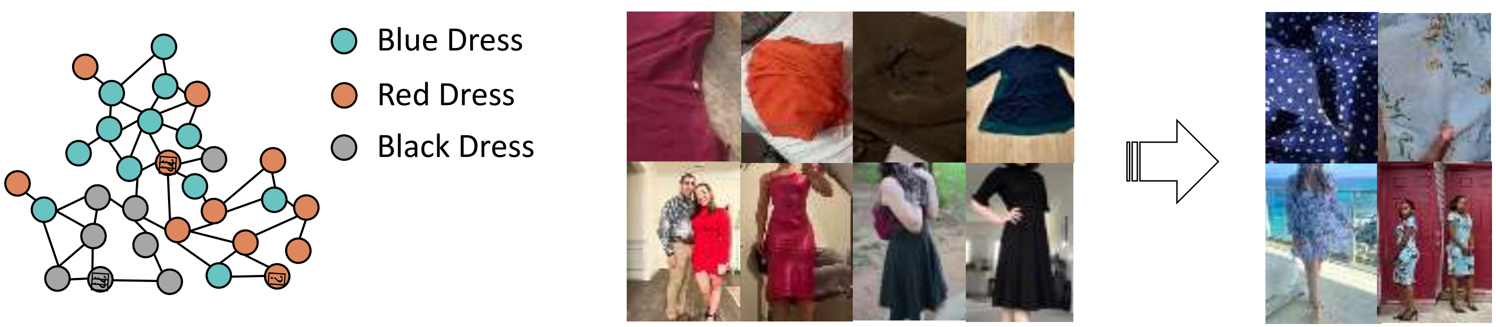
Data-Augmented Curriculum Graph Neural Architecture Search under Distribution Shifts. ***AAAI, 2024***

OMGNAS: Multimodal GraphNAS

- OOD Problem of multimodal graph



Distribution Shift on Multimodal Domain

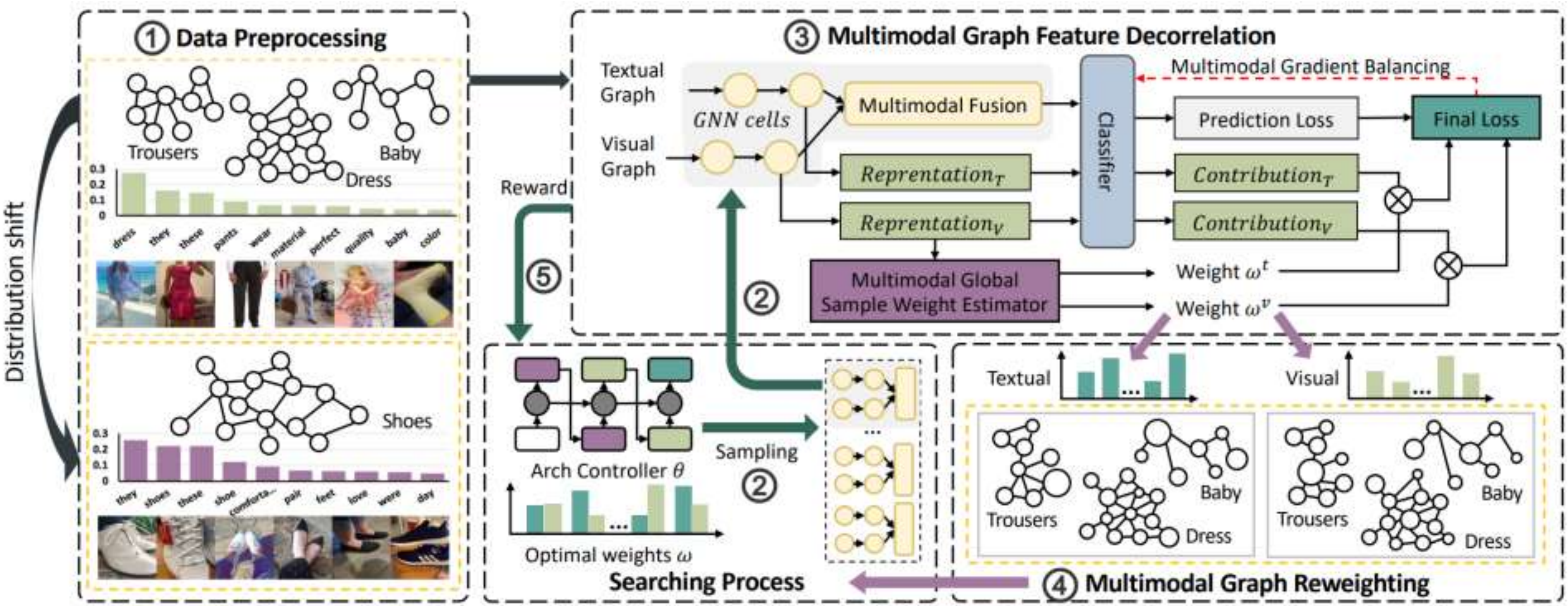


Distribution Shift on Singlemodal Domain (Color Shift)

Multimodal Graph Neural Architecture Search Under Distribution Shifts. ***AAAI, 2024.***

OMGNAS: Multimodal GraphNAS

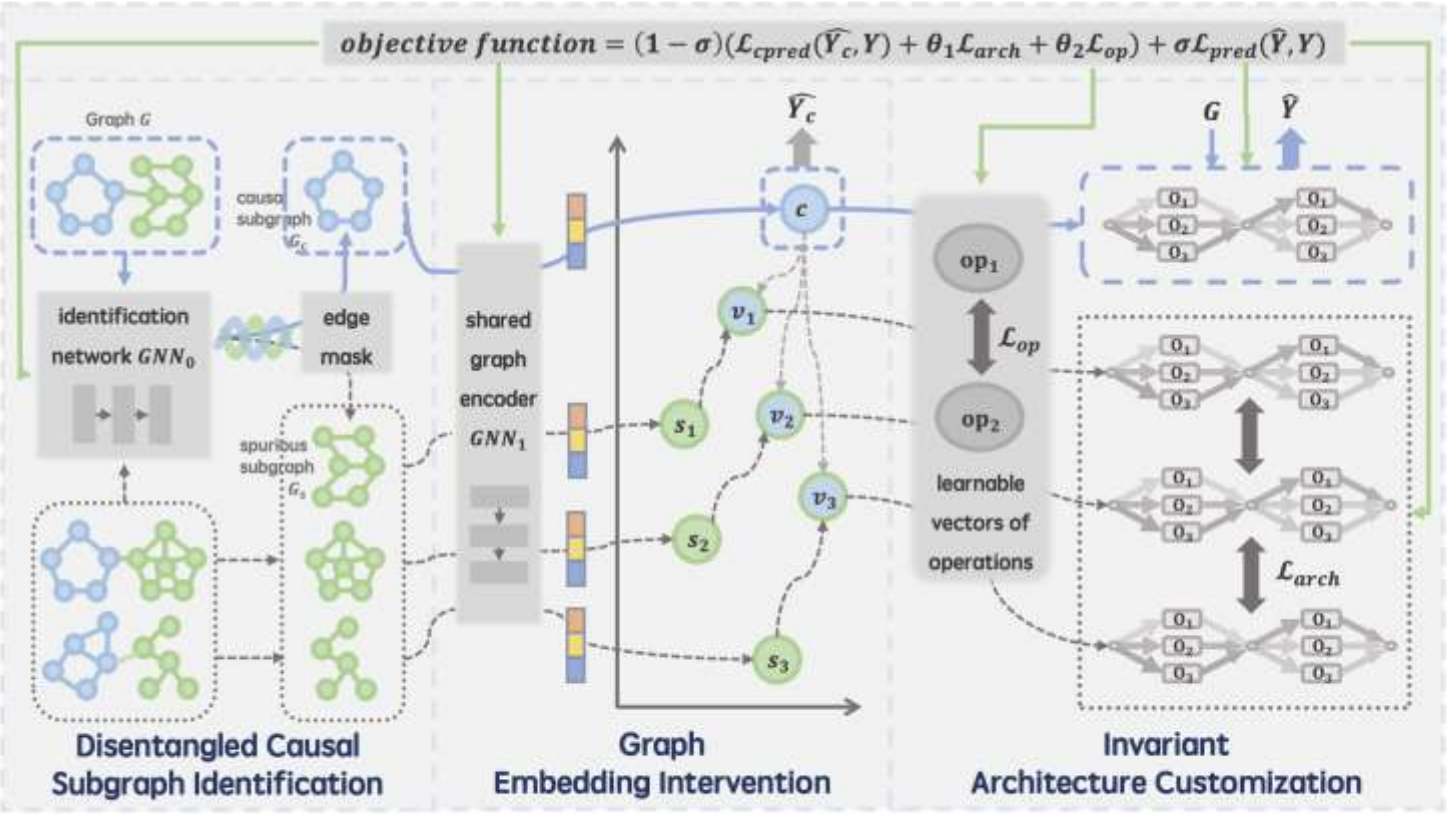
- Main idea: decorrelate multimodal graph features, then customize



Multimodal Graph Neural Architecture Search Under Distribution Shifts. ***AAAI, 2024.***

CARNAS: Causal-aware GraphNAS

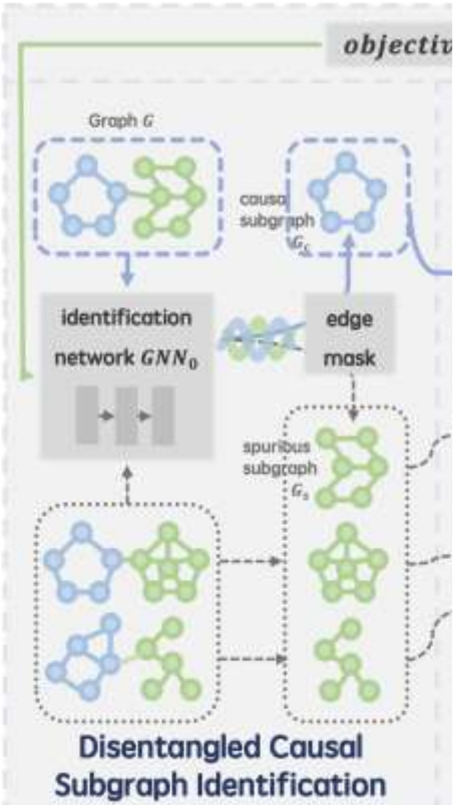
- Combine customization idea with invariance principle



Causal-aware Graph Neural Architecture Search under Distribution Shifts. ***KDD, 2025.***

CARNAS: Causal-aware GraphNAS

- Disentangled Causal Subgraph Identification



- Goal: capture different latent factors and split the input graph instance causal/non-causal subgraph
  - Learnable disentangled GNN layers

$$\mathbf{Z}^{(l)} = \|_{q=1}^{Q} \operatorname{GNN}_0\left(\mathbf{Z}_q^{(l-1)}, \mathbf{D}\right)$$

  - Edge importance mask

$$\mathcal{S}_{\mathcal{E}} = \operatorname{MLP}\left(\mathbf{Z}_{row}^{(L)}, \mathbf{Z}_{col}^{(L)}\right)$$

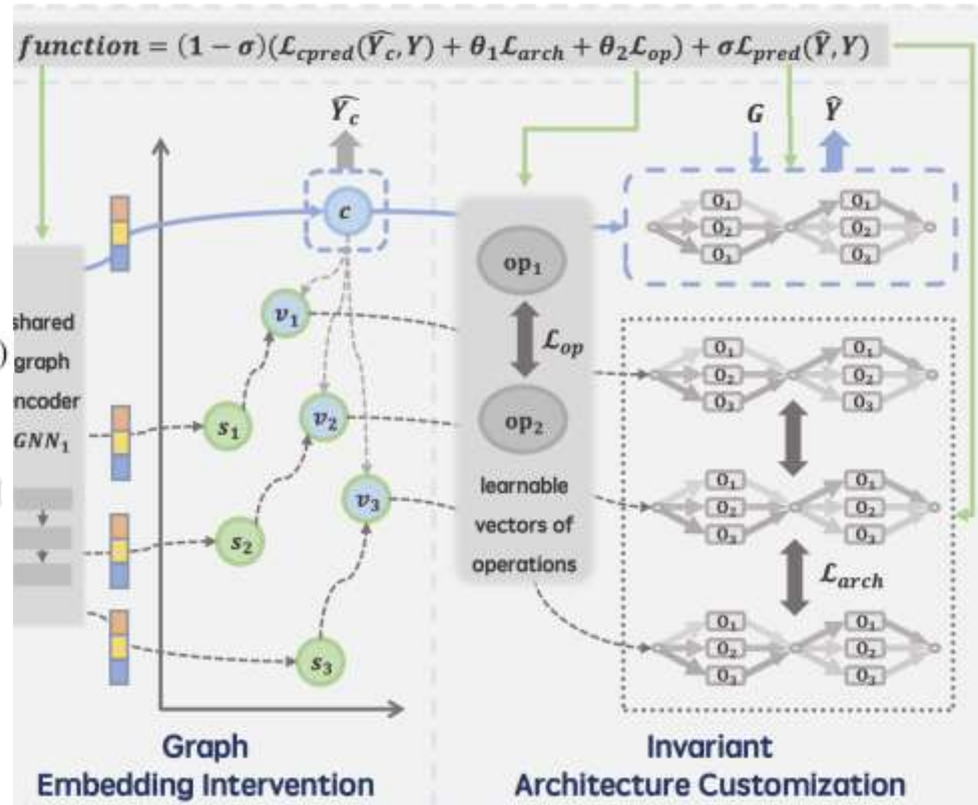
  - Causal/non-causal subgraphs

$$\mathcal{E}_c = \operatorname{Top}_t(\mathcal{S}_{\mathcal{E}}), \ \mathcal{E}_s = \mathcal{E} - \mathcal{E}_c$$

Causal-aware Graph Neural Architecture Search under Distribution Shifts. ***KDD, 2025.***

