

# A Survey on Self-Supervised Pre-Training of Graph Foundation Models: A Knowledge-Based Perspective

Ziwen Zhao<sup>1</sup>, Yuhua Li<sup>1,✉</sup>, Yixiong Zou<sup>1</sup>, Ruixuan Li<sup>1</sup> and Rui Zhang<sup>2,✉</sup>

<sup>1</sup>School of Computer Science and Technology, Huazhong University of Science and Technology

<sup>2</sup>www.ruizhang.info

{zwzhao, idcliyuhua, yixiongz, rxli}@hust.edu.cn, rayteam@yeah.net

## Abstract

Graph self-supervised learning is now a go-to method for pre-training graph foundation models, including graph neural networks, graph transformers, and more recent large language model (LLM)-based graph models. There is a wide variety of knowledge patterns embedded in the structure and properties of graphs which may be used for pre-training, but we lack a systematic overview of self-supervised pre-training tasks from the perspective of graph knowledge. In this paper, we comprehensively survey and analyze the pre-training tasks of graph foundation models from a knowledge-based perspective, consisting of microscopic (nodes, links, etc) and macroscopic knowledge (clusters, global structure, etc). It covers a total of 9 knowledge categories and 25 pre-training tasks, as well as various downstream task adaptation strategies. Furthermore, an extensive list of the related papers with detailed metadata is provided at <https://github.com/Newiz430/Pretext>.

## 1 Introduction

Graphs are prevalent in various real-world applications exhibiting diverse knowledge patterns [Zhang *et al.*, 2022b]. Over time, the techniques for mining graphs have evolved from network embeddings to graph neural networks (GNNs), graph Transformers, and more recent large language model (LLM)-based graph models, collectively referred to as *graph foundation models* [Liu *et al.*, 2023a]. Self-supervised learning (SSL) on graphs has emerged as a powerful approach to uncovering underlying patterns in enormous unannotated data [Kipf and Welling, 2016; Veličković *et al.*, 2019], as depicted in Figure 1. In order to reach better *task generalizability* – a crucial ability of graph foundation models to generalize to various downstream tasks, various kinds of unsupervised pre-training tasks, also referred to as *pretexts*, are designed to extract hidden supervision signals for pre-training a graph model. Afterwards, the pre-trained model is adapted to miscellaneous application scenarios, such as node classification, link prediction, and recommendation [Wang *et al.*, 2023c].

✉Corresponding authors.

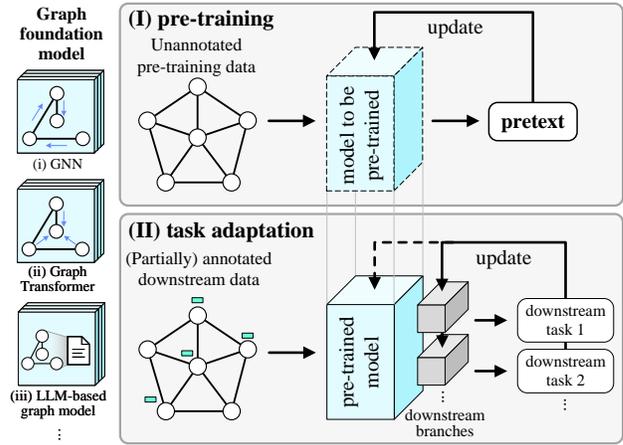


Figure 1: How graph self-supervised learning works: pre-training and task adaptation. We focus mainly on the **pre-training** and **task adaptation** of graph foundation models.

This paper presents a comprehensive survey on self-supervised pre-training strategies for graph foundation models. Our contributions are twofold. (i) *Comprehensiveness*: this is to our knowledge the first survey on self-supervised pre-training that covers all types of graph foundation models, including GNNs, graph Transformers, and LLM-based graph models, enabling a unified analysis for deeper insights. Existing surveys in this area are limited to only one type of graph models, like GNNs [Xia *et al.*, 2022c; Xie *et al.*, 2022b; Liu *et al.*, 2022b] or LLMs on graphs [Liu *et al.*, 2023a; Jin *et al.*, 2023], resulting in an incomplete and separated view which neglects the relationships between the pre-training of GNNs and LLMs. (ii) *A knowledge-based perspective*: existing surveys such as [Xie *et al.*, 2022b; Liu *et al.*, 2022b; Liu *et al.*, 2023a] broadly categorize graph SSL methods as “generative (predictive) - contrastive”. This broad categorization is insufficient to capture the unique characteristics of graphs, which have diverse knowledge patterns embedded in their structure and properties. For instance, tasks like predicting *links* require the knowledge of the local relationships between nodes, whereas tasks like predicting *clusters* require the knowledge of the distribution of nodes on the entire graph. To better analyze different types of self-supervised graph pre-training strategies, we propose a **knowledge-based taxon-**

Notation	Description
$\mathcal{G}$	Graph
$\Gamma$	Graph data space
$\mathcal{V}$	Node set of $\mathcal{G}$
$\mathcal{E}$	Edge set of $\mathcal{G}$
$\mathcal{Y}$	(Pseudo-) label space
$\mathbf{X}$	Node feature matrix of $\mathcal{G}$
$\mathbf{E}$	Edge feature matrix of $\mathcal{G}$
$\mathbf{A}$	Adjacency matrix of $\mathcal{G}$
$\mathbf{L}$	Laplacian matrix of $\mathcal{G}$
$\mathbf{M}$	Masking matrix of $\mathcal{G}$
$\mathbf{Z}$	Node representation matrix of $\mathcal{G}$
$\mathbf{H}$	Projected embedding matrix of $\mathcal{G}$
$n$	Total number of nodes ( $n =  \mathcal{V} $ )
$d$	Number of feature dimensions
$\mathcal{N}_i^k$	$k(1)$ -hop neighborhood of node $i$
$\lambda$	Balance coefficient
$\hat{\psi}$	Predicted result of $\psi$
$\psi$	Perturbed $\psi$
$\mathbf{I}_\psi$	Identity matrix with size of $\psi \times \psi$
$\mathcal{L}$	Loss function to be minimized
$\mathcal{J}$	Objective function to be maximized
$f(\dots; \Theta)$	Encoding function parameterized by $\Theta$
$g(\dots; \Psi)$	Decoding function parameterized by $\Psi$
$p(\cdot), q(\cdot)$	Probabilistic density function
$\mathbb{1}_{[ex]}$	Indicator function, 1 if $ex$ is met else 0
$\sigma(\cdot), \sigma_+(\cdot)$	Sigmoid/Softplus nonlinearity
$\circ, \oslash$	Hadamard product/division operator

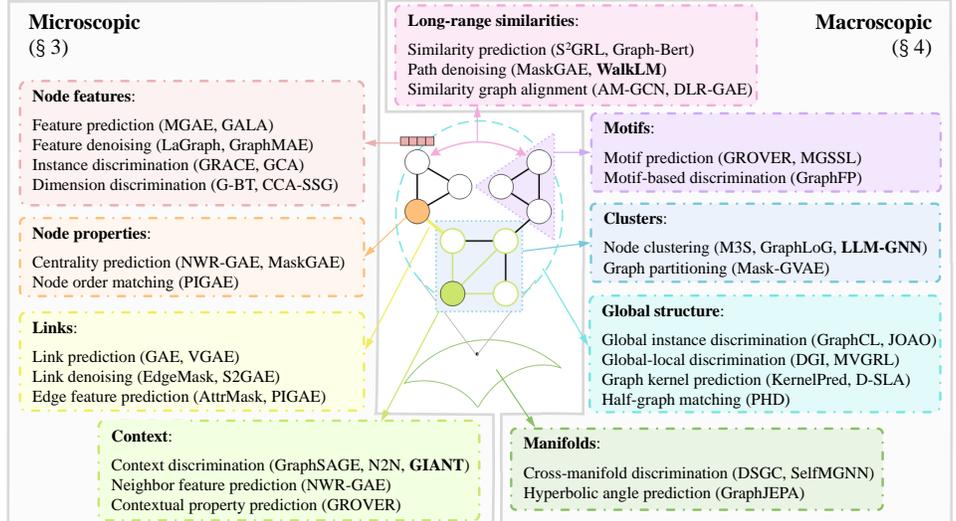


Figure 2: The knowledge-based taxonomy of pretexts with representative examples. LLM-based graph models are **bolded** and particularly discussed in Section 5.

