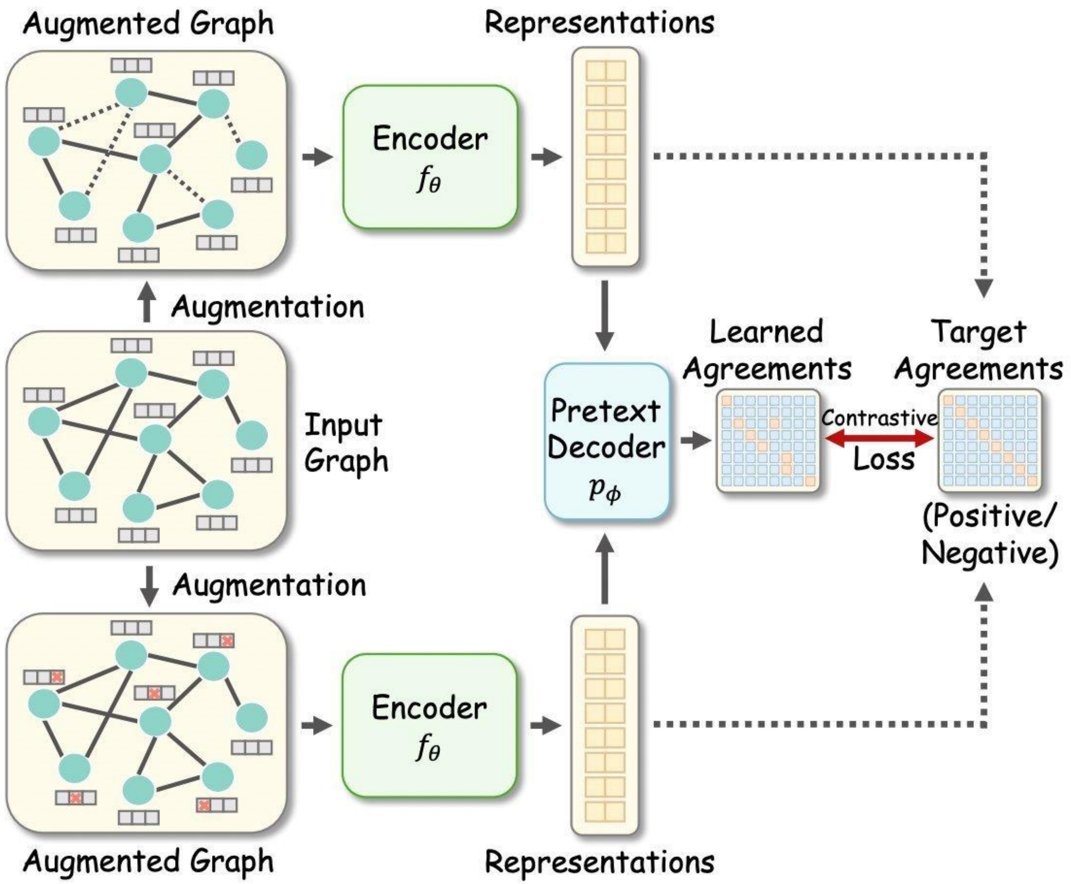
Graph Self-Supervised Learning

Qianqiu Zhang

Outline

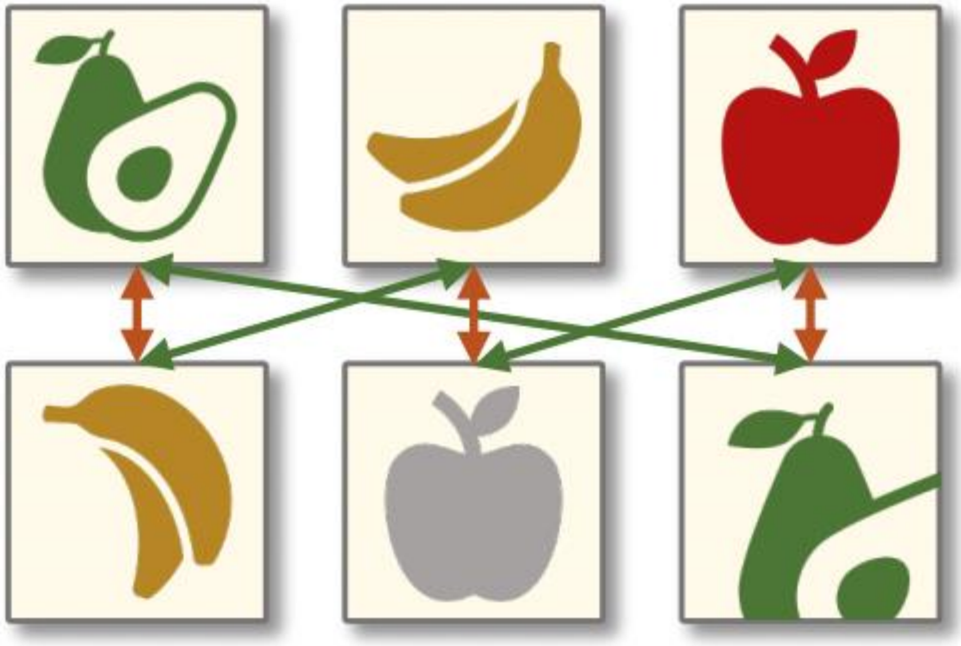
- Contrast-based methods
  - Graph augmentations
  - Graph contrastive learning pretext task
  - Mutual information estimation
- Hybrid methods
- Experiments
- Discussion

