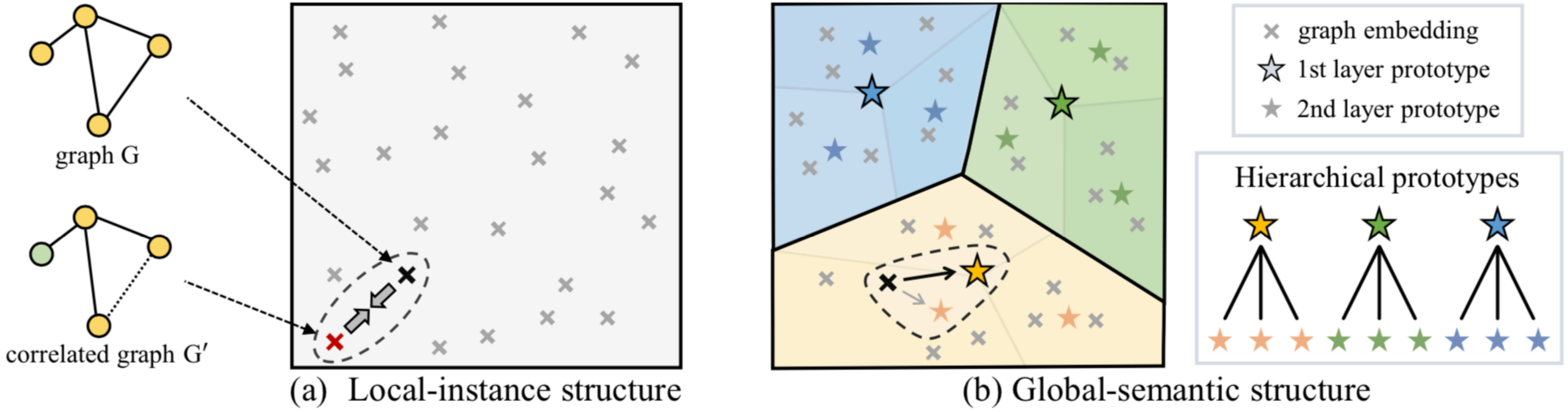
Self-supervised Graph-level Representation Learning with Local and Global Structure

Minghao Xu

Shanghai Jiao Tong University

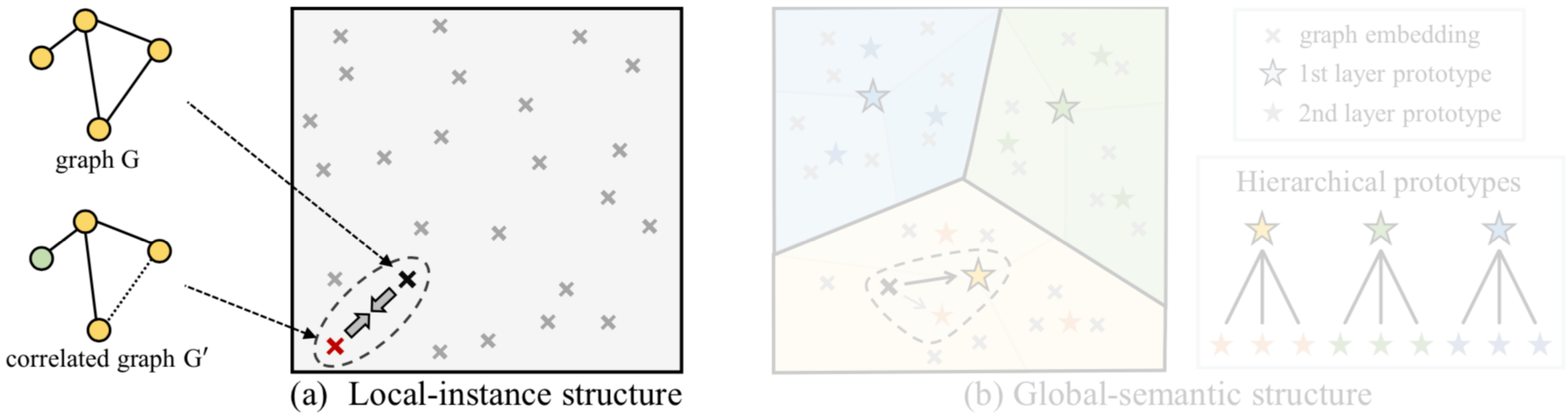
Acknowledgements: Jian Tang, Hongyu Guo, Hang Wang, Bingbing Ni