Table 1: Notations ( $\heartsuit$  is a placeholder).

**omy** that categorizes pre-training tasks based on the types of knowledge that are utilized, as illustrated in Figure 2: *microscopic* knowledge (Section 3) focuses on node-level properties and local relationships between nodes, such as links and contextual subgraphs; *macroscopic* knowledge (Section 4) focuses on large-scale patterns that have an impact on a large portion or the entirety of a graph, such as long-range similarities and clusters. Such a knowledge-based taxonomy provides a unified perspective to analyze not only the pre-training strategies of existing graph models, but also those of the most recent LLM-based graph models (Section 5), and to explore future directions for self-supervised pre-training of graph foundation models (Section 6). It provides inspiration to combine different approaches for a more generalizable and powerful graph learner.

## 2 Definitions

This section provides definitions for some basic concepts in this field.

**Definition 1.** (Graph.) The graph is a data structure consisting of a node (vertex) set and an edge (link) set  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ . The adjacency matrix  $\mathbf{A} \in \{0, 1\}^{n \times n}$  indicates if two nodes are connected by a link. For an attributed graph, each node is associated with a row of the node feature matrix  $\mathbf{X} \in \mathbb{R}^{n \times d}$ .

**Definition 2.** (Graph foundation model.) A graph (foundation) model is a parameterized function  $f(\mathcal{G}; \Theta)$  that can be implemented by GNNs, graph Transformers, LLM-based graph models, etc. It is pre-trained on unannotated graph data in any form (numerical matrices, textualized descriptions, etc) to handle different graph-related tasks.

**Definition 3.** (Pre-training task, a.k.a. pretext.) A pretext  $\mathcal{L} \in \mathcal{T}$  is an unsupervised task performed during the pre-training phase of a graph model, which can also be viewed as a learnable initialization method.  $\mathcal{T}$  represents the task set.

**Definition 4.** (Graph self-supervised learning.) Graph self-supervised learning aims to solve the *task generalization problem*, that is, to learn a set of optimized parameters  $\Theta^*$  for a graph model by one or more pretexts. Its goal is to achieve improved performance on one or multiple downstream tasks  $\mathcal{L}_d$  with additional (relatively simple) branches  $f_d(\mathcal{G}_d; \Phi)$ :

$$\sum_{\mathcal{L}_d \in \mathcal{T}} \min_{\Phi(\cdot, \Theta^*)} \mathcal{L}_d(f_d \cdot f^*, \Gamma_d, \mathcal{Y}_d), \text{ s.t. } f^* = \sum_{\mathcal{L} \in \mathcal{T}} \arg \min_{\Theta} \mathcal{L}(f, \Gamma) \quad (1)$$

where  $\Gamma_d$  and  $\mathcal{Y}_d$  denote the downstream data and supervision information. “ $(\cdot, \Theta^*)$ ” indicates whether the pre-trained model is updated during the task adaptation.

## 3 Microscopic Pretexts

Microscopic pretexts exploit node-level properties and local relationships. This section discusses pretexts based on four types of microscopic knowledge: node features, node properties, links, and context.

### 3.1 Node Features

Node features are commonly available information in attributed graphs, often enriched with semantic knowledge, such as the text-embedded content of papers in citation networks, or atomic number and chirality of molecules in molecular networks. What is encoded in these feature vectors, and how to utilize them in pre-training graph models, depend on their source and the chosen encoding method.

**Feature prediction.** This is one of the fundamental tasks in node feature learning and has gained great popularity among graph autoencoding methods e.g. MGAE [Wang *et al.*, 2017], GALA [Park *et al.*, 2019], and Graph-Bert [Zhang *et al.*, 2020a]. These methods reconstruct the low-dimensional node representations by a parameterized decoder (a feed-forward network or a GALA-style Laplacian sharpening architecture)

Table 2: Summary of pretexts for pre-training graph foundation models. LLM-based graph models are **bolded**.