Graph contrastive learning



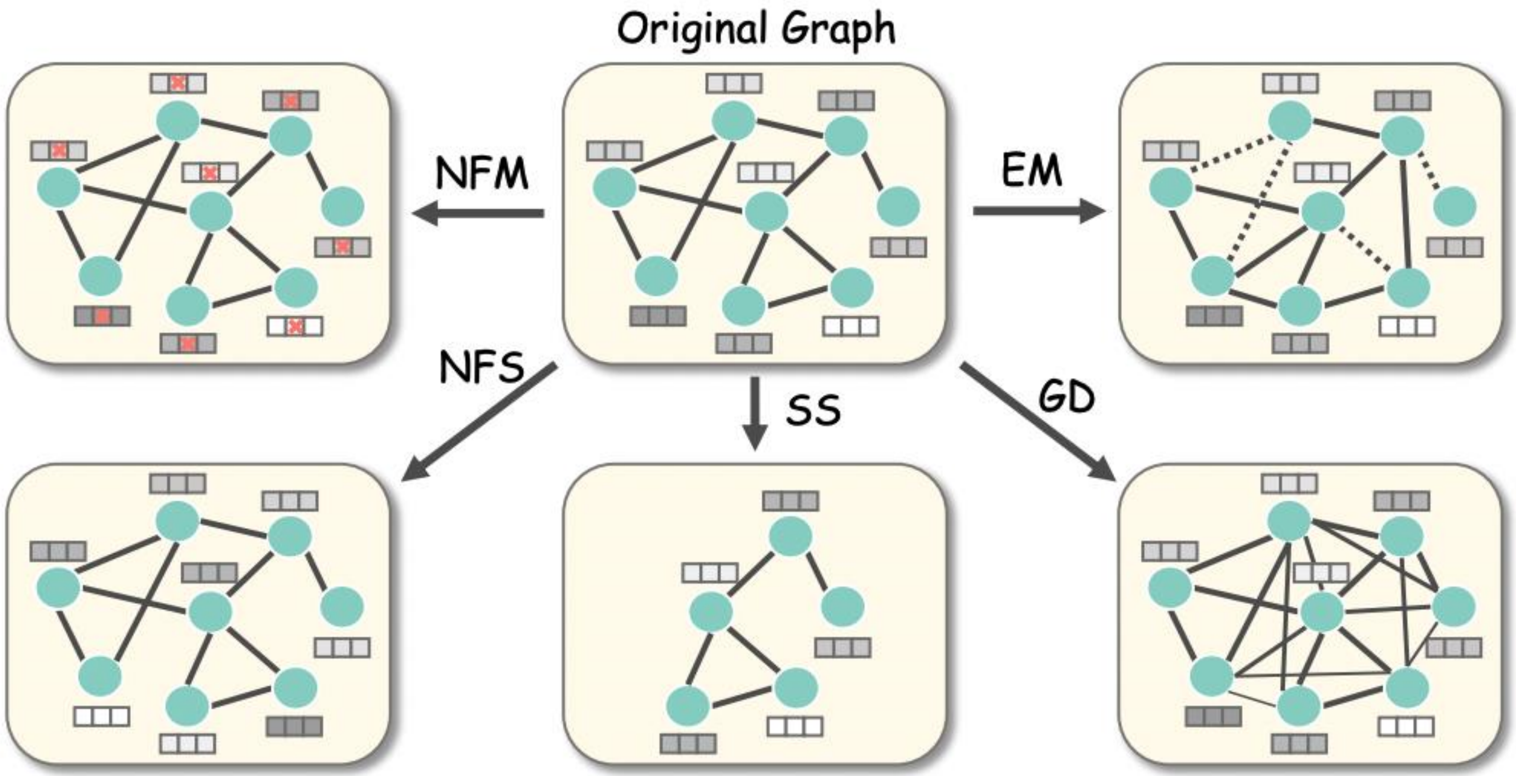
Contrast based methods

- Related works
  - Computer vision: rotation, cutout, cropping, etc.



Graph augmentation

- Attributive-based
- Topological-based
- Hybrid



Brief examples of five types of common graph augmentations, including Node Feature Masking (NFM), Node Feature Shuffle (NFS), Edge Modification (EM), Graph Diffusion (GD), and Subgraph Sampling (SS).

Graph augmentation

- Annotation
    - i-th augmented graph instance $\tilde{\mathcal{G}}^{(i)} = t_i(\mathcal{G})$