GraphLoG – Motivation and Definition



- **Motivation:**
  - In many scientific domains, the labeled graphs are usually insufficient. ➡ *self-supervised learning*
  - For self-supervised graph representation learning, both the *local* and *global* structure should be modeled in the latent space.
- **Definition:**
  - **Local-instance structure**: the local similarity between graph instance pairs.
  - **Global-semantic structure**: some global structure reflecting the clustering patterns of the data.

GraphLoG – Learning Local-instance Structure



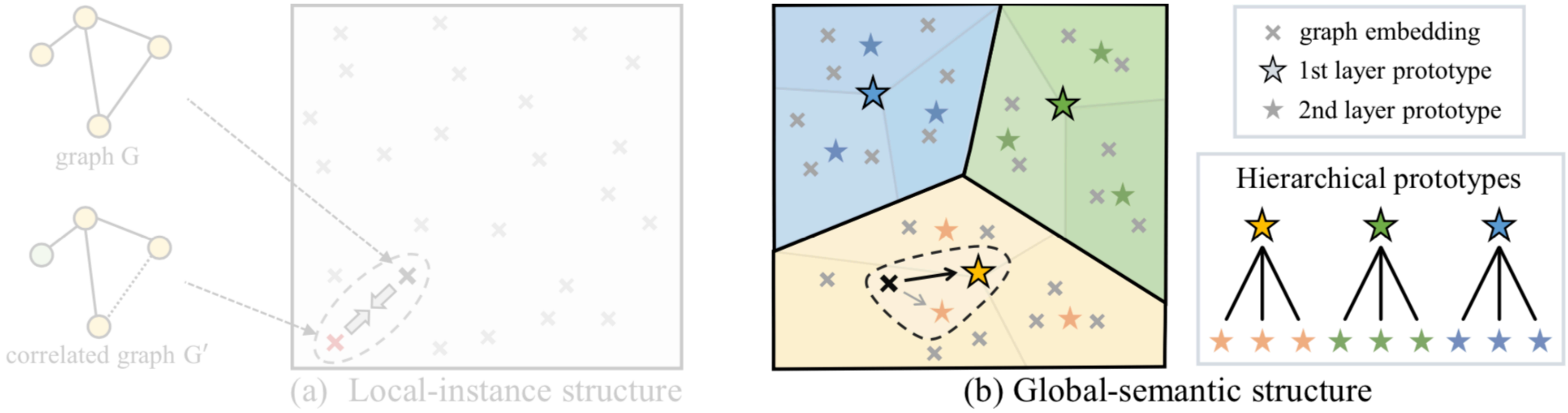
- **Learning scheme:**
  - For a graph G, derive its *correlated counterpart* G' by masking a part of node/edge attributes.
  - Extract the graph and subgraph embeddings for G and G' via a GNN.
  - Measure the similarity of a graph/subgraph pair with the *cosine similarity*.
  - Learning through enhancing the similarity of correlated pairs and diminishing that of negative pairs:

$$\mathcal{L}_{\text{graph}} = -\mathbb{E}_{(\mathcal{G}_+,\mathcal{G}'_+)\sim p(\mathcal{G},\mathcal{G}'),(\mathcal{G}_-,\mathcal{G}'_-)\sim p_n(\mathcal{G},\mathcal{G}')}\left[s(\mathcal{G}_+,\mathcal{G}'_+) - s(\mathcal{G}_-,\mathcal{G}'_-)\right],$$

$$\mathcal{L}_{\text{sub}} = -\mathbb{E}_{(\mathcal{G}_u,\mathcal{G}'_u)\sim p(\mathcal{G}_{\mathbf{v}},\mathcal{G}'_{\mathbf{v}}),(\mathcal{G}_v,\mathcal{G}'_w)\sim p_n(\mathcal{G}_{\mathbf{v}},\mathcal{G}'_{\mathbf{v}})}\left[s(\mathcal{G}_u,\mathcal{G}'_u) - s(\mathcal{G}_v,\mathcal{G}'_w)\right],$$