CARNAS: Causal-aware GraphNAS

- Graph Embedding Intervention



- Goal: do interventions in the latent space
  - Learn representation

$$Z_c = \text{GNN}_1(G_c),\ Z_s = \text{GNN}_1(G_s)$$
$$H_c = \text{READOUT}(Z_c),\ H_s = \text{READOUT}(Z_s)$$

  - Intervention

$$do(S = G_{s\,j}):\ H_{v\,j} = (1-\mu)\cdot H_c + \mu \cdot H_{s\,j},\ j \in [1, N_s]$$

- Architecture customization

$$\alpha_u^k = \frac{\exp\left(\text{op}_u^{k\,T} H\right)}{\sum_{u'=1}^{|O|} \exp\left(\text{op}_{u'}^{k\,T} H\right)}$$

- Joint Optimization

$$\mathcal{L}_{all} = \sigma \mathcal{L}_{pred} + (1-\sigma)\mathcal{L}_{causal}$$
$$\mathcal{L}_{causal} = \mathcal{L}_{cpred} + \theta_1 \mathcal{L}_{arch} + \theta_2 \mathcal{L}_{op}$$

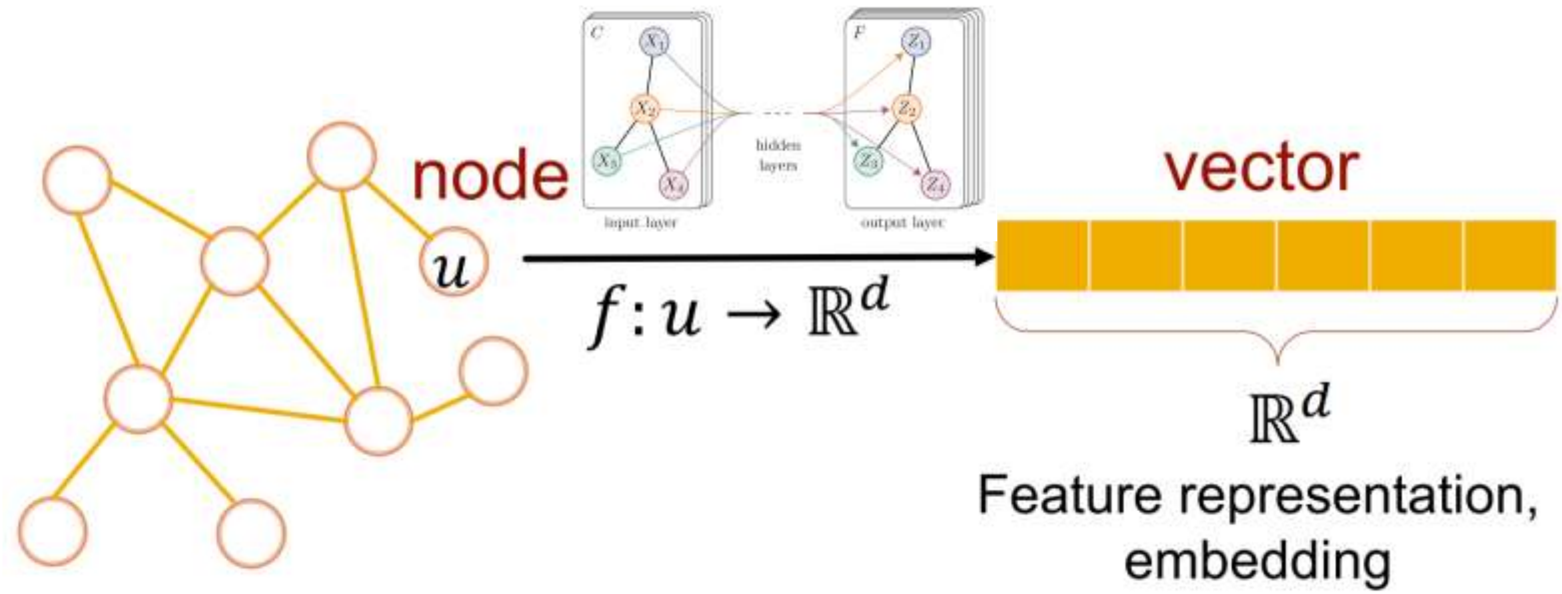
Causal-aware Graph Neural Architecture Search under Distribution Shifts. ***KDD, 2025.***

Recap: Graph Invariant Learning in the Architecture Space

- GRACES (ICML’22): customize architectures
- OMGNAS (AAAI’24): data augmentation + curriculum
- DCGAS (AAAI’24): multimodal decorrelation
- CARNAS (KDD’25): capturing invariant parts

Finding Invariance

How to find invariance in GNN **architectures**?



How to find invariance in the **topology space**?

How to find invariance in the **vector space**?

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

Graph Invariant Learning in the Topology Space

Static graphs:

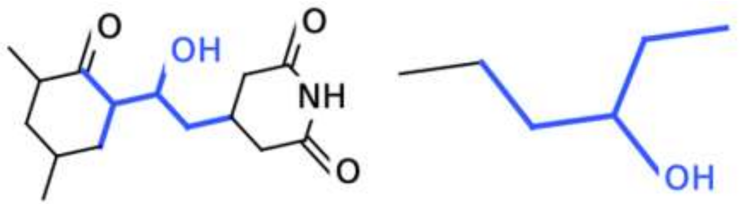
- GIL (NeurIPS'22)
- DIR (ICLR'22)
- NIL (ACM TOIS'23)
- GOODFormer (arXiv'25)

Dynamic graphs:

- DIDA (NeurIPS'22)
- SILD (NeurIPS'23)
- EAGLE (NeurIPS'23)

Invariance-guided Graph Learning

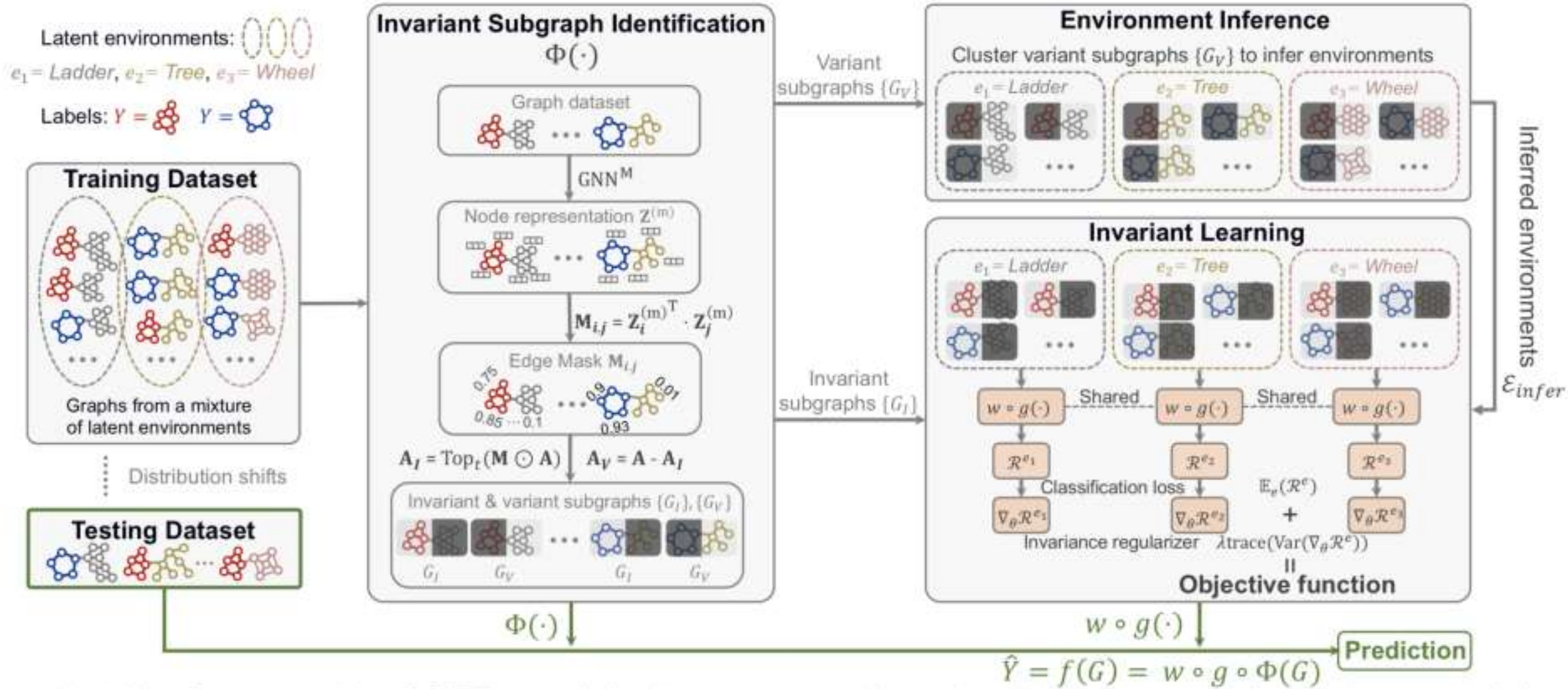
- Previous methods implicitly learn invariant/variant graphs
- Can we explicitly distinguish invariant and variant subgraphs?



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

GIL: Method

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

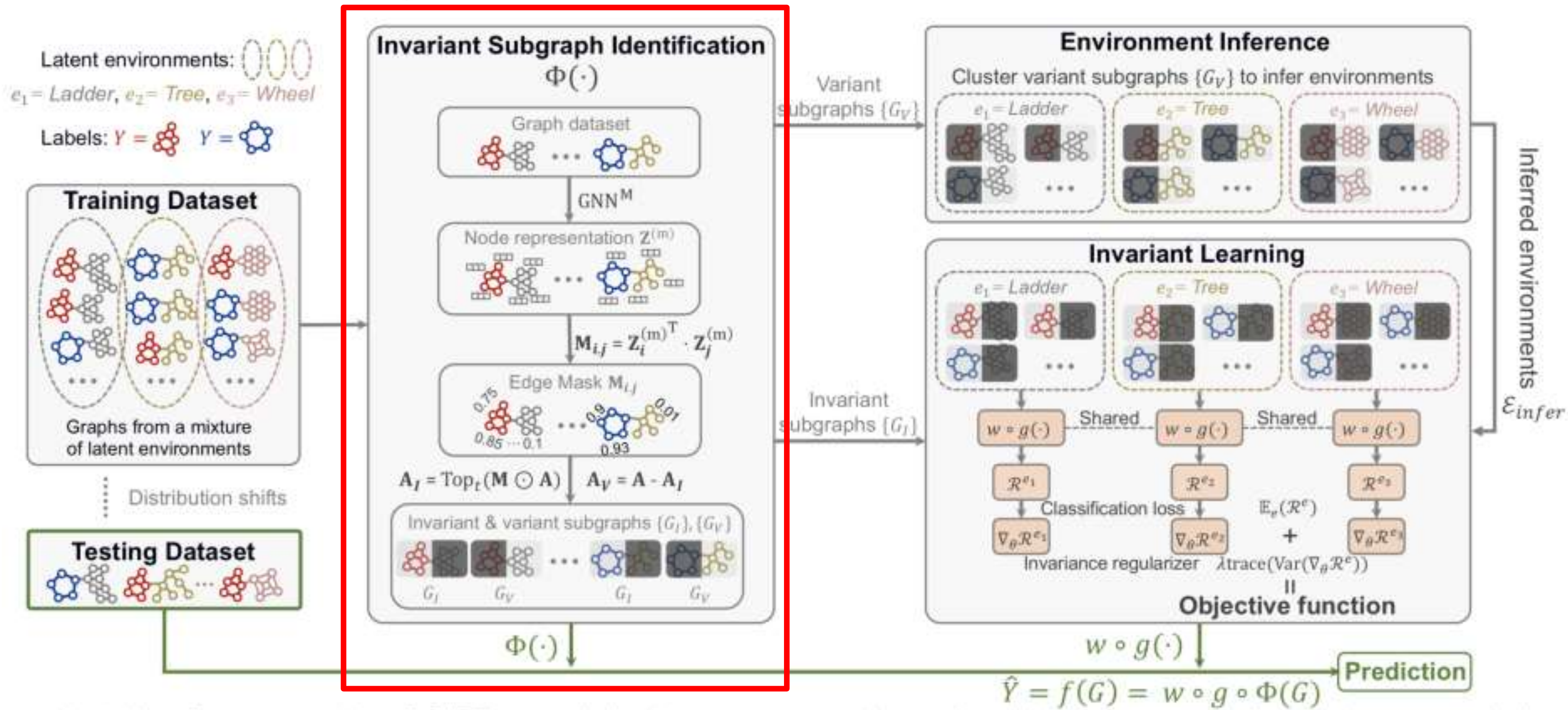


Learning Invariant Graph Representations under Distribution Shifts. ***NeurIPS, 2022.***