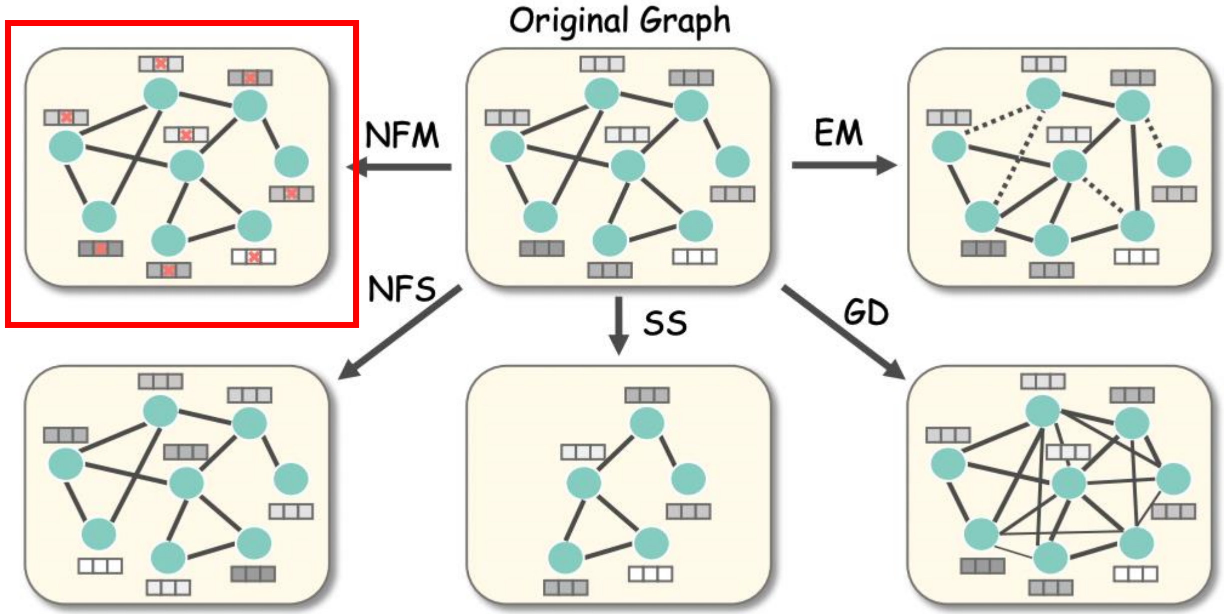
Attributive augmentations

- Only apply to node feature

$$\tilde{\mathcal{G}}^{(i)} = (\mathbf{A}, \tilde{\mathbf{X}}^{(i)}) = \left(\mathbf{A}, t_i(\mathbf{X})\right)$$

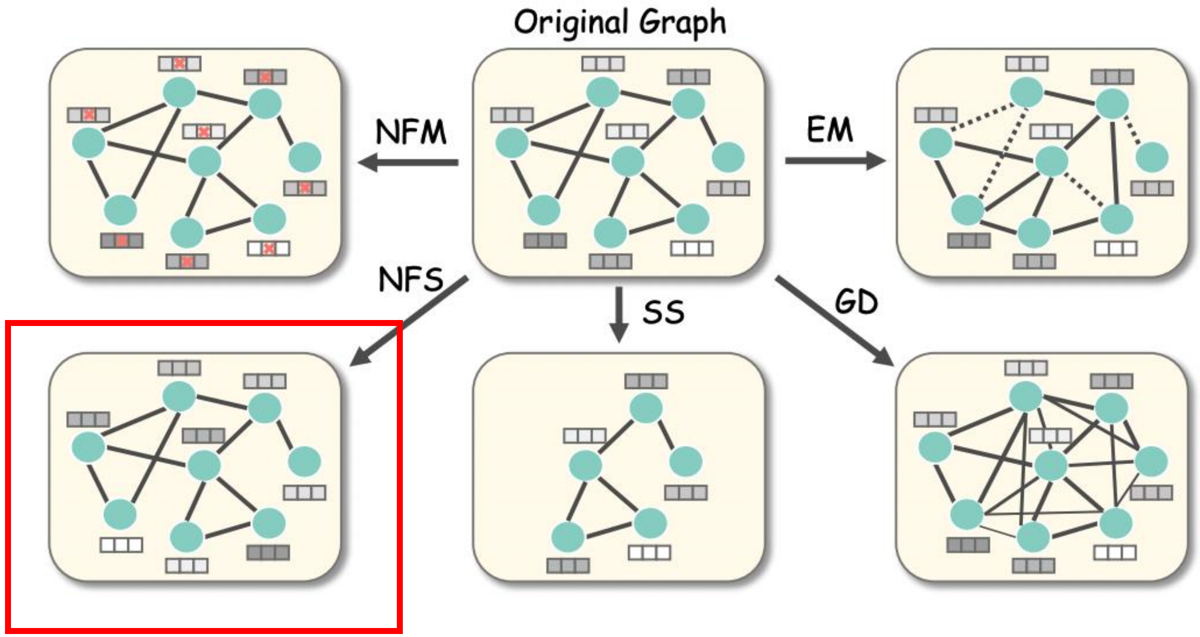
Node feature masking $t_i(\mathbf{X}) = \mathbf{M} \circ \mathbf{X}$

- Node feature masking: randomly mask features of some portions of nodes in the graph
  - GCA: keep important node features unmasked, and assign higher masking probability to less important nodes
    - importance is measured by node centrality i.e., degree centrality, eigenvector centrality, PageRank centrality



Node feature shuffle $t_i(\mathbf{X}) = [\mathbf{X}]_{\tilde{\mathcal{V}}}$

- Several nodes in the augmented graph are placed to other positions when compared with the input graph



Topological augmentations

- Work on adjacency matrix

$$\tilde{\mathcal{G}}^{(i)} = (\tilde{\mathbf{A}}^{(i)}, \mathbf{X}) = \left(t_i(\mathbf{A}), \mathbf{X}\right)$$