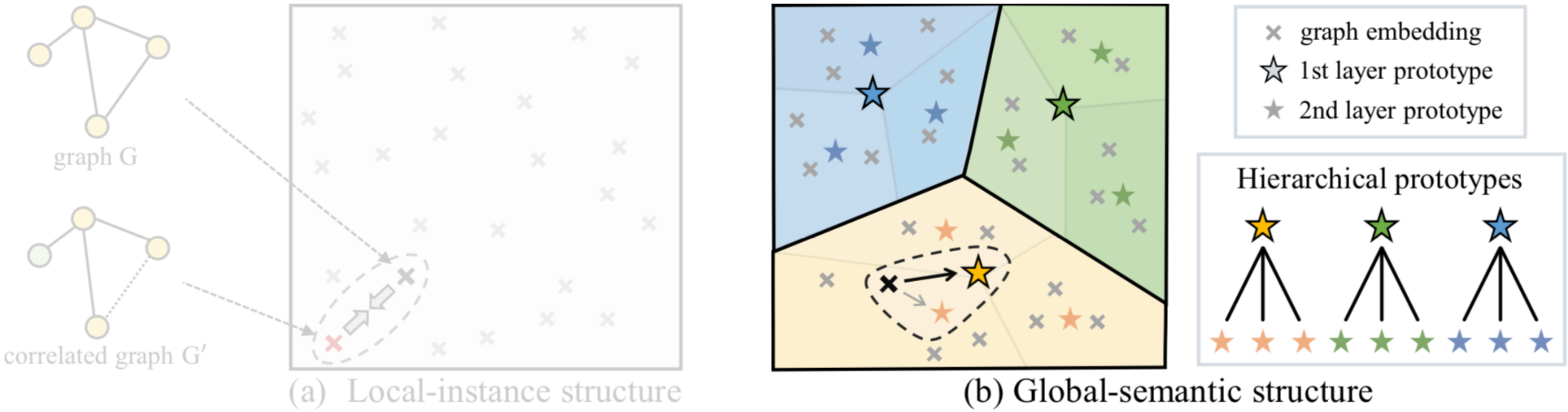
$$\mathcal{L}_{\text{local}} = \mathcal{L}_{\text{graph}} + \mathcal{L}_{\text{sub}}.$$

GraphLoG – Learning Global-semantic Structure



- **Problem formulation: optimizing a latent variable model**
  - Observed data **G**: a set of unlabeled graphs.
  - Model parameters:
    - a. The GNN's parameters θ,
    - b. **Hierarchical prototypes C**: the *representative cluster embeddings* structured as *a set of trees*.
  - Latent variables **Z**: the prototype assignments for all graph samples.
- **Learning objective:**
  - Maximize *the complete data likelihood* governed by model parameters, i.e. $p(\mathbf{G}, \mathbf{Z} | \theta, \mathbf{C})$, via an *online EM algorithm*.