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

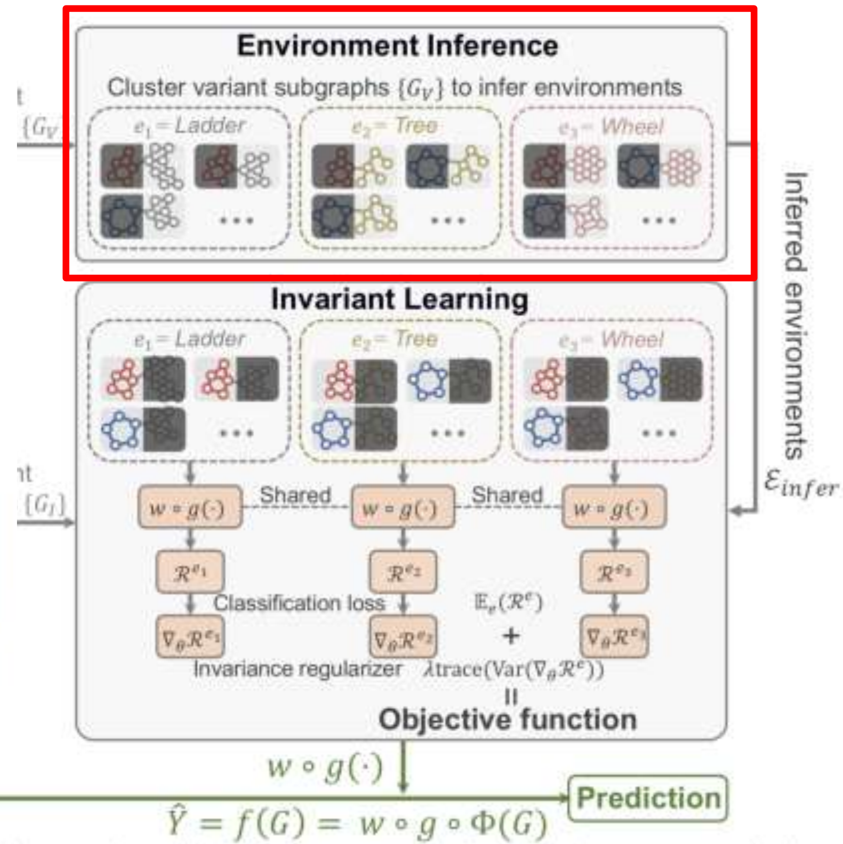
$$\mathbf{Z}^{(m)} = \mathrm{GNN}^{\mathrm{M}}(G) \quad \mathbf{M}_{i,j} = \mathbf{Z}_i^{(m)^\top} \cdot \mathbf{Z}_j^{(m)} \quad \mathbf{A}_I = \mathrm{Top}_t\left(\mathbf{M} \odot \mathbf{A}\right), \mathbf{A}_V = \mathbf{A} - \mathbf{A}_I$$



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

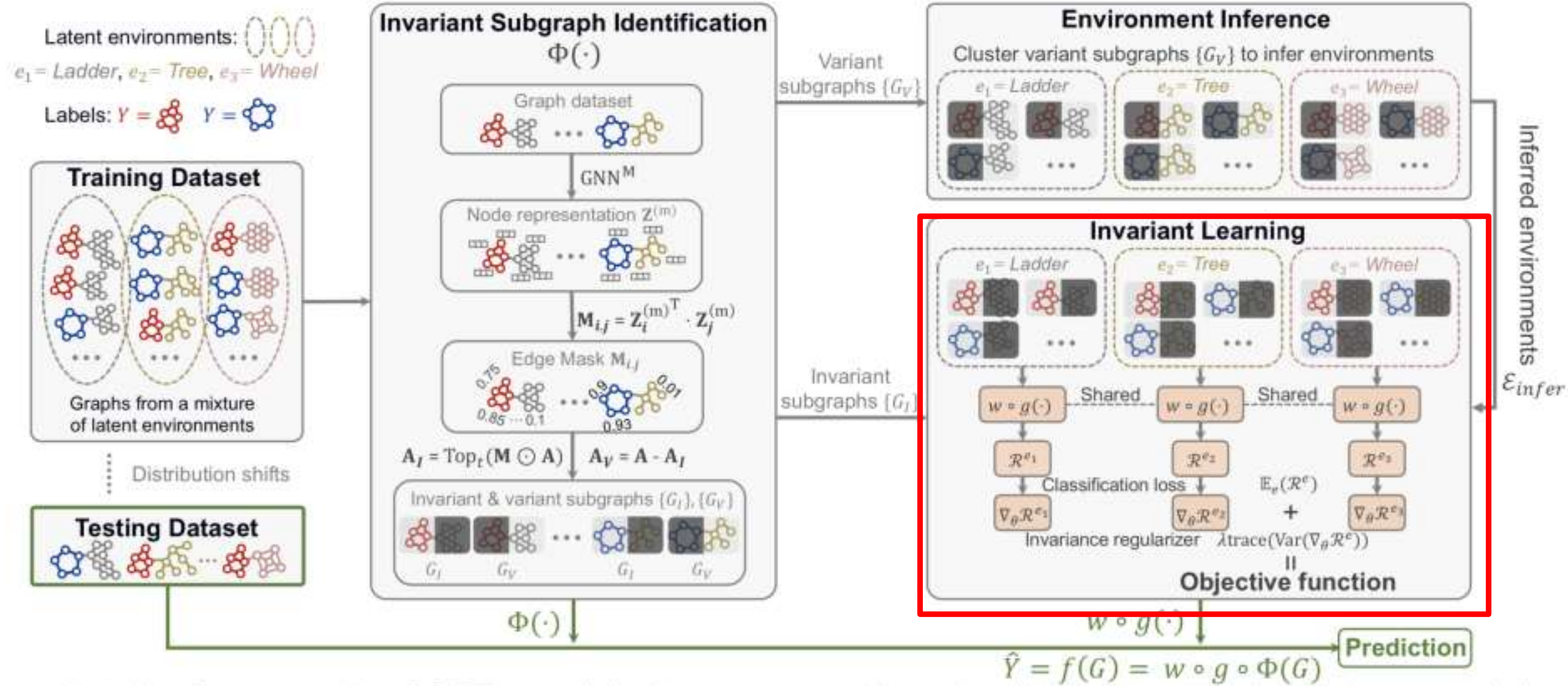
- Goal: find an invariant subgraph generator $\mathcal{I}_{\mathcal{E}} = \{\Phi(\cdot) : P^e(\mathrm{Y}|\Phi(\mathrm{G})) = P^{e'}(\mathrm{Y}|\Phi(\mathrm{G})), e, e' \in \mathrm{supp}(\mathcal{E})\}$
- Optimization:

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



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:*

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

  - Several assumptions:

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

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

- 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 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

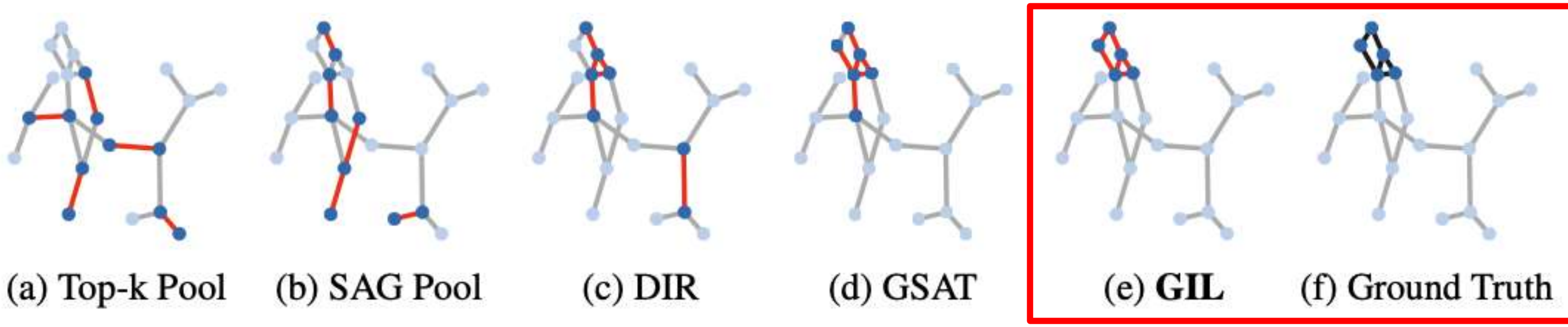
CIN (Rank #8)	**GIL (CIN Backbone)**	HIG (Rank #2)	PAS+FPs (Rank #1)	**GIL (HIG Backbone)**
80.94±0.57	**81.15**±0.46	84.03±0.21	84.20±0.15	**84.23**±0.25

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

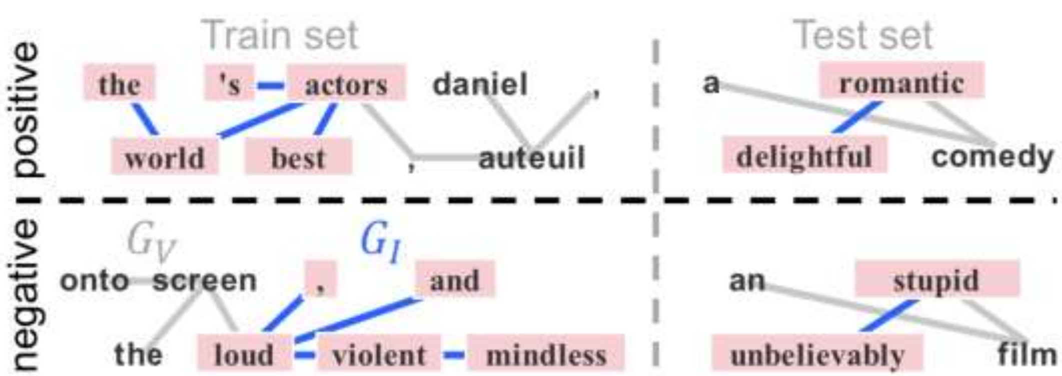
Learning Invariant Graph Representations under Distribution Shifts. ***NeurIPS, 2022.***

GIL: Experiments

- Showcase on Spurious-Motif datasets



- Showcases on Graph-SST2 (human-understandable)

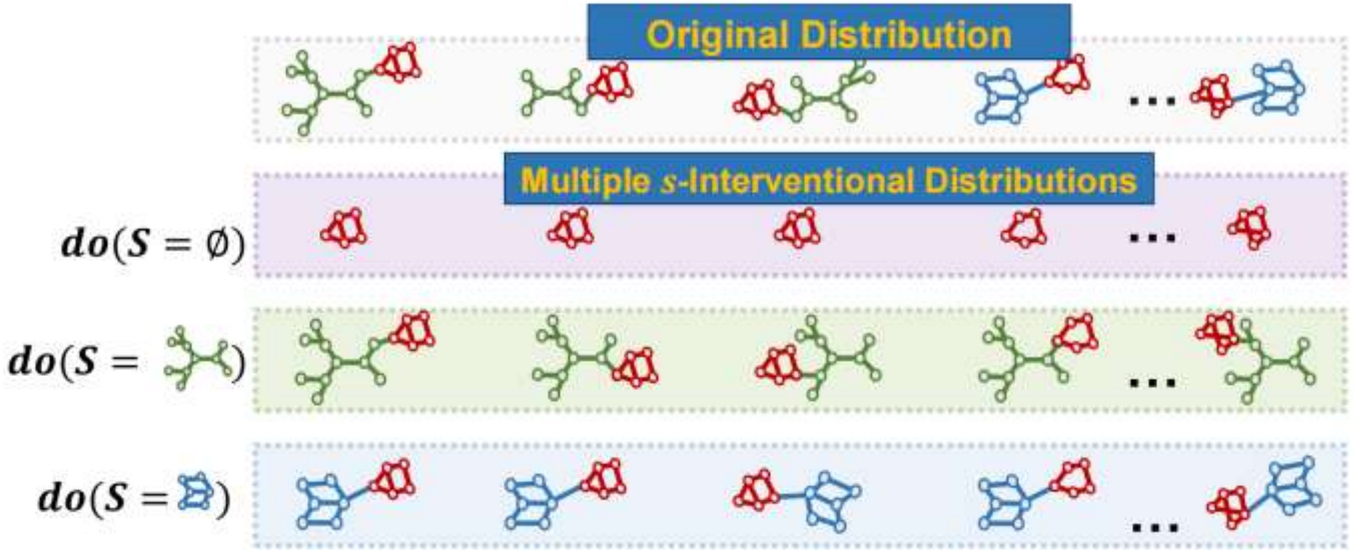


Capture the subgraphs with positive/negative semantics

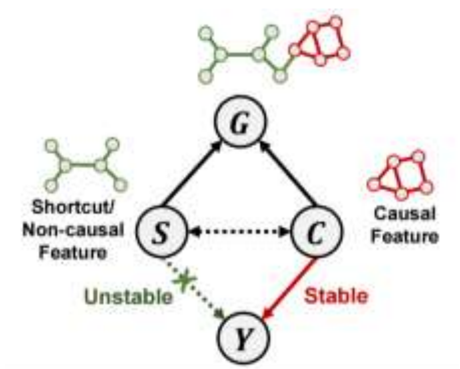
Learning Invariant Graph Representations under Distribution Shifts. ***NeurIPS, 2022.***