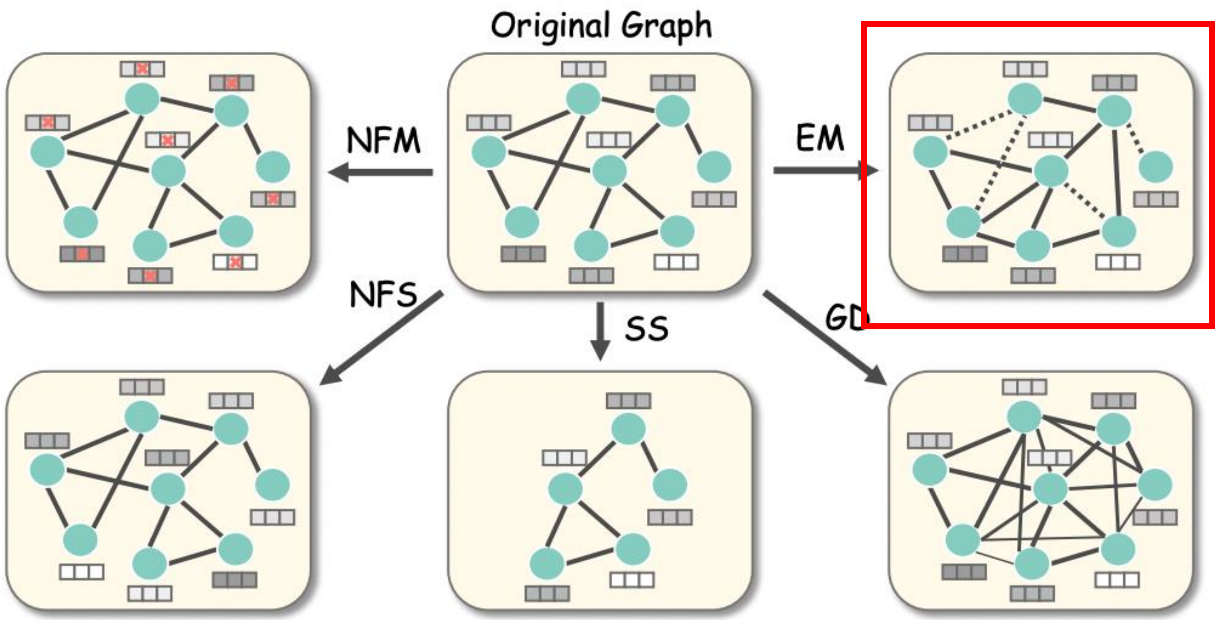
Edge modification

- randomly dropping and inserting a portion of edge

$$t_i(\mathbf{A}) = \mathbf{M_1} \circ \mathbf{A} + \mathbf{M_2} \circ (1 - \mathbf{A})$$

M_1 and M_2 are edge dropping and insertion matrices

- M_1 and M_2 can be generated by adversarial training

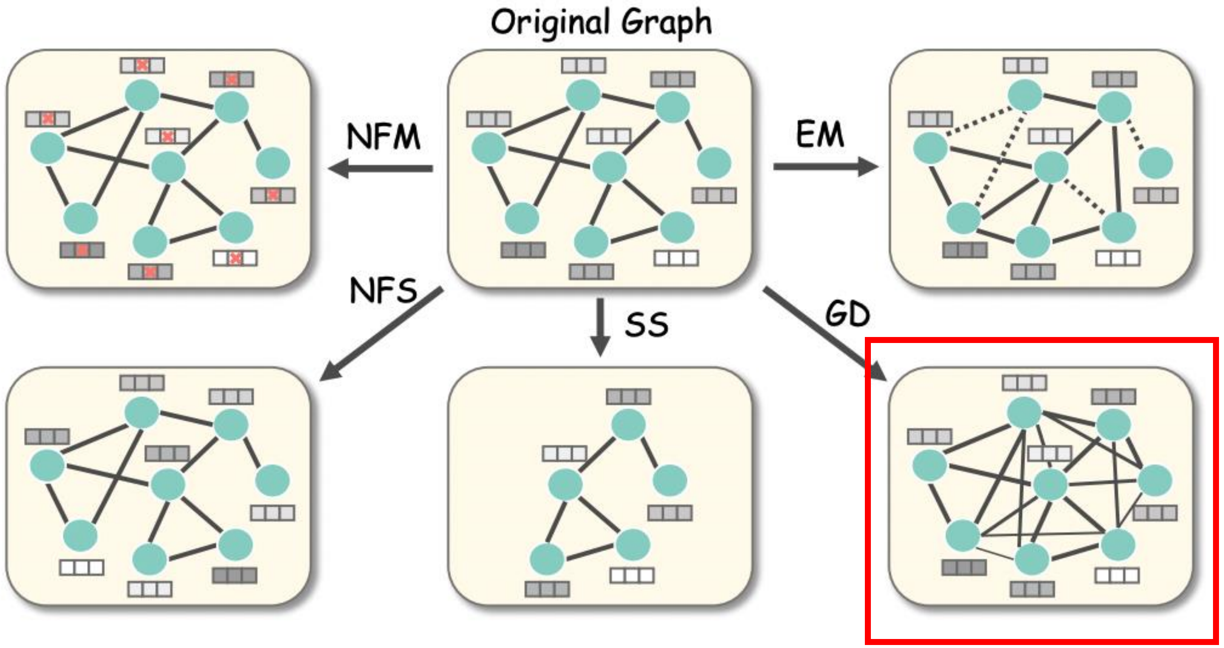


Graph diffusion

- connecting nodes with their indirectly connected neighbors with calculated weights
  - Heat kernel-based

$$t_i(\mathbf{A}) = \exp\left(\iota \mathbf{A}\mathbf{D}^{-1} - \iota\right)$$ where ι denotes the diffusion time