GraphLoG – online EM algorithm for global structure modeling

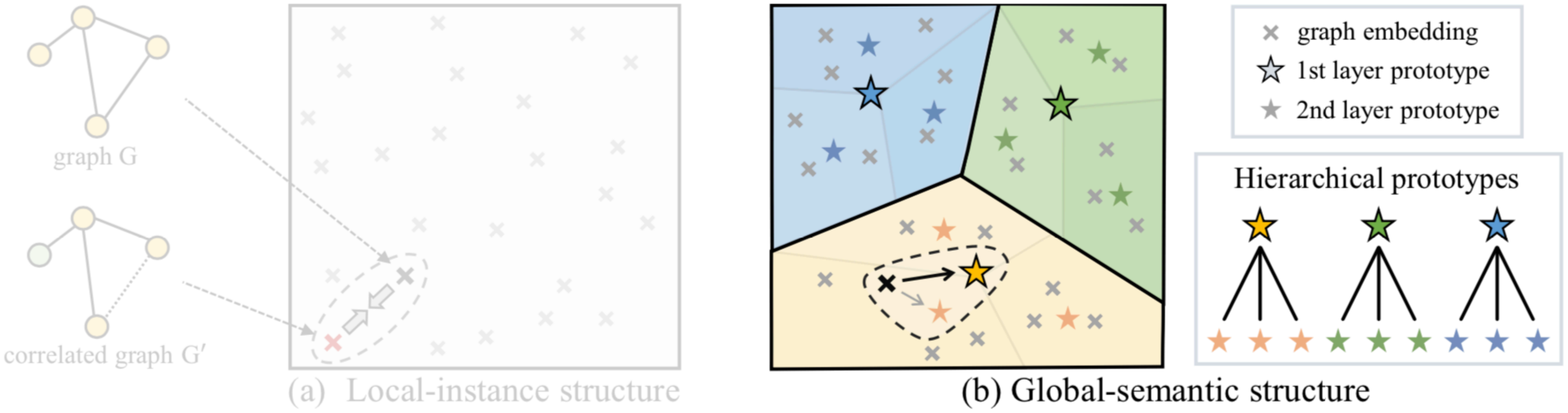


- **E-step:**
  - Sample a mini-batch $\widetilde{\mathbf{G}}$ and estimate the posterior distribution of latent variables in a factorized way:

$$p(\widetilde{\mathbf{Z}}|\widetilde{\mathbf{G}}, \theta_{t-1}, \mathbf{C}_{t-1}) = \prod_{n=1}^{N} p(z_{\mathcal{G}_n}|\mathcal{G}_n, \theta_{t-1}, \mathbf{C}_{t-1}),$$

  - For each graph G_n in the mini-batch, sample a latent variable $\hat{z}_{G_n}$ for the Monte Carlo estimation in the M-step.

GraphLoG – online EM algorithm for global structure modeling



- **M-step:**
  - Maximize *the expected log-likelihood on mini-batch*: $\widetilde{Q}(\theta, \mathbf{C}) \approx \log p(\widetilde{\mathbf{G}}, \widetilde{\mathbf{Z}}_{est} | \theta, \mathbf{C})$
  - Define the likelihoods with *energy-based formulation*:

$$p(\mathcal{G}, z_{\mathcal{G}} | \theta, \mathbf{C}) = \frac{1}{Z(\theta, \mathbf{C})} \exp\big(f(h_{\mathcal{G}}, z_{\mathcal{G}})\big), \qquad f(h_{\mathcal{G}}, z_{\mathcal{G}}) = \sum_{l=1}^{L_p} s(h_{\mathcal{G}}, z_{\mathcal{G}}^l) + \sum_{l=1}^{L_p - 1} s(z_{\mathcal{G}}^l, z_{\mathcal{G}}^{l+1}).$$

  - Define objective function based on *NCE*, which contrasts the positive observed-latent variable pair with the negative pairs sampled from some noise distribution:

$$\mathcal{L}_{\text{global}} = -\mathbb{E}_{(\mathcal{G}^+, z_{\mathcal{G}}^+) \sim p(\mathcal{G}, z_{\mathcal{G}})} \Big\{ \log \tilde{p}(\mathcal{G}^+, z_{\mathcal{G}}^+ | \theta, \mathbf{C}) - \mathbb{E}_{(\mathcal{G}^-, z_{\mathcal{G}}^-) \sim p_n(\mathcal{G}, z_{\mathcal{G}})} \big[\log \tilde{p}(\mathcal{G}^-, z_{\mathcal{G}}^- | \theta, \mathbf{C}) \big] \Big\}$$

GraphLoG – Model Optimization & Downstream Application

Algorithm 1 Optimization Algorithm of GraphLoG.