Knowledge	Pretext	Representative literature	Representative loss/objective functions
Microscopic	Feature prediction	MGAE [Wang et al., 2017], GALA [Park et al., 2019], Graph-Bert [Zhang et al., 2020a], GMI [Peng et al., 2020], WGDN [Cheng et al., 2023]	$\mathcal{L}_{MGAE} = \ell_2(\mathbf{X}) = \mathbb{E}_{i \in V} [\ \mathbf{X}_i - \tilde{\mathbf{X}}_i\ ^2]$ $\mathcal{L}_{GMI} = -\mathbb{E}_{i \in V, j \in V^+} [\sigma(-D(\mathbf{Z}_i; \mathbf{X}_j))] + \mathbb{E}_{k \in V^+} [\sigma(+D(\mathbf{Z}_i; \mathbf{X}_k))]$
	Feature denoising	AttrMask [Hu et al., 2019a; Jin et al., 2020], GraphComp [You et al., 2020b], GPT-GNN [Hu et al., 2020], LaGraph [Xie et al., 2022c], SLAPS [Fatemi et al., 2021], GraphMAE [Hou et al., 2022], GraphMAE2 [Hou et al., 2023], DDM [Yang et al., 2023], AUG-MAE [Wang et al., 2024]	$\mathcal{L}_{AttrMask} = \mathbb{E}_{q((1-M) \circ \mathbf{X})} [\ f(\mathbf{M} \circ \mathbf{X}, \mathbf{A}; \Theta) - \mathbf{X}\ ^2]$ $\mathcal{L}_{GraphMAE} = \mathbb{E}_{q((1-M) \circ \mathbf{X})} \left[ 1 - \left( \frac{\mathbf{X}^\top (M \circ \mathbf{X}, \mathbf{A}, \Theta)}{\ \mathbf{X}\  \ (M \circ \mathbf{X}, \mathbf{A}, \Theta)\ } \right)^\gamma \right]$
	Node features	GRACE [Zhu et al., 2020], GCA [Zhu et al., 2021c], BGRL [Thakoor et al., 2022], SUGRL [Mo et al., 2022], ProGCL [Xia et al., 2022a], COSTA [Zhang et al., 2022c], SpCo [Liu et al., 2022a], GRADE [Wang et al., 2022], MA-GCL [Gong et al., 2023], T-BGRL [Shiao et al., 2023a], POT [Yu et al., 2023b], Sp <sup>2</sup> GCL [Bo et al., 2023], SGCL [Sun et al., 2024]	$\mathcal{L}_{GRACE} = -\mathbb{E}_{i \in V} \left[ \log \frac{\exp(\text{sim}(\mathbf{Z}_i^+, \mathbf{Z}_i^+/\tau))}{\sum_{j \neq i} \exp(\text{sim}(\mathbf{Z}_i^+, \mathbf{Z}_j^+/\tau)) + \sum_{j=1}^n \exp(\text{sim}(\mathbf{Z}_i^+, \mathbf{Z}_j^+/\tau))} \right]$ $\mathcal{L}_{BGRL} = -\mathbb{E}_{i \in V} [\mathbf{Z}_i^+ \mathbf{H}_i^+ / \ \mathbf{Z}_i^+\  \ \mathbf{H}_i^+\ ]$ $\mathcal{L}_{SUGRL} = -\mathbb{E}_{i \in V} [\text{sim}(\mathbf{Z}_i, \mathbf{Z}_i^+)^2 - \text{sim}(\mathbf{Z}_i, \mathbf{Z}_i^-)^2 + \epsilon]$
	Instance discrimination	G-BT [Bielak et al., 2022], CCA-SSG [Zhang et al., 2021a], VICREG [Bardes et al., 2022], BlockGCL [Li et al., 2023a]	$\mathcal{L}_{G-BT} = \ \mathbf{C}^X - \mathbf{I}_d\ _F^2 = \sum_{k=1}^d (1 - C_{k,k}^X)^2 + \lambda \sum_{k=1}^d \sum_{l \neq k} C_{k,l}^{X^2}$ $\mathcal{L}_{CCA-SSG} = \ \mathbf{Z}^+ - \mathbf{Z}^-\ _F^2 + \lambda (\ \mathbf{Z}^+ \mathbf{Z}^+ - \mathbf{I}_d\ _F^2 + \ \mathbf{Z}^+ \mathbf{Z}^- - \mathbf{I}_d\ _F^2)$
	Dimension discrimination	ScoreRank [Hu et al., 2019b], NodeProperty [Jin et al., 2020], NWR-GAE [Tang et al., 2022], MaskGAE [Li et al., 2023b]	$\mathcal{L}_{NodeProperty} = \ell_2(s) = \mathbb{E}_{i \in V} [\ s(i) - \hat{s}(i)\ ^2]$ , $s(\cdot)$ is centrality function $\mathcal{L}_{ScoreRank} = -\mathbb{E}_{i,j \in V} [\log \mathbb{1}_{\{s(i) > s(j)\}}] + \mathbb{E}_{i,j \in V} [\log \mathbb{1}_{\{s(i) \leq s(j)\}}]$
	Node order matching	PIGAE [Winter et al., 2021]	$\mathcal{L}_{PIGAE} = \mathbb{E}_{q_0(\mathbf{Z} \mathcal{G}_\pi)} [\log p_\theta(\mathcal{G}_\pi   \mathbf{Z}, \hat{\mathbf{P}}_{\pi \rightarrow \pi^*} \mathcal{G}_\pi)] - D_{KL}[q_0(\mathbf{Z} \mathcal{G}_\pi) \  p_\theta(\mathbf{Z})]$
	Centrality prediction	GAE [Kipf and Welling, 2016], VGAE [Kipf and Welling, 2016], ARGAE [Pan et al., 2018], ARVGA [Pan et al., 2018], Graphite [Grover et al., 2019], SIG-VAE [Hasanzadeh et al., 2019], SuperGAT [Kim and Oh, 2021], CSSL [Li et al., 2022b]	$\mathcal{L}_{GAE} = CE(\mathbf{A}) = -\mathbb{E}_{(i,j) \in E} [\log \hat{A}_{i,j}] + \mathbb{E}_{(i,j) \in \bar{E}} [-\log(1 - \hat{A}_{i,j})]$ $\mathcal{L}_{VGAE} = \mathbb{E}_{q_0(\mathbf{Z} \mathbf{X}, \mathbf{A})} [\log p_\theta(\mathbf{A} \mathbf{Z})] - D_{KL}[q_0(\mathbf{Z} \mathbf{X}, \mathbf{A}) \  p_\theta(\mathbf{Z})]$
	Link prediction	DenoisingRecon [Hu et al., 2019b], EdgeMask [Jin et al., 2020], GPT-GNN [Hu et al., 2020], S2GAE [Tan et al., 2023a], HGMAE [Tian et al., 2023], SeeGera [Li et al., 2023c], Bandana [Zhao et al., 2024b]	$\mathcal{L}_{EdgeMask} = -\mathbb{E}_{(M \circ \mathbf{A})_{i,j}=1} [\log \hat{A}_{i,j}] + \mathbb{E}_{A_{i,j}=0} [\log(1 - \hat{A}_{i,j})]$
	Link denoising	AttrMask [Hu et al., 2019a], PIGAE [Winter et al., 2021], ASD-VAE [Jiang et al., 2024]	$\mathcal{L}_{AttrMaskEdge} = \ell_2(\mathbf{E}) = \mathbb{E}_{(i,j) \in E} [\ \mathbf{E}_i - \hat{\mathbf{E}}_i\ ^2]$
	Edge feature prediction	GraphSAGE [Hamilton et al., 2017], ContextPred [Hu et al., 2019a], GCC [Qiu et al., 2020], Sub-Con [Jiao et al., 2020], Graph-MLP [Hu et al., 2021], COLES [Zhu et al., 2021a], GLEN [Zhu and Koniusz, 2022], N2N [Dong et al., 2022], AFGR [Lee et al., 2022], <b>GIANT</b> [Chien et al., 2022], S <sup>2</sup> -CL [Ding et al., 2023], HGRL [Chen et al., 2022], NeCo [He et al., 2023], <b>Instruct-GLM</b> [Ye et al., 2023], <b>GraphGPT</b> [Tang et al., 2023]	$\mathcal{L}_{GraphSAGE} = -\mathbb{E}_{i \in V, j \in \mathcal{N}_i} [\log \sigma(\mathbf{Z}_i^\top \mathbf{Z}_j) + \lambda \sum_{k \in V} \log \sigma(-\mathbf{Z}_i^\top \mathbf{Z}_k)]$ $\mathcal{L}_{COLES} = \text{tr}(\mathbf{Z}^\top \mathbf{L} \mathbf{Z}) + \lambda \sum_{i,j} \text{tr}(\mathbf{Z}_i \mathbf{L}_i \mathbf{Z}_j)$ , $s.t. \mathbf{Z}^\top \mathbf{Z} = \mathbf{I}_d$ $\mathcal{L}_{N2N} = -\mathbb{E}_{i \in V} [\log \frac{\exp(\text{sim}(\mathbf{Z}_i, \mathbf{Z}_i \mathbf{L}_i \mathbf{Z}_i))}{\sum_{k \in V} \exp(\text{sim}(\mathbf{Z}_i, \mathbf{Z}_k))}]$
Context discrimination	Neighbor feature prediction	NWR-GAE [Tang et al., 2022]	$\mathcal{L}_{NWR-GAE} = W_2^2(p(\mathcal{N}_i), g_p(\mathbf{Z}; \Psi))$ , $W_2^2(\cdot, \cdot)$ is 2-Wasserstein distance
Contextual property prediction	GROVER [Rong et al., 2020]	$\mathcal{L}_{GROVER} = CE(\mathcal{J}_{ContextProp})$	
Macroscopic	Similarity prediction	Graph-Bert [Zhang et al., 2020a], PairwiseDistance [Jin et al., 2020], PairwiseAttrSim [Jin et al., 2020], S <sup>2</sup> GRL [Peng et al., 2022], AGE [Cui et al., 2020]	$\mathcal{L}_{S^2GRL} = -\mathbb{E}_{i,j \in V} [S_{i,j} \cdot \log g(\Psi; \mathbf{Z})_{i,j}]$ , $S_{i,j}$ is shortest path distance
	Path denoising	MaskGAE [Li et al., 2023b], <b>WalkLM</b> [Tan et al., 2023b]	$\mathcal{L}_{MaskGAE} = \mathcal{L}_{EdgeMask}$ , $\mathbf{M}$ is path masking matrix
	Similarity graph alignment	AM-GCN [Wang et al., 2020], DLR-GAE [Chen et al., 2023b], ASP [Chen and Kou, 2023], MVMI-FT [Fan et al., 2023], AEGCL [Li et al., 2023d]	$\mathcal{L}_{AM-GCN} = \ \mathbf{S} - \mathbf{S}_F\ _F^2$ , $\mathbf{S}_F$ is dot product of feature graph embeddings
	Motif prediction	GROVER [Rong et al., 2020], MGSSL [Zhang et al., 2021c], GraphFP [Luong and Singh, 2023], MoAMa [Inac et al., 2023], DGPM [Yan et al., 2024]	$\mathcal{L}_{MGSSL} = CE(\mathcal{J}_{motif})$
	Motif-based discrimination	MICRO-Graph [Zhang et al., 2021b], GraphFP [Luong and Singh, 2023]	$\mathcal{L}_{GraphFP} = -\mathbb{E}_{i \in V, \mathcal{F} \in \mathcal{F}} [\log \frac{\exp(\text{sim}(\mathbf{Z}_i, \mathbf{Z}_i^{\mathcal{F}}/\tau))}{\sum_{j \neq i} \exp(\text{sim}(\mathbf{Z}_i, \mathbf{Z}_j^{\mathcal{F}}/\tau))}]$ , $\mathcal{F}$ is set of fragment graphs
	Node clustering	NodeCluster [Hu et al., 2019b; You et al., 2020b], M3S [Sun et al., 2020b], CommDGI [Zhang et al., 2020b], GraphLoG [Xu et al., 2021b], HomoGCL [Li et al., 2023c], CARL-G [Shiao et al., 2023b], <b>LLM-GNN</b> [Chen et al., 2024]	$\mathcal{L}_{NodeCluster} = -\mathbb{E}_{i \in V, c \in \mathcal{C}} [C_{i,c} \log \hat{C}_{i,c}]$ , $\mathcal{C}$ is family of node clusters
	Graph partitioning	GraphPar [Hu et al., 2019b; You et al., 2020b], Distance2Clusters [Jin et al., 2020], SHGP [Yang et al., 2022b], DGVAE [Li et al., 2020], Mask-GVAE [Li et al., 2021a], gCool [Li et al., 2022a], CSGCL [Chen et al., 2023a], Struct-Comp [Zhang et al., 2024]	$\mathcal{L}_{Mask-GVAE} = \frac{1}{ \mathcal{C} } \text{tr}(\mathbf{M}_c^\top \mathbf{L} \mathbf{M}_c) \odot (\mathbf{M}_c^\top (\mathbf{L} - \mathbf{A}) \mathbf{M}_c) + \lambda \left\  \frac{ \mathcal{C} }{n} \mathbf{M}_c^\top \mathbf{M}_c - \mathbf{I}_{ \mathcal{C} } \right\ _F^2$
	Global instance discrimination	GraphCL [You et al., 2020a], JOAO [You et al., 2021], AD-GCL [Suresh et al., 2021], GASSL [Yang et al., 2021], GraphCL-LP [You et al., 2022], Sim-GRACE [Xia et al., 2022b], GroupCL [Xu et al., 2022b]	$\mathcal{L}_{GraphCL} = -\mathbb{E}_{\mathcal{G} \in \Gamma} [\log \frac{\exp(\text{sim}(\mathbf{Z}_i^{\mathcal{G}}, \mathbf{Z}_i^{\mathcal{G}}/\tau))}{\sum_{\mathcal{H} \in \Gamma, \mathcal{H} \neq \mathcal{G}} \exp(\text{sim}(\mathbf{Z}_i^{\mathcal{G}}, \mathbf{Z}_i^{\mathcal{H}}/\tau))}]$
	Global-local discrimination	DGI [Veličković et al., 2019], InfoGraph [Sun et al., 2020a], MVGRL [Hassani and Khasanmadi, 2020], InfoGCL [Xu et al., 2021a], GGD [Zheng et al., 2022], D-SLA [Kim et al., 2022], SPAN [Lin et al., 2023]	$\mathcal{L}_{DGI} = \mathbb{E}_{i \in V} [\log D(\mathbf{Z}_i, \mathcal{Z}_i) - \log D(\tilde{\mathbf{Z}}_i, \mathcal{Z}_i)]$
	Graph kernel prediction	KernelPred [Navarin et al., 2018], D-SLA [Kim et al., 2022]	$\mathcal{L}_{D-SLA} = \mathbb{E}_{\mathcal{G}^t, \mathcal{G}^k \in \Gamma, \mathcal{G}^h \in \Gamma} \left[ \left( \frac{g(\mathbf{Z}, \mathbf{Z}^t; \Psi) - g(\mathbf{Z}, \mathbf{Z}^k; \Psi)}{\text{ker}(\mathbf{Z}, \mathbf{Z}^t)} \right)^2 + \lambda \max(0, g(\mathbf{Z}, \mathbf{Z}^t; \Psi) - g(\mathbf{Z}, \mathbf{Z}^k; \Psi) + \epsilon) \right]$
Half-graph matching	PHD [Li et al., 2021b]	$\mathcal{L}_{PHD} = -\sum_{\mathcal{G}^t, \mathcal{G}^k \in \Gamma} [\mathbb{1}_{\{i=j\}} \log \mathbf{Z}^{i \oplus j} + \mathbb{1}_{\{i \neq j\}} \log(1 - \mathbf{Z}^{i \oplus j})]$ , $\oplus$ is half-graph connection	
Cross-manifold discrimination	HGCL [Liu et al., 2021], DSGC [Yang et al., 2022a], SelfMGNN [Sun et al., 2022a]	$\mathcal{L}_{DSGC} = -\mathbb{E}_{i \in V} [\log \frac{\exp(\text{sim}(\mathbf{Z}_i^{\mathbb{H}}, \mathbf{Z}_i^{\mathbb{H}}/\tau))}{\sum_{j=1}^n \exp(\text{sim}(\mathbf{Z}_i^{\mathbb{H}}, \mathbf{Z}_j^{\mathbb{H}}/\tau))}]$ , $\mathbb{H}$ is hyperbolic space	
Manifolds	Hyperbolic angle prediction	GraphJEPa [Skenderi et al., 2023]	$\mathcal{L}_{GraphJEPa} = \text{smooth-}\ell_1(\psi)$ , $\psi_i = (\sinh(\text{avgpool}(\mathbf{Z}_i))) \cdot \cosh(\text{avgpool}(\mathbf{Z}_i))^\top$