  - PageRank diffusion



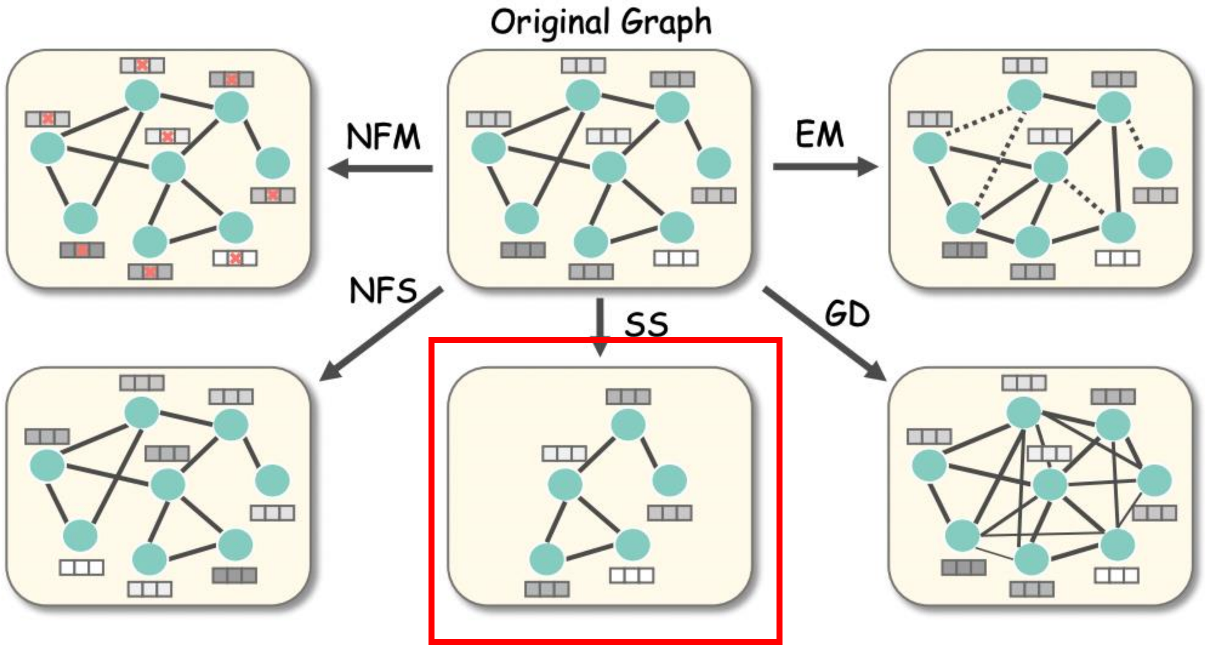
Hybrid augmentation

- Apply both attributive and topological augmentation

$$\tilde{\mathcal{G}}^{(i)} = (\tilde{\mathbf{A}}^{(i)}, \tilde{\mathbf{X}}^{(i)}) = (t_i(\mathbf{A}, \mathbf{X}))$$

Subgraph sampling $t_i(\mathbf{A}, \mathbf{X}) = [(\mathbf{A}, \mathbf{X})]_{\mathcal{V}' \in \mathcal{V}}$

- It samples a portion of nodes and their underlying linkages as augmented graph instances
  - uniform sampling, random walk-based sampling, and top-k importance-based sampling.



Graph contrastive learning

- Mutual information maximization

$$\theta^*, \phi^* = \arg\min_{\theta,\phi} \mathcal{L}_{ssl}\left(p_\phi\Big(f_\theta(\tilde{\mathcal{G}}^{(1)}), f_\theta(\tilde{\mathcal{G}}^{(2)})\Big)\right)$$

- Same-scale: discriminate same scale of graph instance (e.g., node vs node)
- Cross-scale: contrasting across multiple granularities (e.g., node vs graph)

Same-scale

- Node level

$$\theta^* = \arg\min_{\theta} \frac{1}{|\mathcal{V}|} \sum_{v_i \in \mathcal{V}} \mathcal{L}_{con}\Big(p\big([f_\theta(\mathbf{A}, \mathbf{X})]_{v_i}, [f_\theta(\mathbf{A}, \mathbf{X})]_{v_c}\big)\Big),$$

  - v_c denotes the contextual node (i.e., neighboring node) of v_i
  - Discriminator function can be dot product
  - The goal is to maximize the co-occurrence of nodes within the same walk
  - Heterogeneous graph: enforce nodes within the same meta-path to share closer semantic information