Discovering Invariant Rationale

- Main idea: minimize interventional risks to find invariant (causal) features



- Formally, from a causal perspective



$$Y = f_Y(C), \quad Y \perp\!\!\!\perp S \mid C$$

Input graph G

Causal part C

Non-causal part S

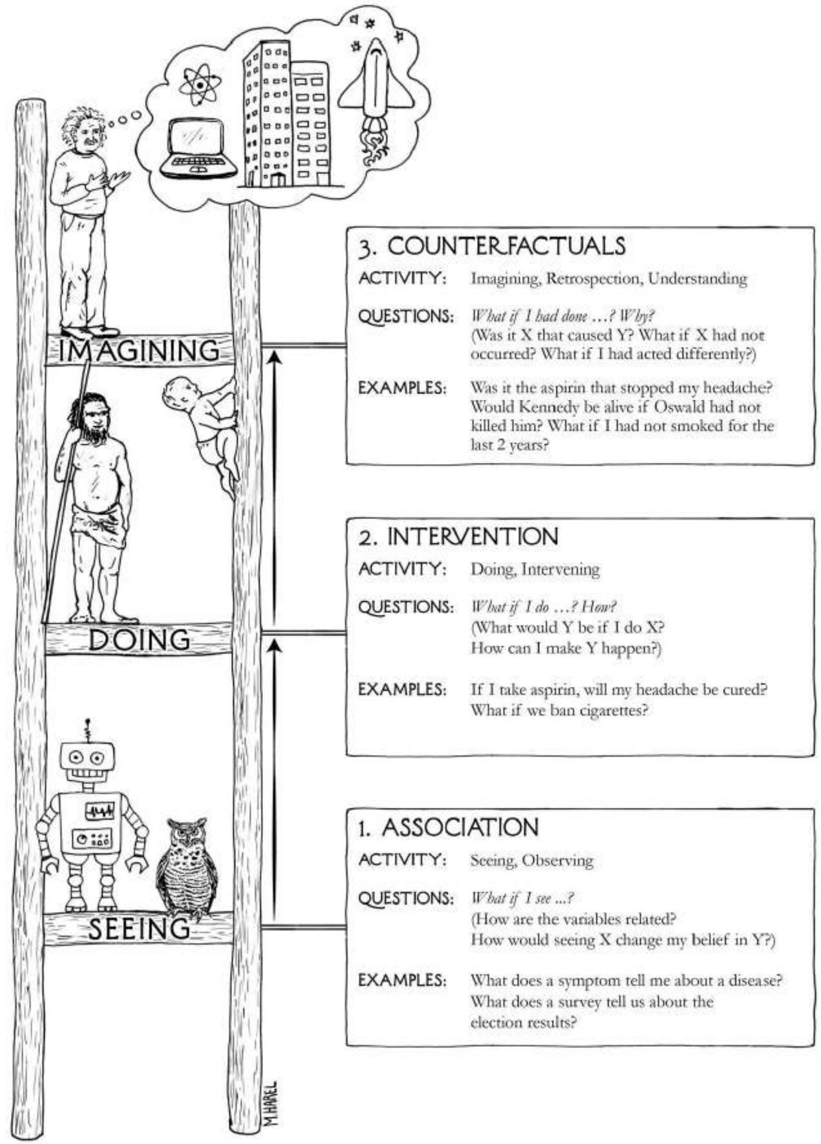
Ground-truth label Y

Discovering Invariant Rationales for Graph Neural Networks. ***ICLR, 2022.***

Discovering Invariant Rationale

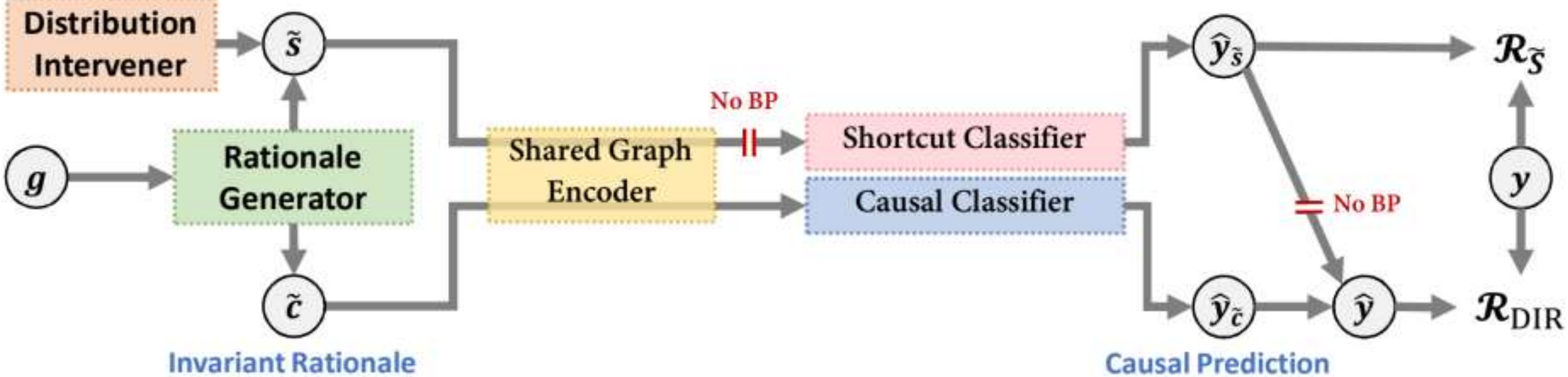
- The causality ladder

Judea Pearl
2011 Turing Award



Discovering Invariant Rationale

- Main idea: minimize interventional risks to find invariant (causal) features



- Rationale generator: split the input graph into causal/non-causal subgraphs

$$\mathbf{Z} = \mathrm{GNN}_1(g), \quad \mathbf{M}_{ij} = \sigma(\mathbf{Z}_i^\top \mathbf{Z}_j) \quad \mathcal{E}_{\tilde{c}} = \mathrm{Top}_r(\mathbf{M} \odot \mathbf{A}), \quad \mathcal{E}_{\tilde{s}} = \mathrm{Top}_{1-r}((1-\mathbf{M}) \odot \mathbf{A})$$

- Distribution Intervener: create interventional distributions by randomly replacing the complement of the causal subgraph
- Optimization: minimize variance of interventions

$$\min \mathcal{R}_{\mathrm{DIR}} = \mathbb{E}_s[\mathcal{R}(h(G), Y | do(S=s))] + \lambda \mathrm{Var}_s(\{\mathcal{R}(h(G), Y | do(S=s))\})$$

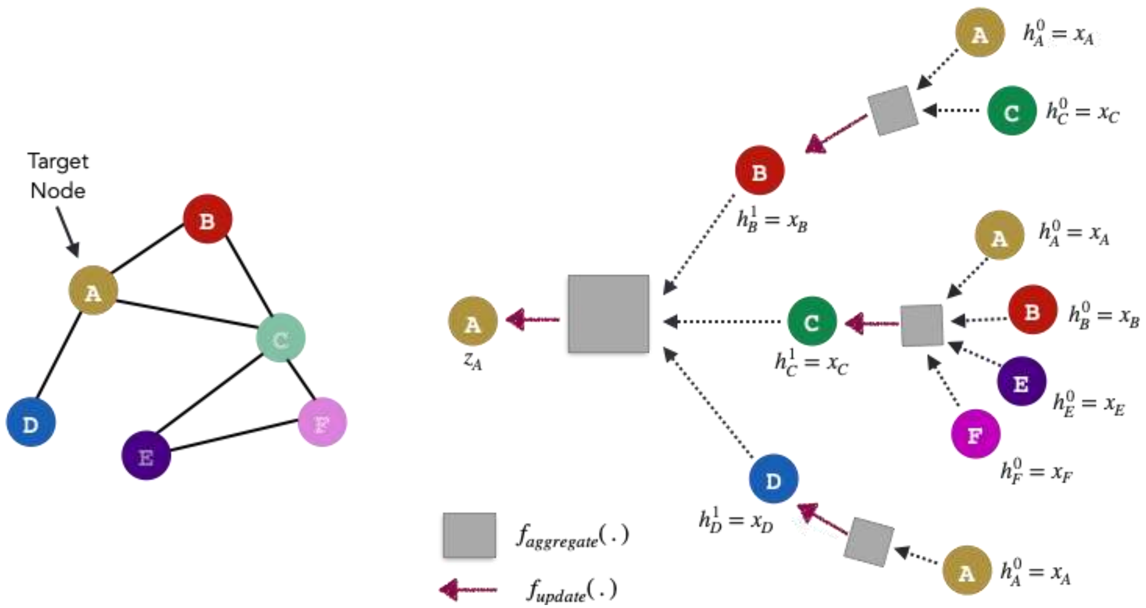
Discovering Invariant Rationales for Graph Neural Networks. ***ICLR, 2022.***

Node Invariant Learning (NIL)

- For graph-level tasks, different graphs can be considered samples

→ how to generalize to node/link-level tasks?

- Basic idea: the receptive field of each node is an ego-subgraph



- Treat node classification as ego-subgraph classifications
- Note: these subgraphs are dependent, but can be assumed to satisfy

conditional independence, i.e., $P(\mathrm{Y}|\mathrm{G}) = \prod_{\mathrm{v}} P(\mathrm{y}|\mathrm{G}_{\mathrm{v}})$

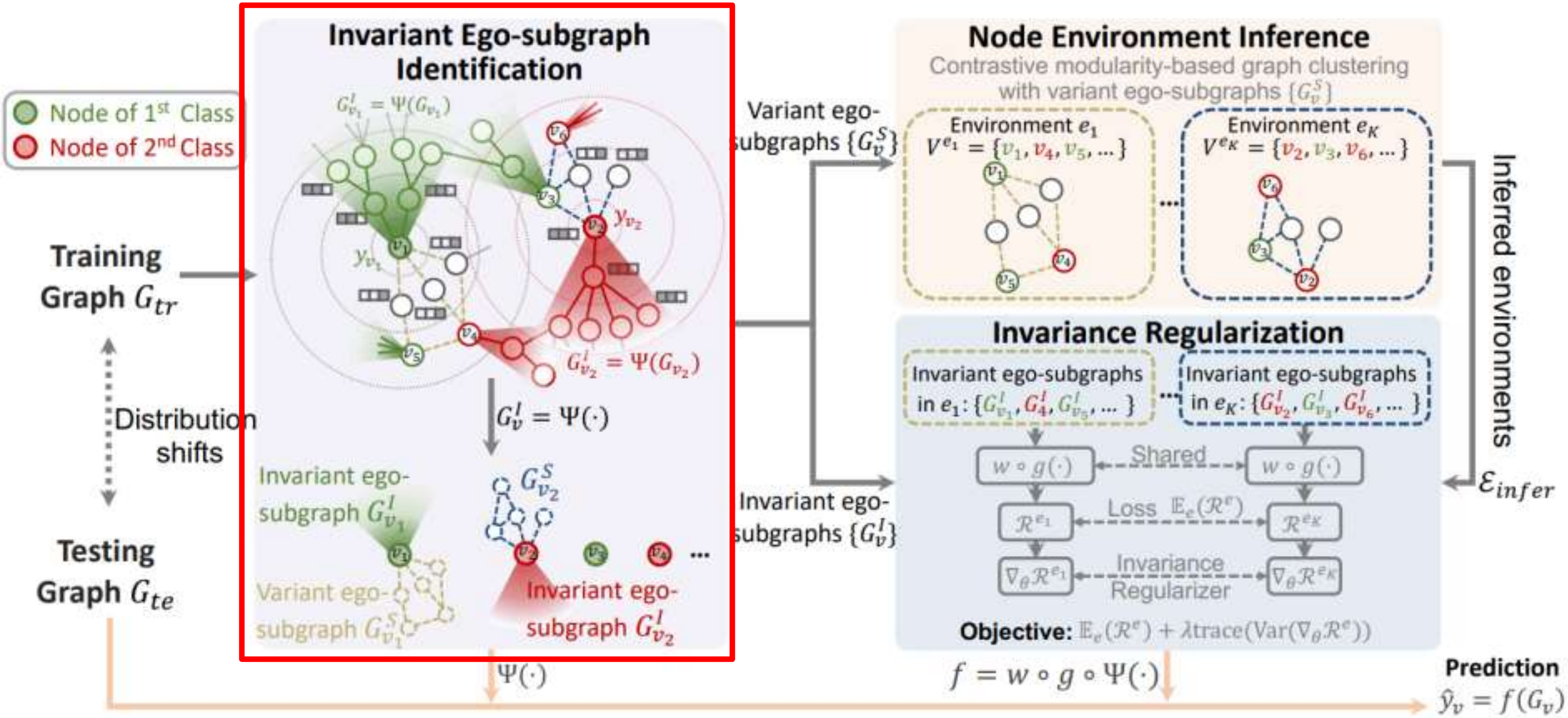
Invariant Node Representation Learning under Distribution Shifts with Multiple Latent Environments. ***ACM TOIS, 2023.***

NIL: Framework

- Goal: mask invariant/variant graph structures and node features

$$M^{A_v}_{u,u'} = \mathrm{Sigmoid}\left(\mathrm{Z}^{\mathrm{M}^\top}_u \cdot \mathrm{Z}^{\mathrm{M}}_{u'}\right), \quad \mathrm{Z}^{\mathrm{M}} = \mathrm{GNN}^{\mathrm{M}}(G_v) \in \mathbb{R}^d.$$

$$A^I_v = M^{A_v} \odot A_v, X^I_v = M^X \odot X_v; \;\; A^S_v = A_v - A^I_v, X^S_v = X_v - X^I_v$$



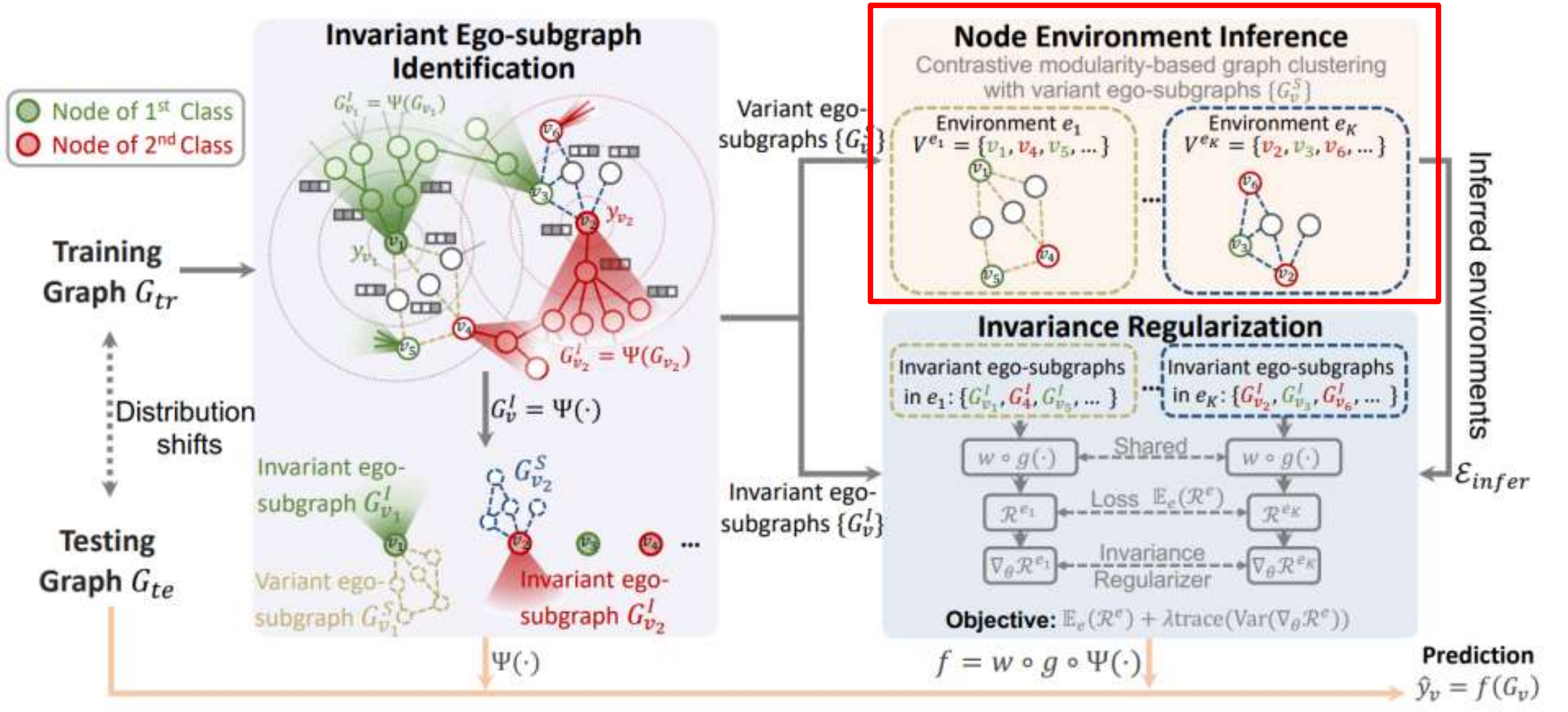
Invariant Node Representation Learning under Distribution Shifts with Multiple Latent Environments. ***ACM TOIS, 2023.***