and match them with the original feature size. Then, an  $\ell_2$  regression loss is optimized between the predicted features and the ground truth. GMI [Peng et al., 2020] employs a discriminator to maximize the Jensen-Shannon divergence between the representations and original features instead. Feature prediction has been shown beneficial for tasks in the node embedding space, e.g. node clustering [Park et al., 2019].

**Feature denoising.** Derived from the traditional denoising autoencoder [Vincent et al., 2008], this pretext first adds noise to node features:  $\tilde{\mathbf{X}} = \mathbf{X} + \epsilon$ , and learns how to denoise the original data. Traditionally, the noise  $\epsilon$  is often sampled from a continuous distribution (e.g. isotropic Gaussian). Recent years have witnessed a more popular denoising task called *masked feature prediction* (a.k.a. masked autoencoding or graph completion), derived from BERT [De-

vlin et al., 2019] and MAE [He et al., 2022]. Research such as AttrMask [Hu et al., 2019a; Jin et al., 2020], LaGraph [Xie et al., 2022c], and SLAPS [Fatemi et al., 2021] samples a set of binary noise from a discrete Bernoulli distribution to create a masking matrix  $\mathbf{M} \in \{0, 1\}^{n \times d}$ . It is used to mask a portion of features by Hadamard product:  $\tilde{\mathbf{X}} = \mathbf{M} \circ \mathbf{X}$ . The perturbed features are then fed into the graph model to predict the masked features. Node feature masking can be performed on all dimensions of some random nodes ( $\mathbf{M}_{i,:} \in \{0\}^d \cup \{1\}^d$ ), some random dimensions of all nodes ( $\mathbf{M}_{:,j} \in \{0\}^n \cup \{1\}^n$ ), or random dimensions of random nodes. Apart from the  $\ell_2$  distance between latent representation vectors, GraphMAE [Hou et al., 2022; Hou et al., 2023] prefers the scaled cosine error, incorporating an exponential focusing parameter to adjust the weight of

each sample.

Unlike masked autoencoding, GPT-GNN [Hu *et al.*, 2020] adopts *autoregressive feature denoising* by a generative pre-training framework. A subset of nodes and edges are first masked to create an initial graph. Then, the masked node attributes and their corresponding edges are generated one by one, during which the  $\ell_2$  loss between the generated and real features is calculated.

With the rapid development of denoising diffusion probabilistic models (DDPMs) [Ho *et al.*, 2020] in the generation field, some studies have pointed out their resemblance to the traditional feature denoising process, considering that the diffusion network is essentially a multi-step denoising autoencoder [Xiang *et al.*, 2023]. Therefore, DDPMs have the potential to obtain highly generalizable self-supervised representations by performing feature denoising. DDM [Yang *et al.*, 2023] is the first work to pre-train a graph diffusion network for discriminative tasks such as node/graph classification, which outperforms masked feature prediction methods by denoising node features in an anisotropic feature space.