Same-scale

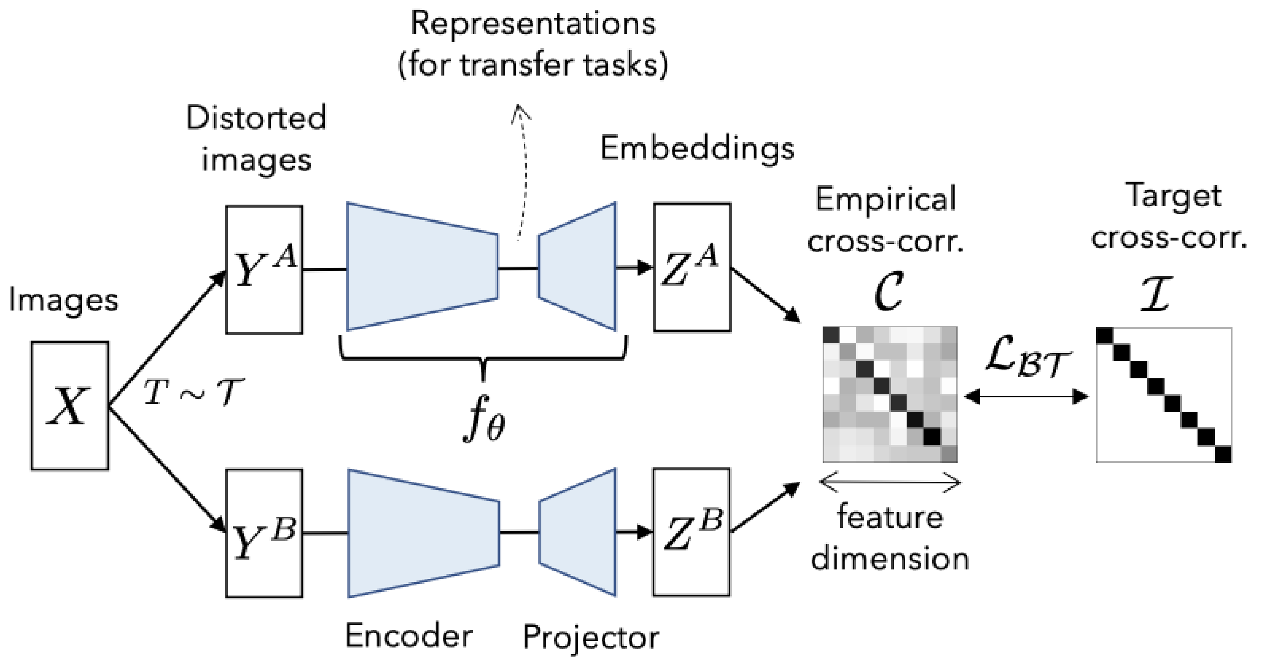
- Graph augmentations

$$\theta^*, \phi^* = \arg\min_{\theta,\phi} \mathcal{L}_{con}\Big(p_\phi\big(f_\theta(\tilde{\mathbf{A}}^{(1)}, \tilde{\mathbf{X}}^{(1)}), f_\theta(\tilde{\mathbf{A}}^{(2)}, \tilde{\mathbf{X}}^{(2)})\big)\Big)$$

  - Generate two views (e.g., node masking and edge modifying)
  - Discriminator function can be parametarized bilinear transformation or cosine similarity
  - The goal is to pull the representations of the same nodes in two views as close as possible

Same-scale

- Omit negative sampling
  - Negati rge batch size
  - BYOL (
    - For predicting *target*.
    - Min
  - Barlow
    - Min ication of node emb



Same-scale

- Graph-level

$$\theta^*, \phi^* = \arg\min_{\theta,\phi} \mathcal{L}_{con}\Big(p_\phi\big(\tilde{\mathbf{g}}^{(1)}, \tilde{\mathbf{g}}^{(2)}\big)\Big)$$

$$\tilde{\mathbf{g}}^{(i)} = \mathcal{R}\big(f_\theta(\tilde{\mathbf{A}}^{(i)}, \tilde{\mathbf{X}}^{(i)})\big)$$

Cross-scale

- Discrimination across various graph topologies
  - Patch-global
  - Context-global

Patch-global Cross-scale

- Contrast node-level embeddings with graph-level readout embeddings

$$\theta^*, \phi^* = \arg\min_{\theta,\phi} \frac{1}{|\mathcal{V}|} \sum_{v_i \in \mathcal{V}} \mathcal{L}_{con}\Big(p_\phi\big(\tilde{\mathbf{h}}_i^{(1)}, \tilde{\mathbf{g}}^{(2)}\big)\Big)$$

$$\tilde{\mathbf{h}}_i^{(1)} = [f_\theta(\tilde{\mathbf{A}}^{(1)}, \tilde{\mathbf{X}}^{(1)})]_{v_i} \quad \tilde{\mathbf{g}}^{(2)} = \mathcal{R}\big(f_\theta(\tilde{\mathbf{A}}^{(2)}, \tilde{\mathbf{X}}^{(2)})\big)$$

  - Apply node embeddings and graph embeddings to 2 augmented views
  - discriminate whether a node belongs to the given graph, mainly for node-level embeddings

Context-global Cross-scale

- Contextual subgraph augmented by subgraph sampling

$$\theta^*, \phi^* = \arg\min_{\theta,\phi} \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \mathcal{L}_{con}\Big(p_\phi(\tilde{\mathbf{h}}_s, \tilde{\mathbf{g}})\Big)$$

$$\tilde{\mathbf{h}}_s = \mathcal{R}\big([f_\theta(\tilde{\mathbf{A}}, \tilde{\mathbf{X}})]_{v_i \in s}\big), \text{ and } \tilde{\mathbf{g}} = \mathcal{R}\big(f_\theta(\tilde{\mathbf{A}}, \tilde{\mathbf{X}})\big)$$