Input: Unlabeled graph data set $\mathbf{G}$, the number of learning steps T.
Output: Pre-trained GNN model GNN_{θ_T}.
Pre-train GNN with local objective function (Eq. 9).
Initialize model parameters θ_0 and $\mathbf{C}_0$.
for $t = 1$ **to** T **do**
  Sample a mini-batch $\widetilde{\mathbf{G}}$ from $\mathbf{G}$.
  ◊ *E-step*:
  Sample latent variables $\widetilde{\mathbf{Z}}_{est}$ with $\text{GNN}_{\theta_{t-1}}$ and $\mathbf{C}_{t-1}$.
  ◊ *M-step*:
  Update model parameters:
    $\theta_t \leftarrow \theta_{t-1} - \nabla_\theta(\mathcal{L}_{\text{local}} + \mathcal{L}_{\text{global}})$,
    $\mathbf{C}_t \leftarrow \mathbf{C}_{t-1} - \nabla_{\mathbf{C}}(\mathcal{L}_{\text{local}} + \mathcal{L}_{\text{global}})$.
end for

- **Model optimization:**
  - Pre-train with only the local objective for one epoch.
  - Conduct E-step and M-step iteratively.
  - In the optimization of M-step, we also add the local objective function for *preserving local smoothness* when pursuing the global structure.
- **Downstream application:**
  - Pre-train a GNN by GraphLoG on *massive unlabeled graphs*.
  - Append a linear classifier and fine-tune on *a small set of labeled graphs*.

GraphLoG – Experimental Results

Table 1. Test ROC-AUC (%) on downstream molecular property prediction benchmarks.

Methods	BBBP	Tox21	ToxCast	SIDER	ClinTox	MUV	HIV	BACE	Avg
Random	65.8 ± 4.5	74.0 ± 0.8	63.4 ± 0.6	57.3 ± 1.6	58.0 ± 4.4	71.8 ± 2.5	75.3 ± 1.9	70.1 ± 5.4	67.0
EdgePred (2016)	67.3 ± 2.4	76.0 ± 0.6	64.1 ± 0.6	60.4 ± 0.7	64.1 ± 3.7	74.1 ± 2.1	76.3 ± 1.0	79.9 ± 0.9	70.3
InfoGraph (2019)	68.2 ± 0.7	75.5 ± 0.6	63.1 ± 0.3	59.4 ± 1.0	70.5 ± 1.8	75.6 ± 1.2	77.6 ± 0.4	78.9 ± 1.1	71.1
AttrMasking (2019)	64.3 ± 2.8	$\mathbf{76.7} \pm 0.4$	$\mathbf{64.2} \pm 0.5$	61.0 ± 0.7	71.8 ± 4.1	74.7 ± 1.4	77.2 ± 1.1	79.3 ± 1.6	71.1
ContextPred (2019)	68.0 ± 2.0	75.7 ± 0.7	63.9 ± 0.6	60.9 ± 0.6	65.9 ± 3.8	75.8 ± 1.7	77.3 ± 1.0	79.6 ± 1.2	70.9
GraphPartition (2020b)	70.3 ± 0.7	75.2 ± 0.4	63.2 ± 0.3	61.0 ± 0.8	64.2 ± 0.5	75.4 ± 1.7	77.1 ± 0.7	79.6 ± 1.8	70.8
GraphCL (2020a)	69.5 ± 0.5	75.4 ± 0.9	63.8 ± 0.4	60.8 ± 0.7	70.1 ± 1.9	74.5 ± 1.3	77.6 ± 0.9	78.2 ± 1.2	71.3
GraphLoG (ours)	$\mathbf{72.5} \pm 0.8$	75.7 ± 0.5	63.5 ± 0.7	$\mathbf{61.2} \pm 1.1$	$\mathbf{76.7} \pm 3.3$	$\mathbf{76.0} \pm 1.1$	$\mathbf{77.8} \pm 0.8$	$\mathbf{83.5} \pm 1.2$	**73.4**

Table 2. Test ROC-AUC (%) on downstream biological function prediction benchmark.