**Instance discrimination.** Instance discrimination has achieved significant success in the visual domain [He *et al.*, 2020; Chen *et al.*, 2020] which stimulates the research on graphs and subsequently becomes a fundamental and general pretext, often referred to as (node-level) “graph contrastive learning”. The goal is to distinguish pairs of node instances by learning their relative similarity. Instance discrimination methods share a similar workflow: they start with one or two perturbed versions of an original graph  $\mathcal{G}^i, \mathcal{G}^{ii}$ , referred to as “views”, through one or more graph augmentation schemes. The underlying semantics of two views are generally considered similar since they are derived from the same graph. Therefore, Nodes at the same position of both views ( $\mathbf{Z}_i^i, \mathbf{Z}_i^{ii}$ ) are considered a positive pair, while the others (in different views ( $\mathbf{Z}_i^i, \mathbf{Z}_j^{ii}$ ) or the same view ( $\mathbf{Z}_i^i, \mathbf{Z}_j^i$ )) are randomly sampled as negative pairs. The objective of instance discrimination is to “pull positive samples closer and push negative samples away”.

One simple implementation is *latent feature matching* [Zhang *et al.*, 2021a; Xie *et al.*, 2022c] that pulls positive samples’ representations closer by minimizing their Euclidean distance. However, latent feature matching is rather simple and suffers from the degeneration problem, i.e. the output of the encoder will degenerate to a scalar regardless of the input, so it is often used as an auxiliary objective. A more comprehensive definition of relative similarity is the mutual information (MI), which captures the statistical dependence between two random variables:

$$I(\mathbf{Z}_i^i; \mathbf{Z}_j^{ii}) = D_{\text{KL}}[p(\mathbf{Z}_i^i, \mathbf{Z}_j^{ii}) \| p(\mathbf{Z}_i^i)p(\mathbf{Z}_j^{ii})] \quad (2)$$

There are various options for the estimation of MI, including those that resort to sampling negative instances: (i) Jenson-Shannon (JS) estimator, rather uncommon in node-level contrast; (ii) InfoNCE estimator [Oord *et al.*, 2018] (including the NT-Xent loss [Chen *et al.*, 2020]), well-known in representative contrastive models like GRACE [Zhu *et al.*, 2020], GCA [Zhu *et al.*, 2021c], and ProGCL [Xia *et al.*, 2022a]; (iii) triplet margin estimator, such as SUGRL [Mo *et al.*, 2022]. In particular, the bootstrapping estimator in

BGRL [Thakoor *et al.*, 2022], InfoGCL [Xu *et al.*, 2021a], and SGCL [Sun *et al.*, 2024] does not resort to negative sampling, thus it serves as an efficient approach of instance discrimination also known as “non-contrastive learning”.

Contrary to feature prediction, instance discrimination essentially performs metric learning in the latent space, which encourages abandoning shallow patterns and focusing on deeper semantic agreement between sample pairs [Zhu *et al.*, 2021c]. Therefore, instance discrimination is more generalizable than feature prediction. Numerous studies based on these MI estimators have sprung up in recent years which bring improvements to data augmentations [Zhu *et al.*, 2021c; Liu *et al.*, 2022a; Gong *et al.*, 2023], architectures [Jin *et al.*, 2021; Zhang *et al.*, 2022c], negative sampling strategies [Xia *et al.*, 2022a], training fairness [Wang *et al.*, 2022; Yu *et al.*, 2023b], and more.

We will also discuss several variants of instance discrimination tasks hereinafter, including discrimination on the contextual and global scale (Subsection 3.4 and 4.4) or based on motifs and manifolds (Subsection 4.2 and 4.5).

**Dimension discrimination.** From another perspective, this pretext aims to explicitly distinguish different dimensions of node representations, as different dimensions are considered independent latent factors that encode different knowledge. Similar to instance discrimination, two views are first generated by perturbations. Then, the same dimensions of both views are considered a positive pair and vice versa. Such dimensional correlations across two views form a batch-normalized cross-correlation matrix  $\mathbf{C}^\times \in \mathbb{R}^{d \times d}$ . The goal of dimension discrimination is to decorrelate the dimensions, i.e.  $\mathbf{C}^\times$  should be close to an identity matrix  $\mathbf{I}_d$ . This strategy is initially proposed by Barlow Twins [Zbontar *et al.*, 2021] and then introduced to the graph domain by G-BT [Bielak *et al.*, 2022], which considers the similarity between dimensions of different instances. On the other hand, CCA-SSG [Zhang *et al.*, 2021a] focuses on the dimensional similarity within a single instance, and performs latent feature matching as an auxiliary task. As representations in CCA-SSG should be normalized, VICReg [Bardes *et al.*, 2022] substitutes a variance-preservation term for the normalization instead.

**Discussion.** Currently, two popular node feature-based pretexts for pre-training graph foundation models are feature prediction and instance discrimination. However, their limitations remain challenging as follows. (i) For feature prediction, traditional autoencoding frameworks resort to under-complete architectures (the latent dimensionality is smaller than the input) or additional regularization schemes [Wang *et al.*, 2017] to avoid degeneration into identity mapping, resulting in limited capacity. Although feature denoising has circumvented this problem, noise addition must be carefully balanced to avoid trivial learning as well as semantic damage. Moreover, predicting the input space encourages the preservation of shallow representations rather than deeper, generalizable semantics. (ii) For instance discrimination, relying solely on the MI of nodes is insufficient to effectively capture the inherent knowledge of graph structure, leading to sub-optimal generalizability when it comes to structural downstream tasks (e.g. link prediction), as the actual distributions

of node relations are conditioned on the graph structure.

### 3.2 Node Properties

Different from node features, node properties are structure-related information, shedding light on the structural role of a node. For example, the degree of a node signifies its local connectivity, and the cluster coefficient measures the gathering tendency of node groups. As node properties can be directly obtained from the given structure, various property-based pretexts have been designed.

**Centrality prediction.** Node centrality encompasses a wide range of measures, including degree, closeness centrality, harmonic centrality, graph centrality, betweenness centrality, eigenvector centrality, etc., reflecting the topological significance of a node from different perspectives. Autoencoding methods like NWR-GAE [Tang *et al.*, 2022] and MaskGAE [Li *et al.*, 2023b] employ an  $\ell_2$  loss to predict the degree of each node. [Hu *et al.*, 2019b] proposes *centrality ranking*, a binary classification task predicting if a node has a higher or lower centrality score  $s$  compared to another node.

**Node order matching.** PIGAE [Winter *et al.*, 2021] first explores the generalizability of node order matching by incorporating a learnable permuter in a variational autoencoder. The permuter assists in aligning the output representation order  $\pi'$  with the input node order  $\pi$  by predicting a node permutation matrix  $\mathbf{P}_{\pi \rightarrow \pi'}$  indicating the corresponding output node index of every input node.

**Discussion.** Despite the structural compensation provided by node properties, there are two main drawbacks: (i) Node properties tend to be more task-specific [Jin *et al.*, 2020], limiting their task generalizability. (ii) Node properties may lack sufficient discriminability. For example, different graph topologies can exhibit the same degree distribution. Efforts should be made to exploit more expressive properties.

### 3.3 Links

Links play a fundamental role in graphs as they represent the relationships between pairs of nodes. The nature and significance of knowledge carried by links vary depending on the specific graph type. For example, links carry information about the type and strength of chemical bonds between atoms in molecular graphs, which is vital for analyzing the physicochemical properties of a molecule. Learning different kinds of links lays the foundation of structure-based SSL.

**Link prediction.** Aiming to encode and predict the adjacency matrix  $\mathbf{A}$ , link prediction serves as a natural SSL scheme for graph models [Wan *et al.*, 2021; Kim and Oh, 2021]. Similar to node feature prediction, link prediction is often realized through autoencoders. A typical example is GAE [Kipf and Welling, 2016], which feeds the learned node representations into a dot-product decoder to compute the existence probability between a pair of nodes ( $\hat{\mathbf{A}} = \sigma(\mathbf{Z}\mathbf{Z}^\top)$ ) and optimizes a binary cross-entropy loss. Variational methods are also commonly employed for predicting links. VGAE [Kipf and Welling, 2016] learns an approximate latent distribution  $q_\phi(\mathbf{Z}|\mathbf{X}, \mathbf{A})$  typically consisting of two Gaussian variables, mean and variance. Representations are then sampled from these distributions to

maximize the expected log-likelihood of the adjacency matrix  $\log p_\theta(\mathbf{A})$ , bounded by ELBO. The following work like ARVGA [Pan *et al.*, 2018] and SIG-VAE [Hasanzadeh *et al.*, 2019] are built upon the aforementioned approaches with various improvements. SELAR [Hwang *et al.*, 2020] presents *meta-path prediction*, another version of link prediction exclusively for heterogeneous graphs.