- For the graph representation, it can either be graph-level representation over all subgraph or obtained from original graph

$$\tilde{\mathbf{g}} = \mathcal{R}\big(f_\theta(\mathbf{A}, \mathbf{X})\big)$$

Context-global Cross-scale

Purpose of encoder	Local contextual	Global representation
Edge-level embeddings	Target edge embedding between two nodes	Readout node embeddings
Graph-level embeddings	Aggregation of sampled (learning based) subgraph embedding	full-graph representation

Mutual information loss

- Loss needs to pull positive samples closer and negative samples more distant
- Joint density to be as positive as possible, and marginal density to be negative

$$\begin{aligned}\mathcal{MI}(\mathbf{h}_i, \mathbf{h}_j) &= KL\big(P(\mathbf{h}_i, \mathbf{h}_j) || P(\mathbf{h}_i)P(\mathbf{h}_j)\big) \\ &= \mathbb{E}_{P(\mathbf{h}_i, \mathbf{h}_j)}\Big[\log \frac{P(\mathbf{h}_i, \mathbf{h}_j)}{P(\mathbf{h}_i)P(\mathbf{h}_j)}\Big],\end{aligned}$$

Jensen-Shannon Estimator

$$\mathrm{JSD}(P \parallel Q) = \frac{1}{2}D(P \parallel M) + \frac{1}{2}D(Q \parallel M),$$

where $M = \frac{1}{2}(P + Q)$ is a mixture distribution of P and Q.

Jensen-Shannon Estimator

- h_i, h_j are from same distribution/augmentation, h'_j from other distribution (binary cross entropy loss)

$$\mathcal{L}_{con}\Big(p_\phi(\mathbf{h}_i, \mathbf{h}_j)\Big) = -\mathcal{MI}_{JSD}(\mathbf{h}_i, \mathbf{h}_j)$$

$$= \mathbb{E}_{\mathcal{P}\times\widetilde{\mathcal{P}}}\Big[\log\big(1 - p_\phi(\mathbf{h}_i, \mathbf{h'}_j)\big)\Big] - \mathbb{E}_{\mathcal{P}}\Big[\log\big(p_\phi(\mathbf{h}_i, \mathbf{h}_j)\big)\Big]$$

Averaged over number of negative and positive samples

h_i is usually readout of node embeddings

Noise contrastive estimator

- One positive and N negative samples

$$\mathcal{L}_{con}\Big(p_\phi(\mathbf{h}_i, \mathbf{h}_j)\Big) = -\mathcal{MI}_{NCE}(\mathbf{h}_i, \mathbf{h}_j)$$

$$= -\mathbb{E}_{\mathcal{P}\times\widetilde{\mathcal{P}}^N}\left[\log \frac{e^{p_\phi(\mathbf{h}_i, \mathbf{h}_j)}}{e^{p_\phi(\mathbf{h}_i, \mathbf{h}_j)} + \sum_{n\in N} e^{p_\phi(\mathbf{h}_i, \mathbf{h}'_n)}}\right]$$

Triplet loss

- Triplet loss requires three inputs (anchor, positive, and negative)
- The goal is to minimize the distance between the anchor and the positive example while raising the gap between the anchor and the negative example.

$$\mathcal{L}_{con}\Big(p(\mathbf{h}_i, \mathbf{h}_j)\Big) = \mathbb{E}_{\mathcal{P}\times\widetilde{\mathcal{P}}}\Big[\max\Big[p_\phi(\mathbf{h}_i, \mathbf{h}_j) - p_\phi(\mathbf{h}_i, \mathbf{h'}_j) + \epsilon, 0\Big]\Big],$$

BYOL loss

- For a given representation, *target*, train a network named *online* by predicting *target.*
- Minimize similarity loss i.e., L_2 loss between *target* and *online*

True representation

Online network

$$\mathcal{L}_{con}\Big(p(\mathbf{h}_i, \mathbf{h}_j)\Big) = \mathbb{E}_{\mathcal{P}\times\mathcal{P}}\left[2 - 2\cdot \frac{[p_\psi(\mathbf{h}_i)]^T \mathbf{h}_j}{\|p_\psi(\mathbf{h}_i)\|\,\|\mathbf{h}_j\|}\right]$$

Barlow twins loss

- Minimize the difference between identity matrix and matrix multiplication of node embeddings on two augmented graph views

$$\mathcal{L}_{\mathcal{BT}} \triangleq \underbrace{\sum_i (1-\mathcal{C}_{ii})^2}_{\text{invariance term}} + \lambda \underbrace{\sum_i \sum_{j \neq i} {\mathcal{C}_{ij}}^2}_{\text{redundancy reduction term}}$$

$$\mathcal{L}_{con}\left(\mathbf{H}^{(\mathbf{1})}, \mathbf{H}^{(\mathbf{2})}\right) = \mathbb{E}_{\mathcal{B} \sim \mathcal{P}^{|\mathcal{B}|}} \left[\sum_a \left(1 - \frac{\sum_{i \in \mathcal{B}} \mathbf{H}_{ia}^{(1)} \mathbf{H}_{ia}^{(2)}}{\left\|\mathbf{H}_{ia}^{(1)}\right\| \left\|\mathbf{H}_{ia}^{(2)}\right\|}\right)^2 + \lambda \sum_a \sum_{b \neq a} \left(\frac{\sum_{i \in \mathcal{B}} \mathbf{H}_{ia}^{(1)} \mathbf{H}_{ib}^{(2)}}{\left\|\mathbf{H}_{ia}^{(1)}\right\| \left\|\mathbf{H}_{ib}^{(2)}\right\|} \right)^2 \right],$$