Methods	ROC-AUC (%)
Random	64.8 ± 1.0
EdgePred (Kipf & Welling, 2016)	70.5 ± 0.7
InfoGraph (Sun et al., 2019)	70.7 ± 0.5
AttrMasking (Hu et al., 2019)	70.5 ± 0.5
ContextPred (Hu et al., 2019)	69.9 ± 0.3
GraphPartition (You et al., 2020b)	71.0 ± 0.2
GraphCL (You et al., 2020a)	71.2 ± 0.6
GraphLoG (ours)	$\mathbf{72.9} \pm 0.7$

Table 3. Test ROC-AUC (%) of different methods under four GNN architectures. (All results are reported on biology domain.)

Methods	GCN	GraphSAGE	GAT	GIN
Random	63.2 ± 1.0	65.7 ± 1.2	68.2 ± 1.1	64.8 ± 1.0
EdgePred (2016)	68.0 ± 0.9	67.8 ± 0.7	67.9 ± 1.3	70.5 ± 0.7
AttrMasking (2019)	68.3 ± 0.8	69.2 ± 0.6	67.3 ± 0.8	70.5 ± 0.5
ContextPred (2019)	67.6 ± 0.3	69.6 ± 0.6	66.9 ± 1.2	69.9 ± 0.3
GraphCL (2020a)	69.1 ± 0.9	70.2 ± 0.4	68.4 ± 1.2	71.2 ± 0.6
GraphLoG (ours)	$\mathbf{71.2} \pm 0.6$	$\mathbf{70.8} \pm 0.8$	$\mathbf{69.5} \pm 1.0$	$\mathbf{72.9} \pm 0.7$

GraphLoG – Visualization

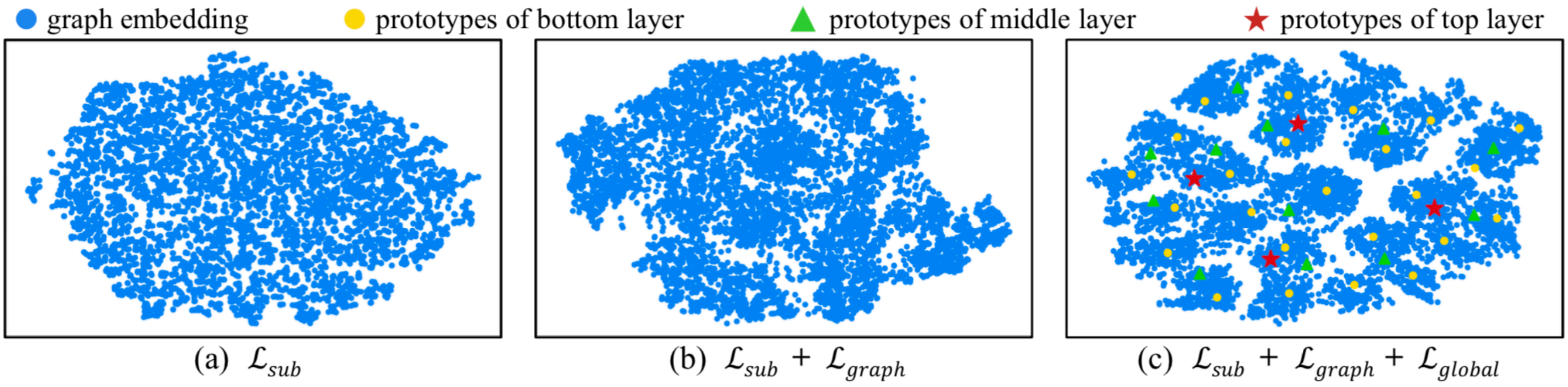


Figure 2. The t-SNE visualization on ZINC15 database (*i.e.* the pre-training data set for chemistry domain).

Thanks for watching!