NIL: Framework

- Goal: infer environments
  - Contrastive modularity-based clustering
  - Based on homophily assumption

$$\min_{C} \ell = -\frac{1}{K}\sum_{k=1}^{K}\log\frac{\exp(B_{k,k})}{\sum_{k'=1,k'\neq k}^{K}\exp(B_{k,k'})},$$

$$B = \frac{1}{2m}\left(C^{\top}A^{S}C - \frac{1}{2m}\operatorname{diag}\left(C^{\top}\mathbf{d}\mathbf{d}^{\top}C\right)\right).$$

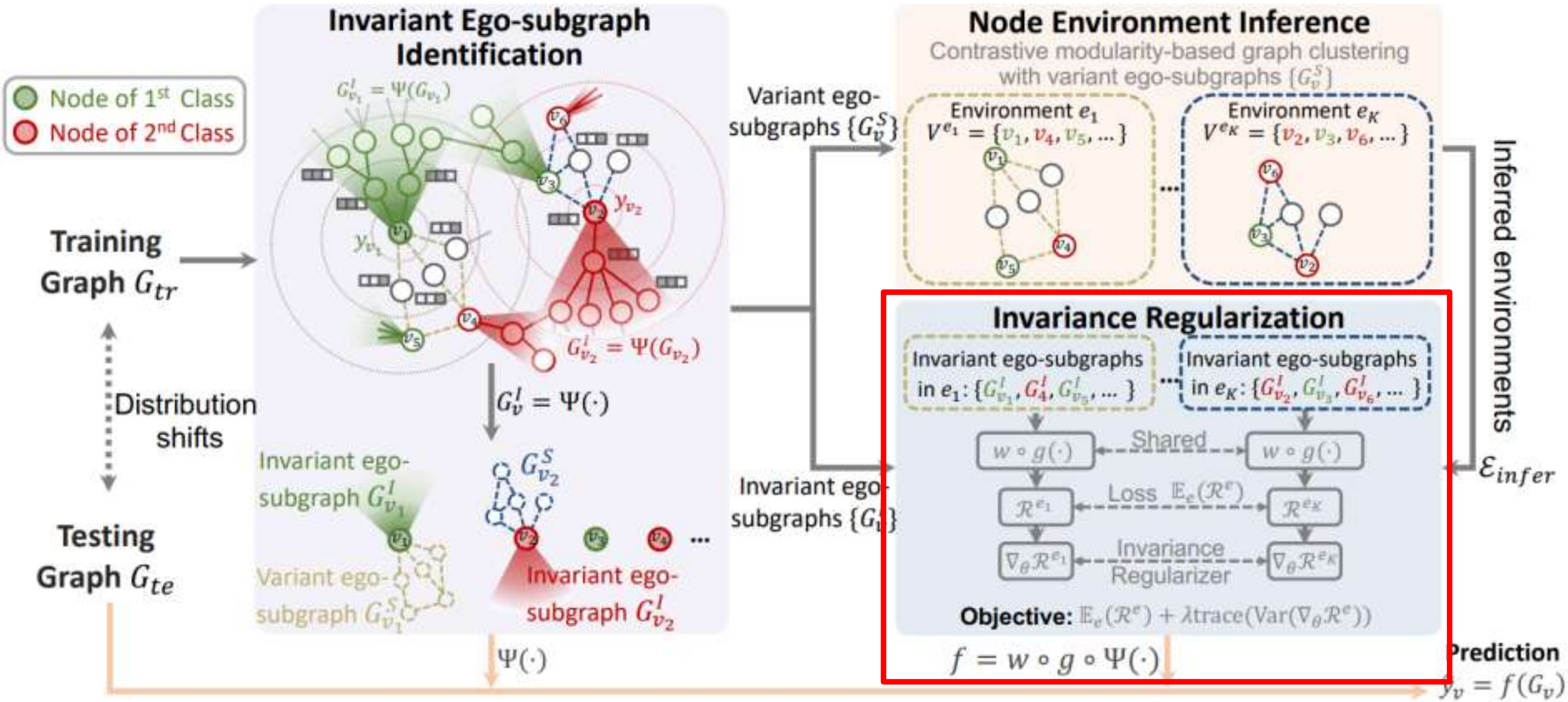


Invariant Node Representation Learning under Distribution Shifts with Multiple Latent Environments. ***ACM TOIS, 2023.***

NIL: Framework

- Goal: encourage to only use invariant parts for prediction

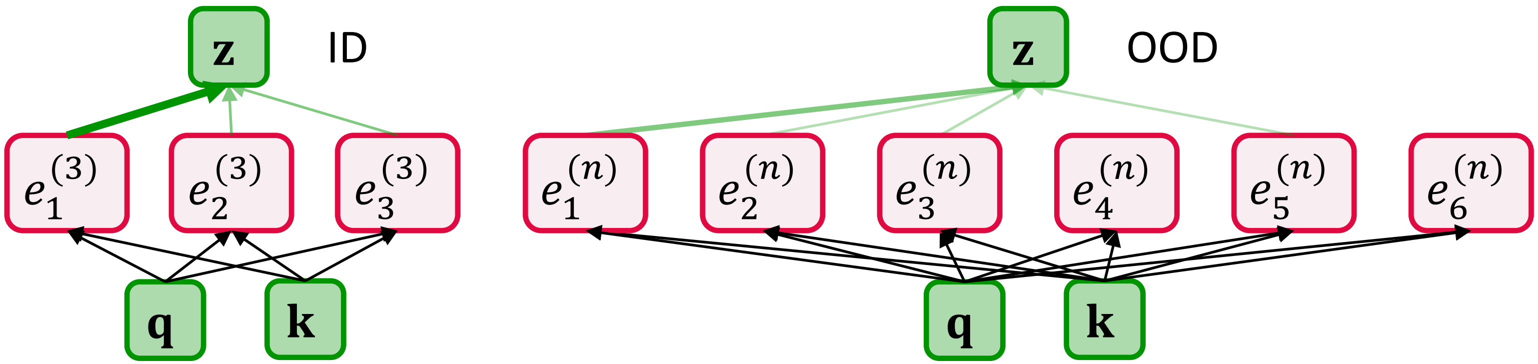
$$\mathbb{E}_{e\in \text{supp}(\mathcal{E}_{infer})}\mathcal{R}^e\left(f\left(\mathrm{G_v}\right),\mathrm{y};\theta\right)+\lambda\text{trace}\left(\text{Var}_{\mathcal{E}_{infer}}\left(\nabla_\theta\mathcal{R}^e\right)\right)$$



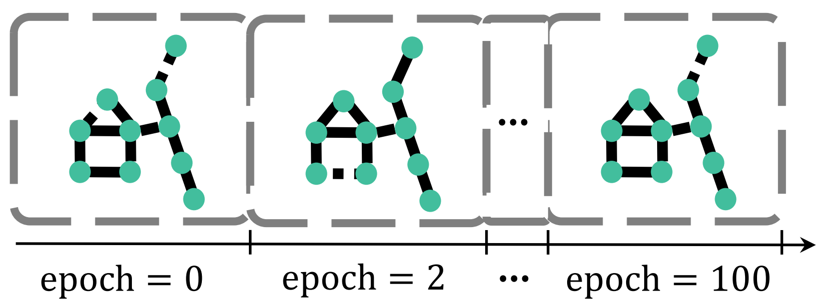
Invariant Node Representation Learning under Distribution Shifts with Multiple Latent Environments. ***ACM TOIS, 2023.***

Graph OOD Transformer (GOODFormer)

- How to tackle the OOD generalization problem of graph Transformers?
- Challenges:
  - The classical self-attention cannot guarantee sharpness.



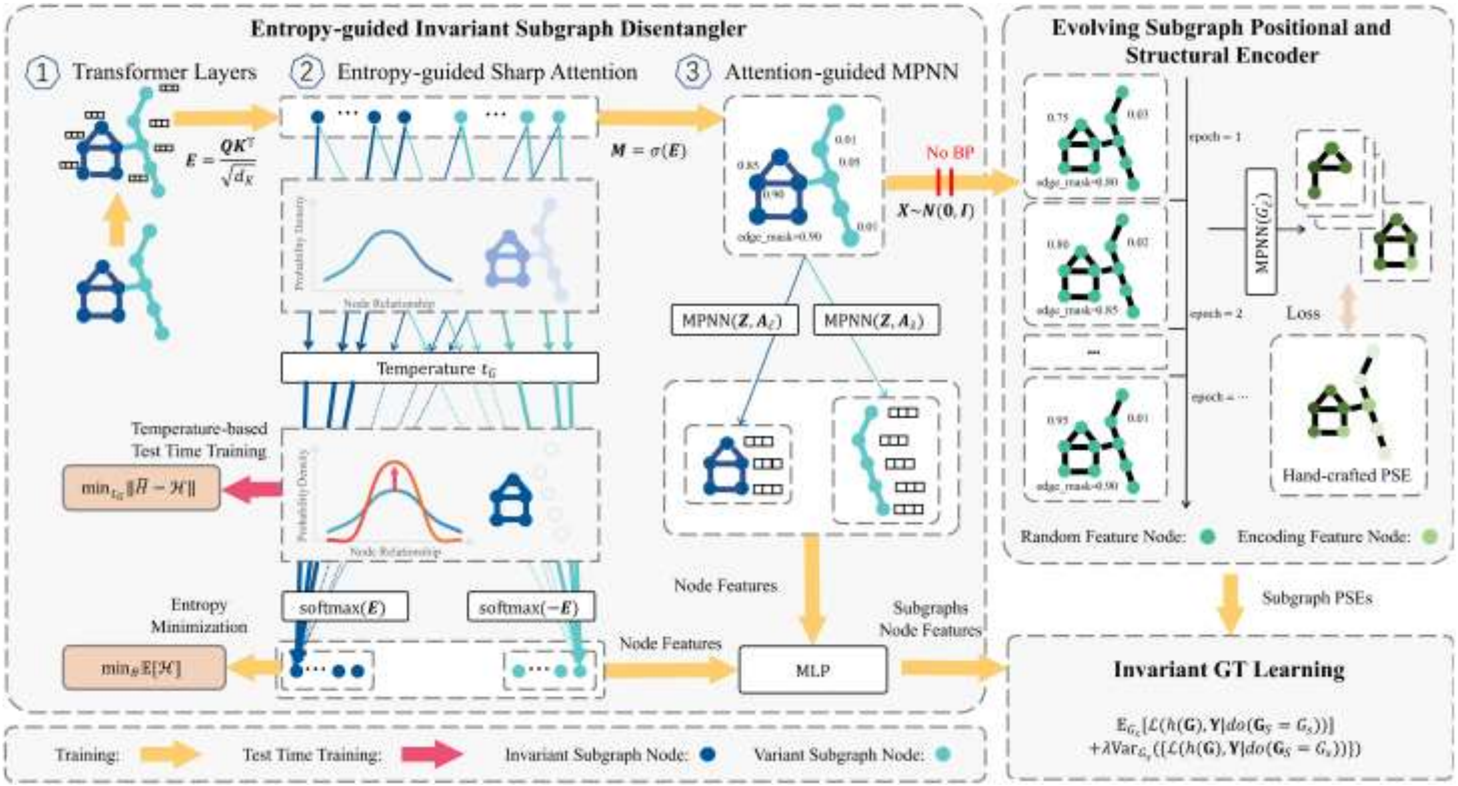
  - The dynamically evolving invariant subgraph leads to prohibitive computational complexity for positional and structural encoding.