**Link denoising.** Similar to feature denoising, link denoising randomly masks a portion of edges and predicts their existence. In the vast majority of cases, noises on links appear in binary masks and the denoising process turns into *masked link prediction*. The objective is similar to binary link prediction, but only the masked edges are considered positive. EdgeMask [Jin *et al.*, 2020], S2GAE [Tan *et al.*, 2023a], and MaskGAE [Li *et al.*, 2023b] prefer a learnable decoder  $\hat{\mathbf{A}} = \sigma(g(\mathbf{Z}\mathbf{Z}^\top; \Psi))$  for expressive performance. GPT-GNN [Hu *et al.*, 2020] also proposes an autoregressive version that generates and predicts links one by one. HGMAE [Tian *et al.*, 2023] presents *masked meta-path prediction* as a heterogeneous version of masked link prediction. Recently, Bandana [Zhao *et al.*, 2024b] makes an exception that adds continuous noises on the entire edge set, differentiating itself from other binary masking methods by preserving the integrity of both global and local structures.

**Edge feature prediction.** Edge features usually encode additional information related to the node relationships, such as the co-authored paper information of two authors. Methods for learning node features are mostly suitable for edge features too. For example, PIGAE [Winter *et al.*, 2021] and ASD-VAE [Jiang *et al.*, 2024] perform edge feature prediction by a variational framework, while AttrMask [Hu *et al.*, 2019a] performs masking and reconstruction on both links and their features.

**Discussion.** A common viewpoint towards link prediction as a pretext is that it over-emphasizes the structural information, leading to insufficient generalization to non-link prediction tasks [Veličković *et al.*, 2019; Hou *et al.*, 2022]. To address this limitation, there is an urgent need for further exploring the synergy between feature and structure-based pretexts.

### 3.4 Context

Neighborhoods are core storage units of local structural knowledge, as the message-passing GNNs propagate messages neighborhood-wise. Neighborhoods are a subset of graph context which may sometimes encompass a broader scope. The majority of context learning methods capitalize on the *homophily assumption* of graphs: representations of adjacent nodes should be similar and vice versa.

**Context discrimination.** Context learning can be traced back to network embedding algorithms e.g. DeepWalk [Perozzi *et al.*, 2014], which sample the sequences from the graph using random walks and then iteratively update their embeddings using text embedding methods. However, most of them are only applicable to the transductive scenario. GraphSAGE [Hamilton *et al.*, 2017] expands them to inductive settings through a GNN-based framework and random sampling on the  $k$ -hop neighborhood, which redefines “context” from

sequence-based to structure-based. GraphSAGE optimizes a JS estimator with a non-parameterized discriminator between  $Z_i$  and its contextual nodes  $Z_j, j \in \mathcal{N}_i$ . It is essentially a neighborhood-wise instance discrimination task, where the central node and its contextual nodes are treated as positive pairs and other nodes as negative pairs. Later efforts [Zhu *et al.*, 2021b; Zhao *et al.*, 2023] utilize a parameterized discriminator to determine whether one node is the neighbor of another node. Departing from JS divergence, COLES [Zhu *et al.*, 2021a] captures neighborhood similarity by equipping Laplacian Eigenmaps with negative sampling, which is further generalized by GLEN [Zhu and Koniusz, 2022] as a rank optimization problem of representation scatter matrices.

For other instance discrimination targets like InfoNCE, neighborhood contrastive methods such as Graph-MLP [Hu *et al.*, 2021] and N2N [Dong *et al.*, 2022] define their positive samples as every node and its aggregated neighborhood representation. Subg-Con [Jiao *et al.*, 2020] selects  $k$ -nearest neighbors by personalized PageRank scores as positive samples of a triplet loss. AFGRL [Lee *et al.*, 2022] selects  $k$ -nearest neighbors in the context of both structure and feature as positive samples of a bootstrapping estimator. Some research [Chen *et al.*, 2022; He *et al.*, 2023] further extends the utilization of the homophily assumption by selecting homophilic neighbors as more precise positive samples.

Differing from the node-level discrimination tasks above, ContextPred [Hu *et al.*, 2019a] presents *contextual subgraph discrimination* based on their aggregated representations. It first samples a “context graph” for each node containing the structure around its neighborhood. Then, the neighborhood subgraph and context graph sharing the same central node are matched as a positive pair and vice versa. GCC [Qiu *et al.*, 2020] induces two subgraphs from the  $k$ -hop neighborhood of each node as a positive pair instead. S<sup>3</sup>-CL [Ding *et al.*, 2023] performs contrastive learning between intermediate representations of different layers to aggregate neighborhoods of varying scales.

**Neighbor feature prediction.** This challenging pretext is built upon node feature prediction and tries to reconstruct the feature set of  $k$ -hop neighbors. NWR-GAE [Tang *et al.*, 2022] learns the neighborhood distribution by a parameterized decoder and optimizes the 2-Wasserstein distance between the predicted and real neighborhood distributions.

**Contextual property prediction.** GROVER [Rong *et al.*, 2020] leverages the node/edge properties in molecular graphs to define the properties of contextual subgraphs. For example, if a one-hop subgraph consists of three atoms (O,C,N) along with a C-O and a C=N bond, they are combined as a subgraph label (O-C=N) for a multi-class classification task.

**Discussion.** It is empirically verified that some contextual learning pretexts have limited contributions to SSL performance [Jin *et al.*, 2020]. This is due to the inherent capability of message-passing to extract local structural information. As a result, pretexts are suggested to put more emphasis on uncovering macroscopic knowledge of the graph.

## 4 Macroscopic Pretexts

Macroscopic pretexts focus on global information that has an impact on a large portion or the entirety of a graph. This section discusses pretexts based on five types of macroscopic information: long-range similarities, motifs, clusters, global structure, and graph manifolds.

### 4.1 Long-Range Similarities

Similarities between non-adjacent nodes are crucial for understanding the global structure of a graph. For example, the small-world property in social networks is one of the semantic attributes of long-range relationships. There are two types of similarities between non-adjacent nodes that can be learned: (i) topologically accessible similarities measured along the *paths*; (ii) inaccessible ones measured by the relative distance of features. While instance discrimination methods learn pairwise similarity between views, they often overlook the long-range dependency between non-adjacent nodes and treat them all as negative samples. Thus, it is crucial to study long-range similarity-based pretexts.

**Similarity prediction.** This pretext calculates and predicts a similarity matrix  $S \in \mathbb{R}^{n \times n}$  between nodes. S<sup>2</sup>GRL [Peng *et al.*, 2022] and PairwiseDistance [Jin *et al.*, 2020] consider the shortest path distance (a.k.a. hop count) between two nodes as the prediction target and optimizes the negative log-likelihood loss. For other similarity measures, GraphBert [Zhang *et al.*, 2020a] defines  $S$  as a PageRank or Jaccard’s coefficient matrix and performs regression by an  $\ell_2$  loss. AGE [Cui *et al.*, 2020] and PairwiseAttrSim [Jin *et al.*, 2020] select a batch of node pairs with the highest and lowest cosine similarity as positive and negative samples, respectively.

**Path denoising.** MaskGAE [Li *et al.*, 2023b] presents a variant of masked link prediction, which utilizes path masking through random walks instead of individual link masking to encourage the exploitation of the structural dependencies.

**Similarity graph alignment.** A similarity graph includes an alternative adjacency matrix of  $\mathcal{G}$  that is built based on pairwise distances of nodes. AM-GCN [Wang *et al.*, 2020] and DLR-GAE [Chen *et al.*, 2023b] aim to learn the commonality between the original and similarity graph by  $\ell_2$  and cross-entropy respectively. Such commonalities are crucial for establishing connections between the feature space and the topological space. Moreover, some contrastive methods [Chen and Kou, 2023; Fan *et al.*, 2023; Li *et al.*, 2023d] align two versions of similarity graphs by setting them as different views.

**Discussion.** Node similarities are also influenced by their neighborhoods for message-passing-based methods. However, nodes with similar features do not necessarily have similar neighborhoods, which should be considered by long-range similarity learning methods.