Hybrid method

- Use multiple SSL method
- Objective function is the weighted sum of two or more self-supervised objectives

Hybrid method

- Combine different generation-based tasks
- Integrate generative and contrastive pretext tasks
- Combine multiple contrast-based tasks
- Combine different auxiliary property-based tasks

Approach	Pretext Task Categories	Downstream Task Level	Training Scheme	Data Type of Graph
GPT-GNN [9]	FG/SG	Node/Link	PF	Hetero.
Graph-Bert [39]	FG/SG	Node	PF	Attributed
PT-DGNN [105]	FG/SG	Link	PF	Dynamic
M. et al. [45]	FG/FG/FG	Node	JL	Attributed
GMI [41]	SG/NSC	Node/Link	URL	Attributed
CG^3 [106]	SG/NSC	Node	JL	Attributed
MVMI-FT [107]	SG/PGCC	Node	URL	Attributed
GraphLoG [108]	NSC/GSC/CGCC	Graph	PF	Attributed
HDMI [109]	NSC/PGCC	Node	URL	Multiplex
G-Zoom [110]	NSC/NSC/GSC	Node	URL	Attributed
LnL-GNN [111]	NSC/NSC	Node	JL	Attributed
Hu et al. [50]	SG/APC/APC	Node/Link/Graph	PF	Attributed
GROVER [10]	APC/APC	Node/Link/Graph	PF	Attributed
Kou et al. [112]	FG/SG/APC	Node	JL	Attributed

- FG: feature generation, SG: structure generation
- NSC: node-level same scale, PGCC: patch-global cross-scale, GSC: graph-level same-scale
- APC: auxiliary property classification
- PF: pretraining and finetuning, JL: joint learning, URL: unsupervised representation learning

Empirical results

- Node classification
  - Semi-supervised transductive learning (Cora, Citeseer and Pubmed)
    - 20 nodes per class are used for training, 500/1000 nodes are used for validation/testing
  - Supervised inductive learning (PPI dataset)
    - 20 graphs are employed to train the model, while 2 graphs are used to validate and 2 graphs are used to test.

Empirical results

Group	Approach	Category	Cora	Citeseer	Pubmed	PPI
Base-lines	GCN [1]	-	81.5	70.3	79.0	-
	GAT [2]	-	83.0	72.5	79.0	97.3
URL	GAE [32]	SG	80.9	66.7	77.1	-
	SIG-VAE [47]	SG	79.7	70.4	79.3	-
	S^2GRL [57]	PAPC	83.7	72.1	82.4	66.0
	DeepWalk [30]	NSC	67.2	43.2	65.3	-
	GraphSAGE [78]	NSC	78.7	69.4	78.1	50.2
	GRACE [33]	NSC	80.0	71.7	79.5	-
	GCA [69]	NSC	81.2	71.8	82.8	-
	GraphCL(N) [81]	NSC	83.6	72.5	79.8	65.9
	BGRL [84]	NSC	80.5	71.0	79.5	-
	G-BT [87]	NSC	81.0	70.8	79.0	-
	MERIT [68]	NSC	83.1	74.0	80.1	-
	DGI [13]	PGCC	82.3	71.8	76.8	63.8
	MVGRL [14]	PGCC	82.9	72.6	79.4	-
	SubG-Con [77]	PGCC	83.5	73.2	81.0	66.9
	GMI [41]	Hybrid	82.7	73.0	80.1	65.0
	MVMI-FT [107]	Hybrid	83.1	72.7	81.0	-
	G-Zoom [110]	Hybrid	84.7	74.2	81.2	-
PF/JL	G. Comp. [17]	FG	81.3	71.7	79.2	
	SuperGAT [49]	SG	84.3	72.6	81.7	74.4
	N. Clu. [17]	CAPC	81.8	71.7	79.2	-
	M3S [40]	CAPC	81.6	71.9	79.3	-
	G. Part. [17]	CAPC	81.8	71.3	80.0	-
	SimP-GCN [58]	APR	82.8	72.6	81.1	-
	Graph-Bert [39]	Hybrid	84.3	71.2	79.3	-
	M. et al. [45]	Hybrid	82.2	71.1	79.3	-
	CG^3 [106]	Hybrid	83.4	73.6	80.2	-

1. URL: purely trained on SSL pretext tasks, learned representations feed into classification decoder; PF/JL: the training labels are accessible for encoders' learning
2. Early methods (random walk-based contrastive and autoencoder-based generative) perform worse than the majority of graph SSL methods
3. Methods employing advanced CV contrastive learning techniques do not show a superior performance
4. Bridge the gap between supervised and SSL methods

Why graph SSL

- Recommender system
  - Pretraining
- Anomaly detection
  - Usually trained with unlabeled data
- Chemistry

Future work

- Pretext Tasks for Complex Types of Graphs
- Augmentation for graph
  - Current methods have limited diversity and uncertain invariance when generating multiple graph views
- Robustness
  - Most graph SSL methods assume input data is perfect, even though real-world data is often noisy.

Thank you