Invariant Graph Transformer for Out-of-Distribution Generalization. ***arXiv, 2025.***

GOODFormer: Method

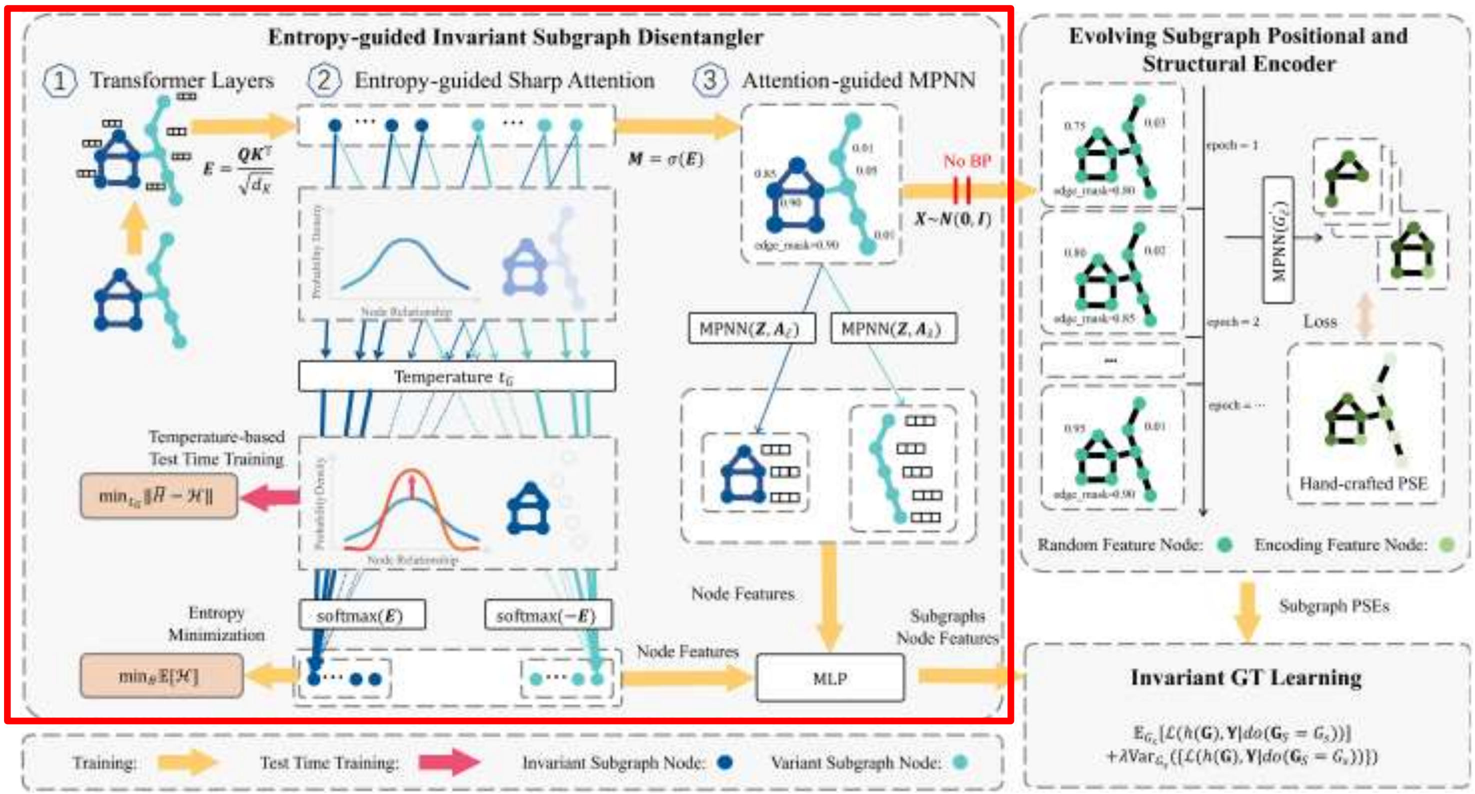
- **Key Idea**: design attention mechanisms and postional and structural encodings based on graph invariant learning principles
  - Attention: how to guarantee sharpness
  - Positional and structural encoding: how to maintain efficiency and expressiveness



Invariant Graph Transformer for Out-of-Distribution Generalization. ***arXiv, 2025.***

GOODFormer: Method

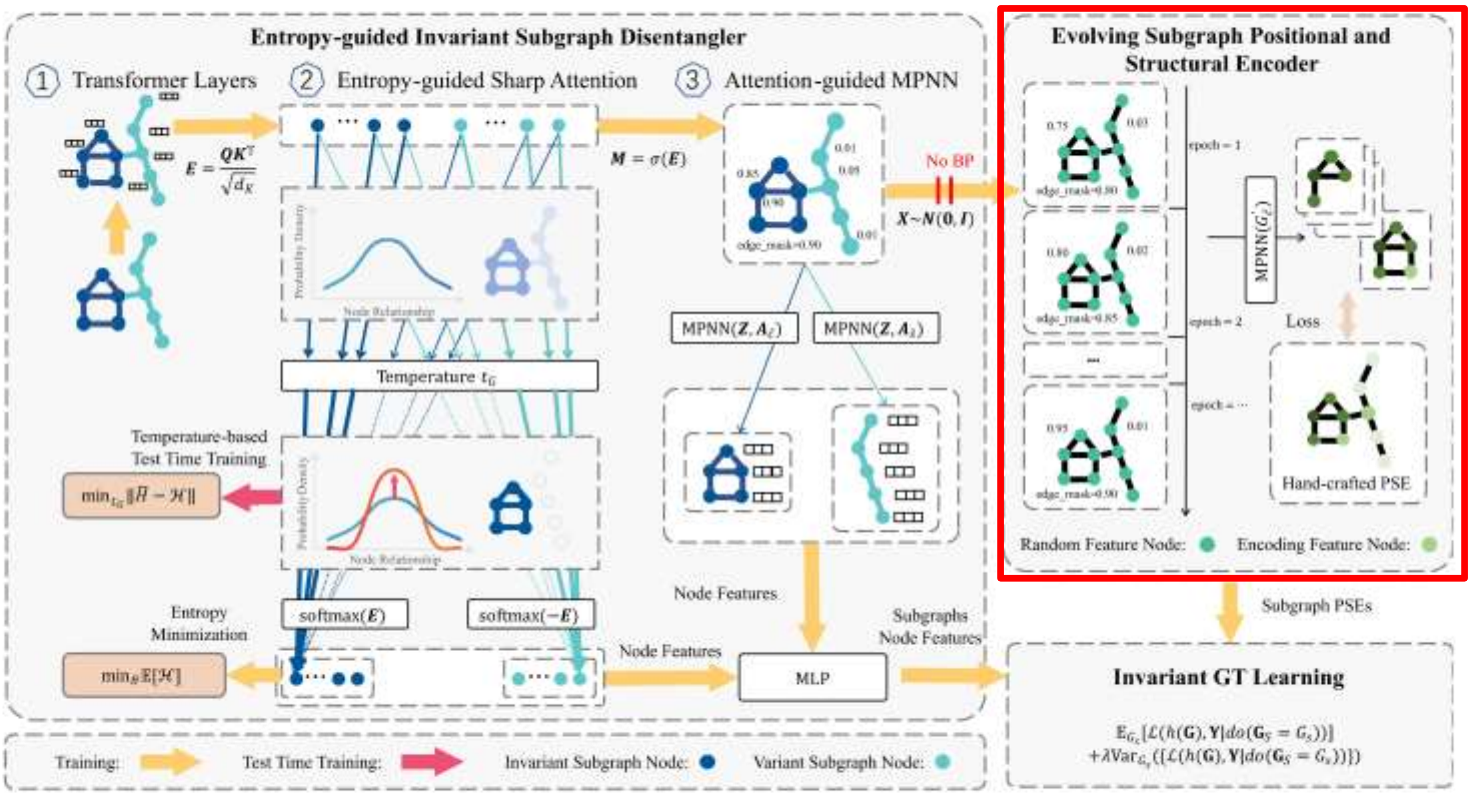
- **Goal**: separate invariant and variant subgraphs for graph Transformer
- Proposed method:
  - Two attentions for invariant/variant subgraphs $Z_{\mathrm{Attn},\tilde{c}} = \mathrm{Softmax}(\boldsymbol{E})\boldsymbol{Z}\boldsymbol{W}_V, \boldsymbol{Z}_{\mathrm{Attn},\tilde{s}} = \mathrm{Softmax}(-\boldsymbol{E})\boldsymbol{Z}\boldsymbol{W}_V$
  - Test-time training: minimize entropy difference between training and test data



Invariant Graph Transformer for Out-of-Distribution Generalization. ***arXiv, 2025.***

GOODFormer: Method

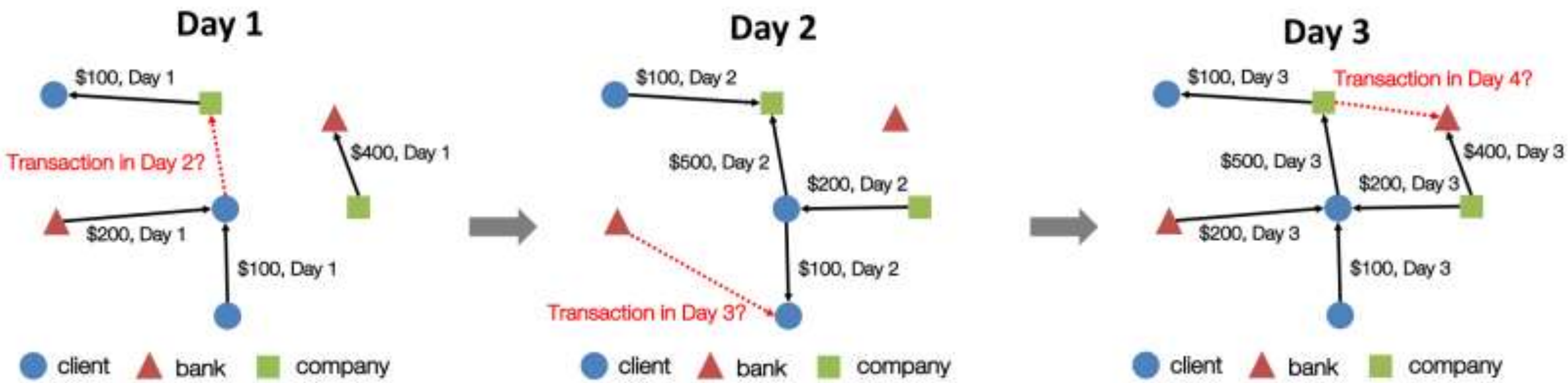
- **Goal**: Capture the positional/structural information of dynamically evolving subgraphs
  - Learnable MPNN-based encoder instead of hand-crafted encoding
  - Distillation loss to transfer knowledge from full-graph to subgraph encodings



Invariant Graph Transformer for Out-of-Distribution Generalization. ***arXiv, 2025.***

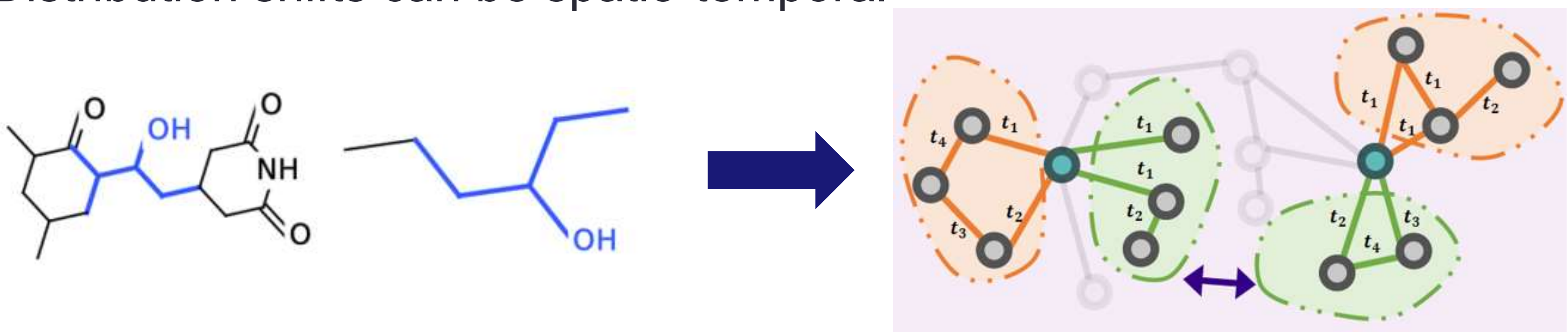
Invariant Learning for Dynamic Graphs

- Many graphs are dynamic in nature



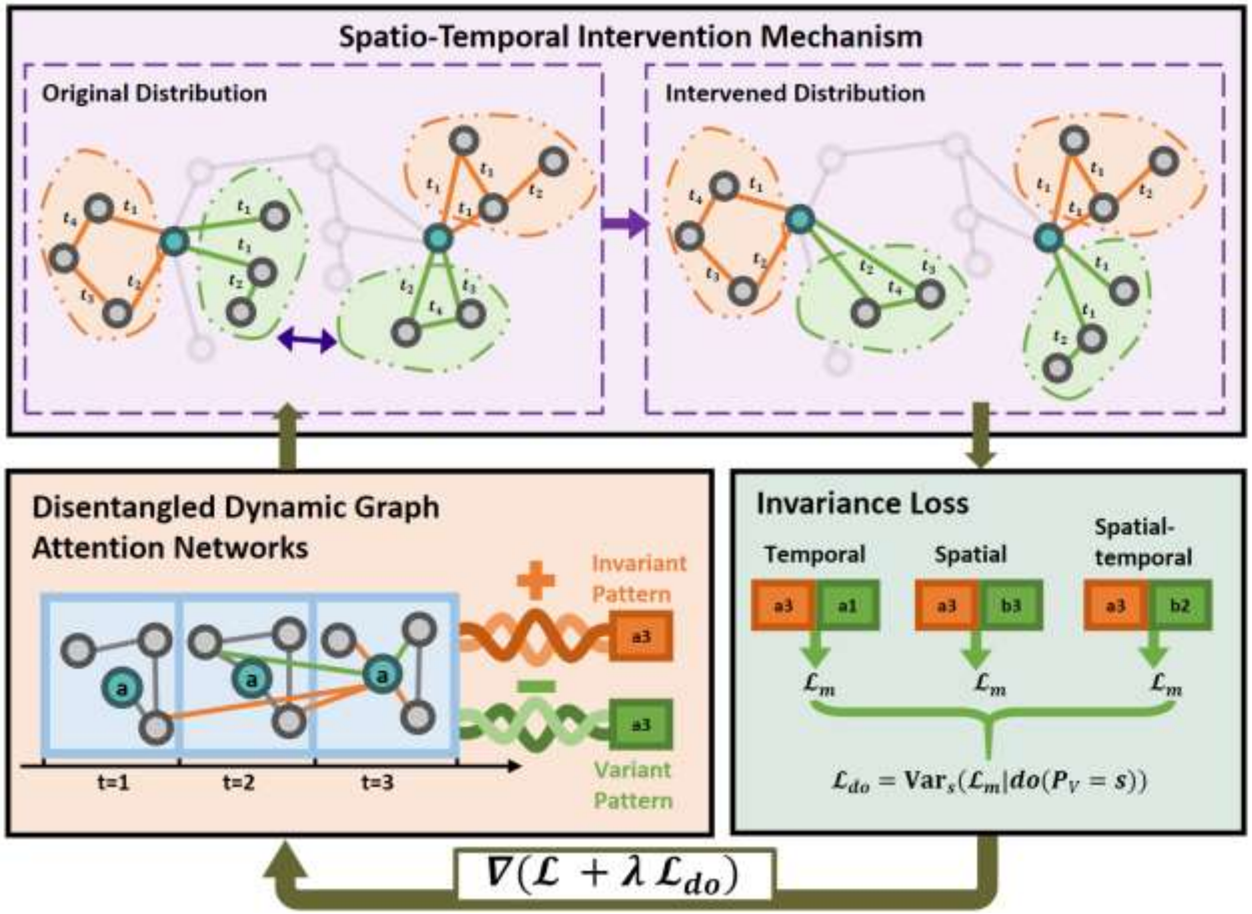
Picture credit: ROLAND: graph learning framework for dynamic graphs, KDD 2022