### 4.2 Motifs

Motifs are subgraphs with a simple structure that appear frequently in various networks. They usually carry specific properties, such as functional groups in molecular graphs,

coregulators in regulatory networks, and cliques of people in social networks. While motif discovery has been a subject of research for decades, self-supervised motif learning methods have emerged in recent years.

**Motif prediction.** As motifs can be recognized by unsupervised algorithms beforehand, GROVER [Rong *et al.*, 2020] assigns a motif pseudo-label to each molecular graph and trains a GNN to classify the instances, and MoAMa [Inae *et al.*, 2023] conducts motif-wise feature masking and prediction. Following methods prefer to build a “fragment graph” in which nodes are supernodes aggregated from subgraphs of the original graph, each including a specific motif. Meanwhile, the aggregated supernode representations are collected as a motif dictionary. In this way, motif prediction is transformed into a node feature matching task: the representation vector of each node corresponds to an entry in the motif dictionary. MGSSL [Zhang *et al.*, 2021c] proposed an autoregressive method to generate and classify the supernodes one by one, while GraphFP [Luong and Singh, 2023] directly performs a multi-label classification on the whole graph. Meanwhile, GraphFP utilizes another structural prediction task, that is to assign a unique pseudo-label to each connection backbone of the fragment graph and predict them, as a structural compensation for the feature-based objective.

**Motif-based discrimination.** This discrimination task also resorts to the fragment graph to create contrastive instances. MICRO-Graph [Zhang *et al.*, 2021b] and GraphFP [Luong and Singh, 2023] match the original and fragment graph embeddings as a positive sample of the InfoNCE objective.

**Discussion.** Most motif learning pretexts are designed specifically for molecular graphs. However, we should not overlook the motif information in large-scale networks as well. Additionally, motif dictionaries incur extra memory overhead due to the diverse range of motifs. It is worth researching how to further condense and extract the common knowledge from different motifs.

### 4.3 Clusters

Clusters indicate a higher degree of internal connection among nodes compared to the external. The connections can be defined by either node feature similarities or links, the latter are also known as “communities”.

**Node clustering.** First introduced by M3S [Sun *et al.*, 2020b], feature-based clustering leverages the unsupervised clustering algorithms to pseudo-label every node and train a GNN for node classification. [Hu *et al.*, 2019b; You *et al.*, 2020b] follow the idea and predict a one-hot cluster indicator matrix by an encoder-decoder architecture. HomoGCL [Li *et al.*, 2023c] utilizes *soft node clustering* based on a Gaussian Mixture Model and optimizes an  $\ell_2$  loss of the cluster assignment probabilities between the two ends of an edge. CARL-G [Shiao *et al.*, 2023b] evaluates the  $\ell_1$  error of Cluster Validation Indices, a family of clustering properties estimating the compactness or separation of node clusters. Another common pretext is *cluster-based instance discrimination* [Zhang *et al.*, 2020b; Ding *et al.*, 2023; Li *et al.*, 2023c] which learns cluster-aware latent feature distances by contrasting node embeddings with several learnable

cluster prototypes. Taking into account the hierarchical nature of clustering, GraphLoG [Xu *et al.*, 2021b] models the clustering hierarchy by setting prototypes at different levels and organizing them in a tree structure.

**Graph partitioning.** This is also called “non-overlapping community detection” in certain contexts. Unlike node clustering, graph partitioning is based on structural clustering patterns and thus is available to unattributed graphs. [Hu *et al.*, 2019b; You *et al.*, 2020b; Jin *et al.*, 2020] leverage graph partitioning methods to predict the cluster membership of a node. SHGP [Yang *et al.*, 2022b] further adapts this strategy to heterogeneous graph partitioning. DGVAE [Li *et al.*, 2020] employs the Dirichlet distribution as a prior for the latent cluster memberships in a VGAE framework for *partition-conditioned link prediction*. Mask-GVAE [Li *et al.*, 2021a] further performs partition-based masking and reconstruction with an auxiliary spectral clustering objective to guarantee a stable graph partition. Structural clusters are also considered in *partition-based instance discrimination*: gCool [Li *et al.*, 2022a] enlarges the positive set by intra-community instances between two views, while CSGCL [Chen *et al.*, 2023a] uses the modularity-based community strength to weight node samples. Recently, StructComp [Zhang *et al.*, 2024] compresses features of nodes in the same community and performs community-wise contrast with compressed features.

**Discussion.** As the computational cost of some clustering algorithms becomes prohibitive for large networks, efficiency should be taken into account for cluster-based pretexts. Moreover, current cluster-based pretexts mainly rely on non-overlapping algorithms which assume that each node belongs to a single cluster. In real-world scenarios, however, a node often belongs to multiple clusters that necessitate overlapping clustering algorithms, which remains a challenge for existing SSL methods.

### 4.4 Global Structure

Compared to large-scale networks, it is generally easier to comprehend and utilize the global information of small graphs. However, the aggregation of global representations can also provide a holistic view of larger networks, which in turn facilitates the learning of microscopic instances.

**Global instance discrimination.** In analogy to node-level instance discrimination, global instance discrimination concentrates on the whole graph-level representation  $Z_G = \text{agg}(\mathbf{Z})$ , aggregated usually by a simple readout function. Representative methods include GraphCL [You *et al.*, 2020a], JOAO [You *et al.*, 2021], and SimGRACE [Xia *et al.*, 2022b], which apply the InfoNCE estimator to batches of small-scale graphs. In general, various augmentation schemes are adopted to construct positive views of a graph instance, while other graphs in the same batch are considered negative. Subsequent works include [Suresh *et al.*, 2021; Yang *et al.*, 2021; You *et al.*, 2022; Xu *et al.*, 2022b].

**Global-local discrimination.** This is a cross-scale approach applicable to both small and large graphs [Xu *et al.*, 2021a; Zhao *et al.*, 2023; Lin *et al.*, 2023]. The global representation  $Z_G$  acts like a “barycenter” that can match every

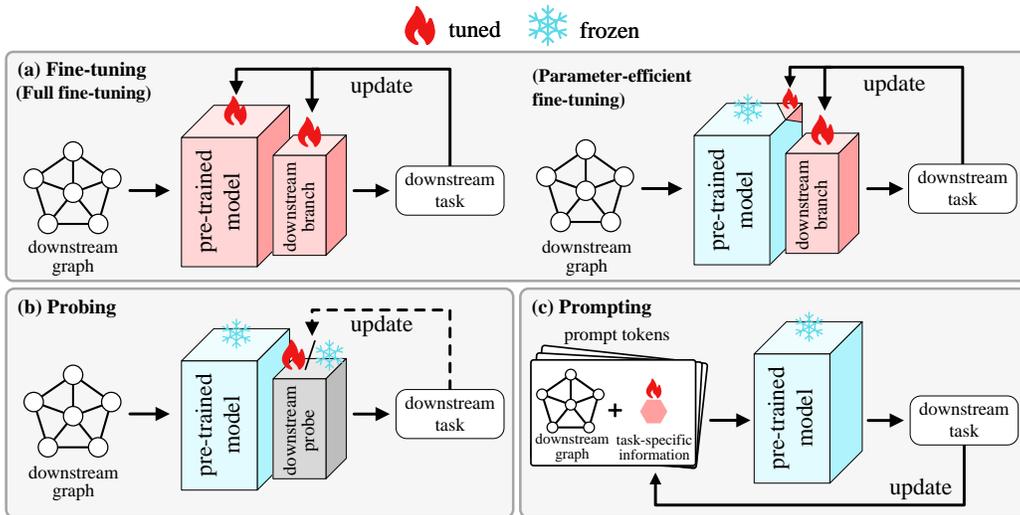


Figure 3: Task adaptation strategies: (a) fine-tuning, including full fine-tuning and parameter-efficient fine-tuning; (b) probing; (c) prompting.

single node in  $\mathcal{G}$  to form a positive pair. On the contrary, negative pairs are generated by one-sided perturbation. For example, DGI [Veličković *et al.*, 2019] and InfoGraph [Sun *et al.*, 2020a] optimizes a JS estimator between the “original node-original graph” pair  $(\mathbf{Z}_i, \mathbf{Z}_{\mathcal{G}})$  and the “perturbed node-original graph” pair  $(\tilde{\mathbf{Z}}_i, \mathbf{Z}_{\mathcal{G}})$ . MVGRL [Hassani and Khasahmadi, 2020] performs cross-view cross-scale contrast by further introducing the “original node-perturbed graph” pair  $(\mathbf{Z}_i, \mathbf{Z}_{\tilde{\mathcal{G}}})$ . This task is simplified by GGD [Zheng *et al.*, 2022] and D-SLA [Kim *et al.*, 2022] to *group discrimination* which performs binary classification of whether an instance belongs to the original or the perturbed view.