- Distribution shifts can be spatio-temporal



DIDA: Method

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



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

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

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

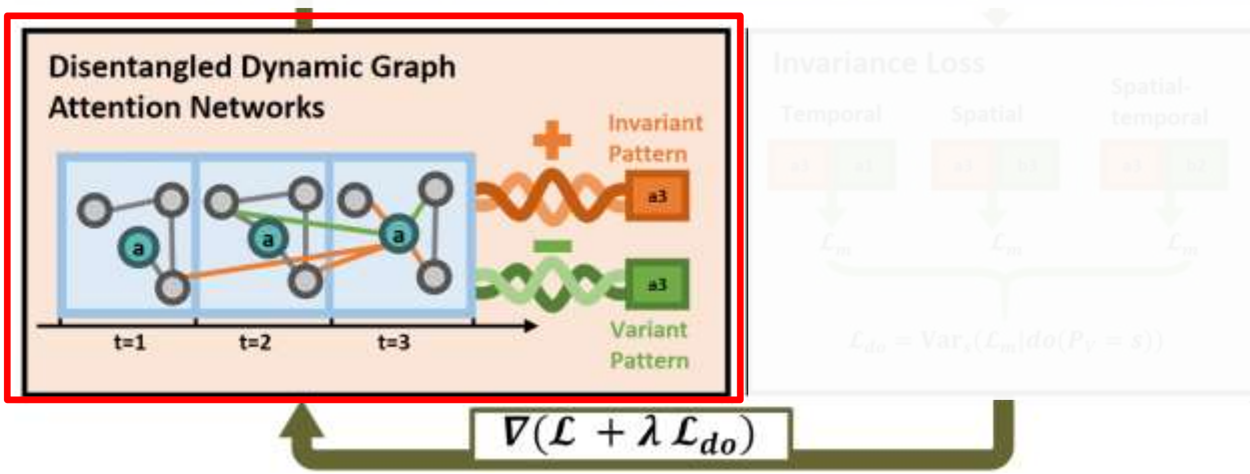
  - Then calculate message-passing

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

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

  - Updating node representation

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

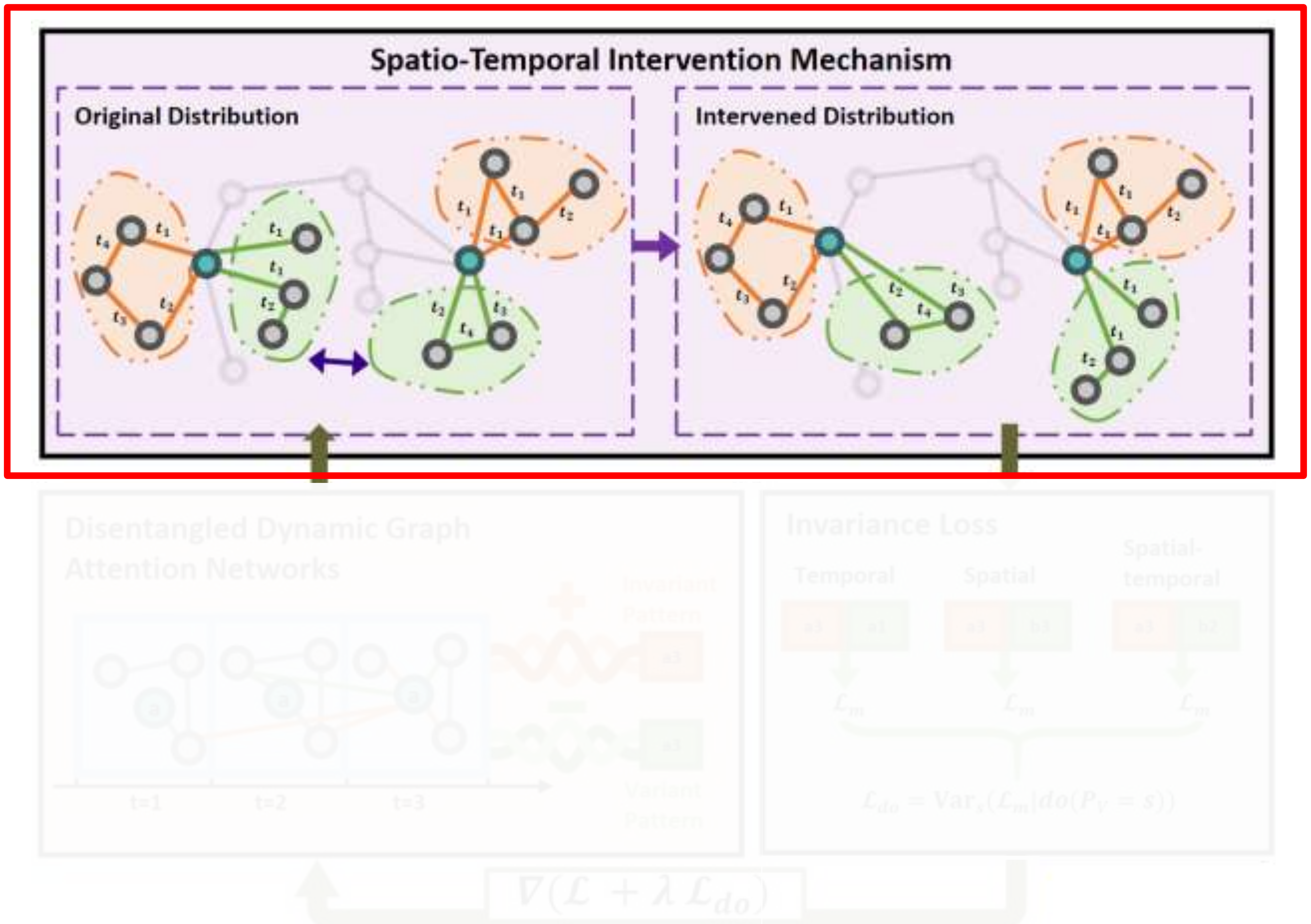


Dynamic Graph Neural Networks Under Spatio-Temporal Distribution Shift. ***NeurIPS, 2022.***

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:

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

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

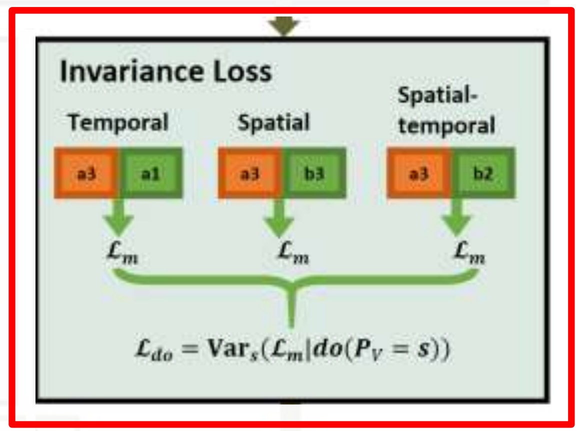
- Practical version: intervention-invariant regularization

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

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

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

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



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**

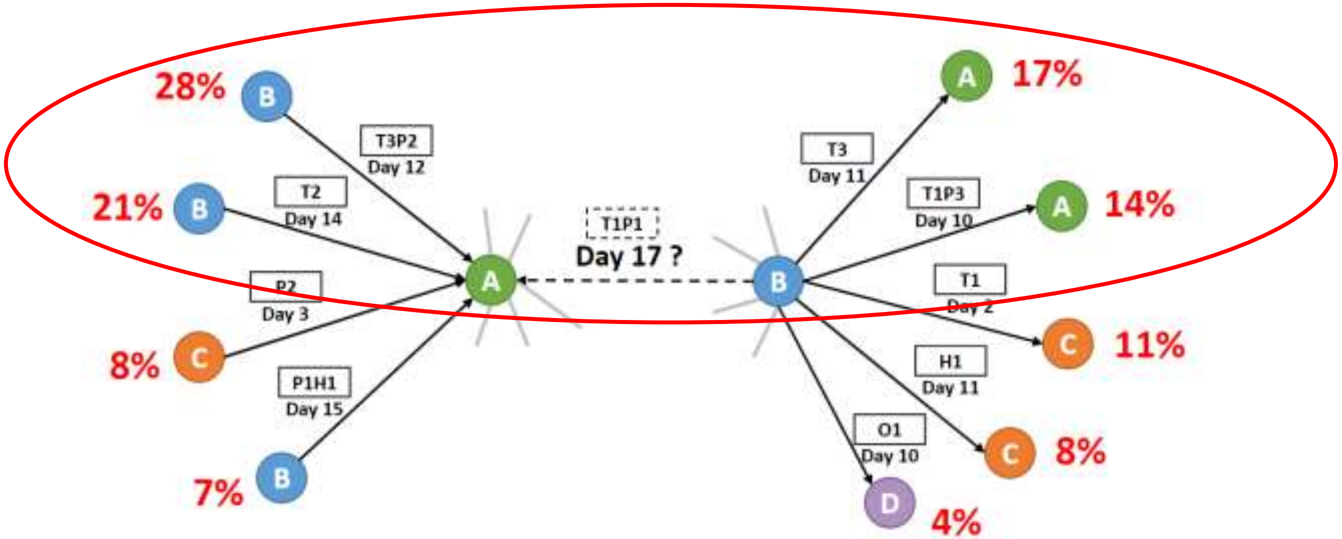
- Real-world datasets

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**

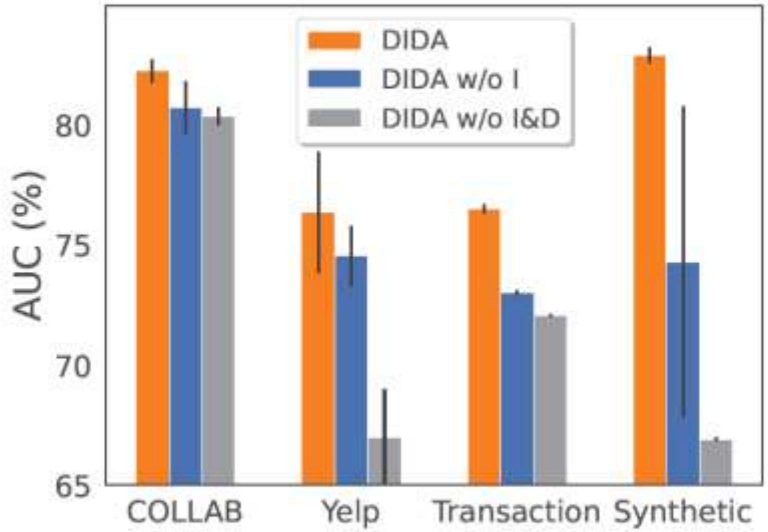
Dynamic Graph Neural Networks Under Spatio-Temporal Distribution Shift. ***NeurIPS, 2022.***

DIDA: Experiments

- Showcases:



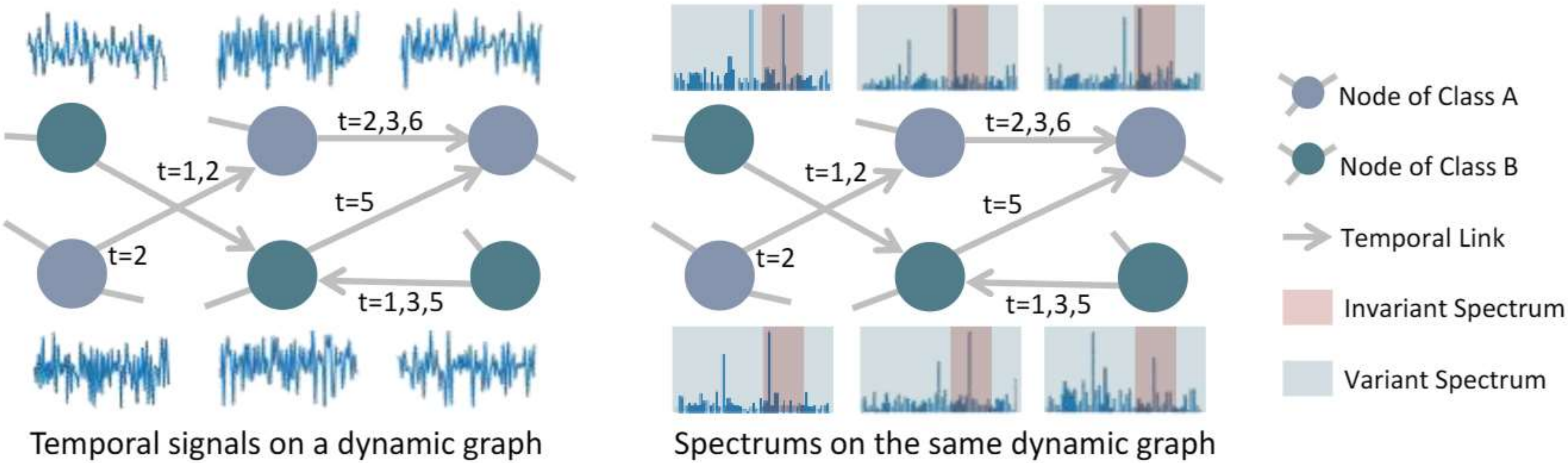
- Ablation studies:



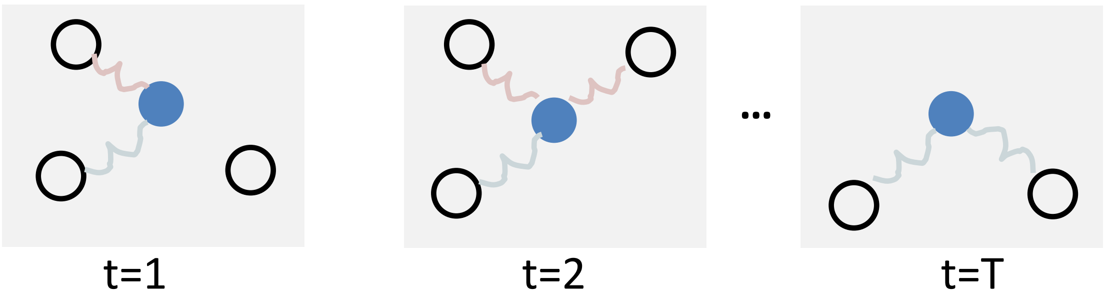
Dynamic Graph Neural Networks Under Spatio-Temporal Distribution Shift. ***NeurIPS, 2022.***

Spectral Invariant Learning for Dynamic Graphs (SILD)

- However, in many cases, distribution shift may be unobservable in the time domain while observable in the spectral domain

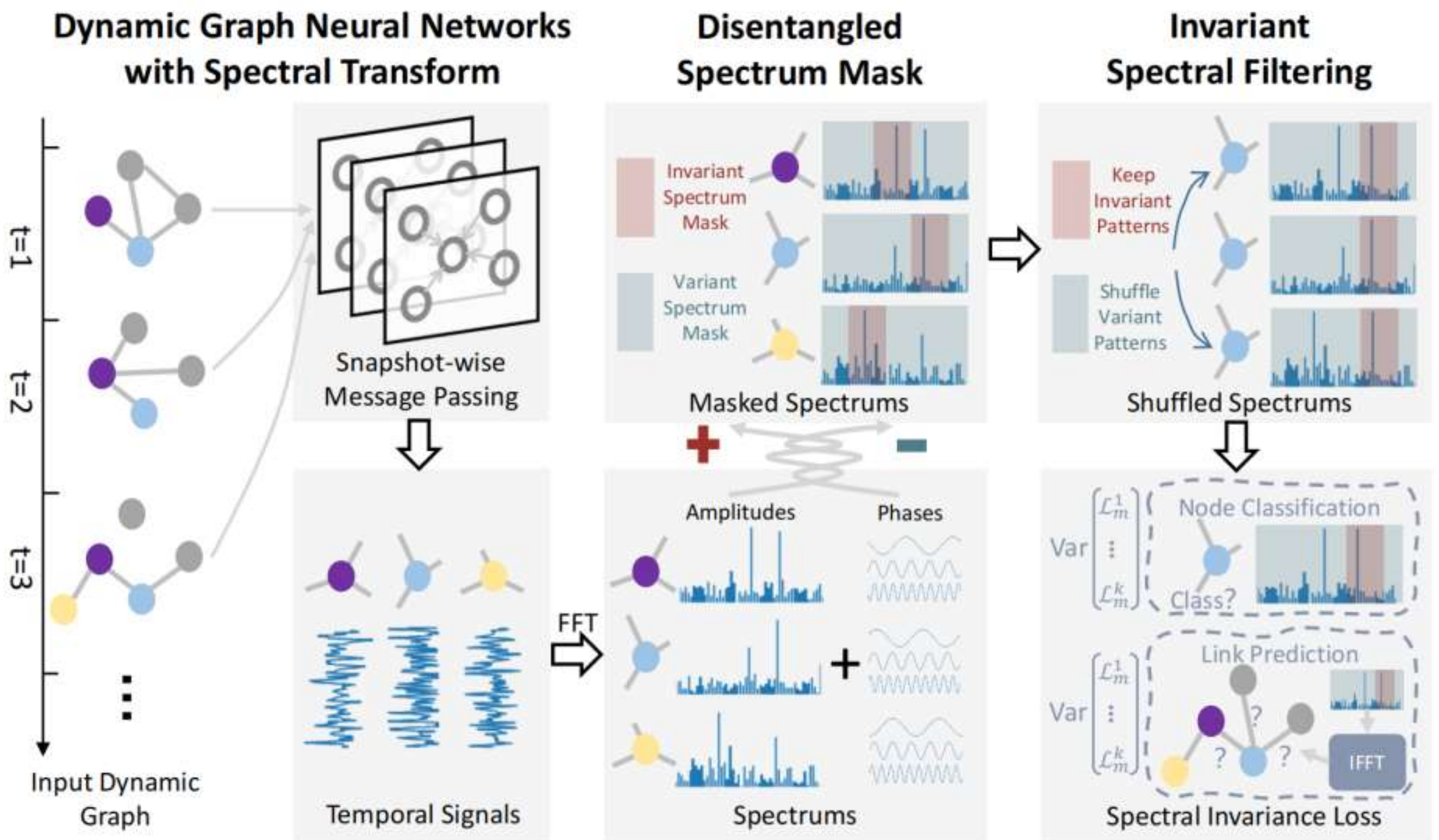


- **Motivation example:** A simple dynamic graph with two frequency components (having variant & invariant relationship with the labels)



Spectral Invariant Learning for Dynamic Graphs under Distribution Shifts. ***NeurIPS, 2023.***

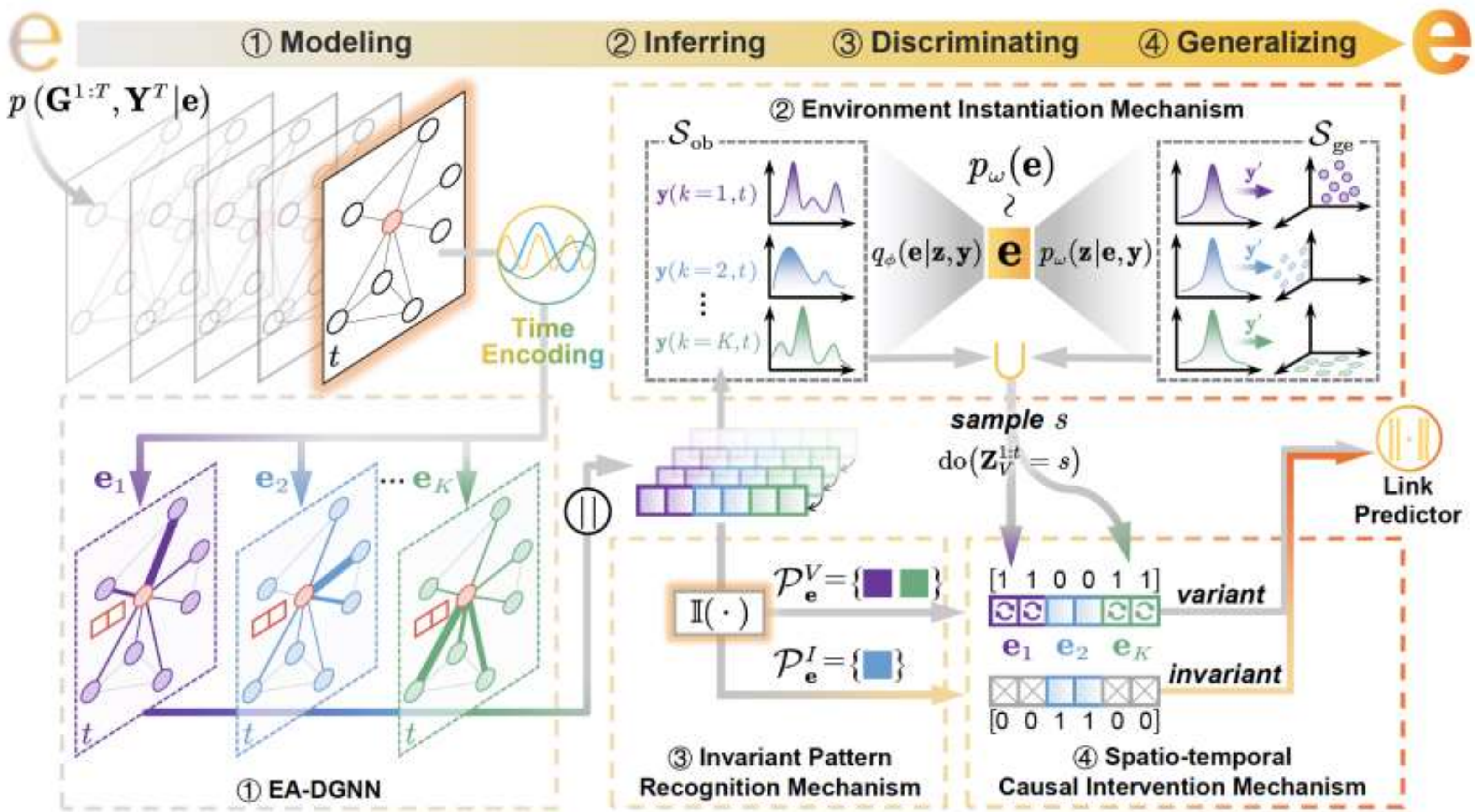
SILD: Framework



Spectral Invariant Learning for Dynamic Graphs under Distribution Shifts. ***NeurIPS, 2023.***

EAGLE: Environment-Aware dynamic Graph Learning

- **Key Idea**: investigate environments carefully
  - Find the spatio-temporal invariant patterns then apply causal inference to decorrelations



Environment-aware dynamic graph learning for out-of-distribution generalization. ***NeurIPS, 2023.***

Recap: Graph Invariant Learning in the Topology Space

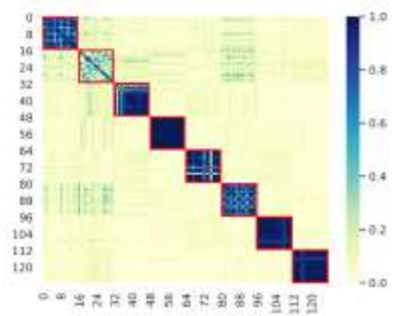
Static graphs:

- GIL (NeurIPS'22): cluster variant subgraphs into environments
- DIR (ICLR'22): intervention to capture invariance/variance
- NIL (ACM TOIS'23): generalize to node-centric tasks
- GOODFormer (arXiv'25): invariance for attention and position encoding

Dynamic graphs:

- DIDA (NeurIPS'22): intervention for spatio-temporal invariance
- SILD (NeurIPS'23): invariance in the spectral domain
- EAGLE (NeurIPS'23): model invariance from environments

Take-Home Message



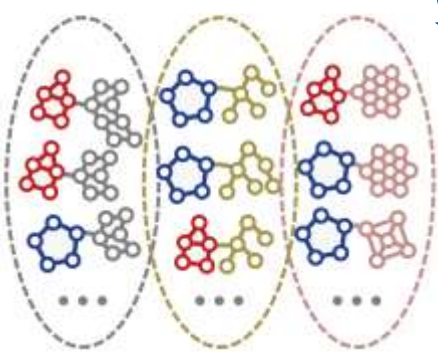
Invariance-guided Graph Representation Learning

Encourage Independence and Disentanglement in representations

OOD-GNN *(IEEE TKDE'22)*
StableGNN *(IEEE TPAMI'23)*
IDGCL *(IEEE TKDE'22)*
OOD-GCL *(ICML'24)*

Customize architectures that can generalize

GRACES *(ICML'22)*
OMGNAS *(AAAI'24)*
DCGAS *(AAAI'24)*
CARNAS *(KDD'25)*



Finding invariant and variant graph structures

GIL *(NeurIPS'22)*
DIR *(ICLR'22)*
NIL *(ACM TOIS'23)*
GOODFormer *(arXiv'25)*

DIDA *(NeurIPS'22)*
SILD *(NeurIPS'23)*
EAGLE *(NeurIPS'23)*

Survey

Out-Of-Distribution Generalization on Graphs: A Survey

Haoyang Li, Xin Wang, *Member, IEEE,* Ziwei Zhang, *Member, IEEE,* Wenwu Zhu, *Fellow, IEEE*

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 in-distribution hypothesis, i.e., testing and training graph data are identically distributed. However, this in-distribution 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 in-distribution hypothesis, has made great progress and attracted ever-increasing attention from the research community. In this paper, we comprehensively survey OOD generalization on graphs and present a detailed review of recent advances in this area. First, we provide a formal problem definition of OOD generalization on graphs. Second, we categorize existing methods into three classes from conceptually different perspectives, i.e., data, model, and learning strategy, based on their positions in the graph machine learning pipeline, followed by detailed discussions for each category. We also review the theories related to OOD generalization on graphs and introduce the commonly used graph datasets for thorough evaluations. Finally, we share our insights on future research directions.

Index Terms—Graph Machine Learning, Graph Neural Network, Out-Of-Distribution Generalization.

Out-Of-Distribution Generalization on Graphs: A Survey. ***IEEE TPAMI 2025***.

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

THANK YOU!

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