**Graph kernel prediction.** Graph kernels are bivariate functions quantifying the similarity between graphs, which have been widely used for graph classification. Beyond that, [Navarin *et al.*, 2018] predicts various graph kernels as a task-agnostic pre-training strategy, including the graphlet kernel, random walk kernel, propagation kernel, WL subtree kernel, etc. D-SLA [Kim *et al.*, 2022] generates a perturbed graph by adding and removing edges and uses the graph edit distance kernel (the number of edge modifying steps between original and perturbed graphs) to guide the learning of embedding distances.

**Half-graph matching.** PHD [Li *et al.*, 2021b] learns the small-scale global structure by dividing each graph into two halves and pairing them randomly. Then, each half-graph pair is concatenated by a virtual collection node. A GNN is trained to predict if the two halves are from the same original graph through a cross-entropy loss.

**Discussion.** Most graph pre-training approaches still resort to coarse-grained readout functions for a global view. On the other hand, perturbation-based pretexts like half-graph matching should be more cautious of the perturbation strength, especially when dealing with small graphs, as perturbations have a larger impact on their global semantics.

## 4.5 Manifolds

As some inherent topological properties cannot be effectively captured by GNNs parameterized in a Euclidean space, some special pretexts have explored the use of alternative Riemannian manifolds for improved latent space modeling. For example, the hyperbolic space is generally accepted as a better space for processing tree-like graphs.

**Cross-manifold discrimination.** Literally, this pretext creates contrastive views in different manifolds [Liu *et al.*, 2021]. DSGC [Yang *et al.*, 2022a] uses pairs of Euclidean and hyperbolic GNNs to encode views. SelfMGNN [Sun *et al.*, 2022a] builds a product space mixed by Euclidean, hyperbolic, and hypersphere spaces, to learn the manifold components of each graph by cross-view contrasting.

**Hyperbolic angle prediction.** This task is proposed by GraphJEPa [Skenderi *et al.*, 2023] for modeling the hierarchical structure of graphs. It aggregates high-dimensional representations and expresses them as angle vectors in a unit hyperbola, which are predicted by a smooth- $\ell_1$  loss.

**Discussion.** SSL on Riemannian manifolds remains to be explored. Since different data domains can reside in different manifolds, more flexible and efficient approaches beyond the product space [Sun *et al.*, 2022a] are expected.

## 5 Task Adaptation

This section provides an overview of *how to transfer graph knowledge from the pretexts to the downstream*, as formalized in eq (1). Downstream task adaptation methods are classified into three mainstreams: fine-tuning, probing, and prompting, illustrated in Figure 3.

**Fine-tuning.** Fine-tuning is one of the most prevalent task adaptation methods. It concatenates a pre-trained model with a simple downstream branch (e.g. a single-layer feed-forward network or GNN). Fine-tuning *jointly trains two modules with downstream supervision signals*. This strategy helps

bridge the gap between the pretext and the downstream, enabling effective adaptation of the pre-trained model. Previous GNN-based fine-tuning methods fully update all pre-trained parameters, referred to as *full fine-tuning*. Various fine-tuning techniques have been proposed to enhance knowledge transfer: AUX-TS [Han *et al.*, 2021] jointly uses the pretext and downstream objective for fine-tuning and adaptively weights them based on their task similarity. L2P-GNN [Lu *et al.*, 2021] adopts a meta-learning framework by dividing the pre-training data into support and query sets, simulating the adaptation process during pre-training. GTOT [Zhang *et al.*, 2022a] improves the distance between pre-trained and fine-tuned node embeddings to constrain the excessive change of model parameters during fine-tuning. However, full fine-tuning results in prohibitive computational cost when the pre-training large-scale models, and downstream tasks may introduce biases to the pre-trained parameters, causing a loss of generalizability. The latest work shifts towards *parameter-efficient fine-tuning* (PEFT) strategies [Houlsby *et al.*, 2019; Hu *et al.*, 2022] which, in general, only update part of the pre-trained parameters that has been embedded in the pre-training model beforehand. The embedded modules are typically small in size, e.g. feed-forward network layers, allowing for precise fine-tuning with minimal resource requirements. For instance, AdapterGNN [Li *et al.*, 2024] and G-Adapter [Gui *et al.*, 2023] design dedicated adapters for GNNs and graph Transformers, respectively.

**Probing.** Formerly known as “freezing” [Qiu *et al.*, 2020] or “feature extraction” [Hu *et al.*, 2019b], probing attaches a simple branch, i.e., “probe”, which can either be a network, a linear model, or even a non-parameterized mapping function, to the pre-trained model. *The pre-trained model remains frozen during the adaptation*, which can only provide deterministic representations for downstream branch training. Probing offers fairer evaluations of the pre-trained model since the performance directly reflects the generalizability of the learned representations from the unsupervised pretext. However, a simple probe is usually not able to support better downstream performance than that of fine-tuning.

**Prompting.** This is an emerging task adaptation strategy coupled with the rise of large language models. What we discuss here, however, are *graph prompts* adopted by graph models [Sun *et al.*, 2023b]. Its core idea is to explicitly bridge the task gap by jointly encoding downstream data and the corresponding task information as tokens called “prompts”. However, unlike prompts in natural language, graph prompts do not follow a deterministic form. They can take the form of a single node feature vector [Fang *et al.*, 2023], an aggregation of contextual features [Sun *et al.*, 2022b; Liu *et al.*, 2023b; Yu *et al.*, 2023a], or a specialized prompt graph [Sun *et al.*, 2023a; Huang *et al.*, 2023; Liu *et al.*, 2024]. These prompts usually contain encoded task-specific instructions to guide the model towards various downstream requirements. During the task adaptation phase, only the learnable part of the prompts is updated. Prompt tuning enables large pre-trained models to achieve high task generalizability without retraining the model parameters, so it has gained significant popularity in transfer learning and few-shot scenar-

ios [Zhu *et al.*, 2024; Zhao *et al.*, 2024a]. In spite of the promising advancements, graph prompting is still a developing area. For example, many graph prompts remain challenging for humans to comprehend.

## 6 Pre-training LLM-based Graph Models

In the last couple of years, there is a rapidly increasing interest in building large graph foundation models via graph textualization and large language models. The pre-training and adaptation strategies of LLMs differ significantly from existing graph models such as GNNs and graph Transformers, so it is compelling to effectively integrate them for diverse graph tasks. Given the uniqueness of language-based models, this section specifically focuses on the pre-training and task adaptation strategies for constructing large-scale graph models with LLMs.

**Pre-training of LLMs.** The most successful pretexts for LLMs to date include *autoregressive generation* (AG) and *mask language modeling* (MLM), which respectively lead to the GPT [Radford *et al.*, 2018] and BERT [Devlin *et al.*, 2019] family. AG inputs a sequence of tokens from a segment of some corpus and maximizes the log-likelihood of the next token. MLM randomly masks a set of tokens with “[MASK]” tokens or random tokens and learns to predict the original tokens. LLMs can handle many downstream tasks in the text domain solely based on AG and MLM due to the relatively homogeneous knowledge patterns of natural language.

**Pre-trained LLMs on graphs.** Despite the more complex knowledge patterns of graphs, the aforementioned pretexts are still used by most LLMs on graphs. Thus an issue arises: *how to adapt an LLM to graph-related downstream tasks?* (i) One may unify graph downstream tasks into a self-supervised language task for LLM fine-tuning, which may also align with the pretext. For example, WalkLM [Tan *et al.*, 2023b] fine-tunes a BERT-family model using MLM, where the textual sequences are generated through random walks on text-attributed graphs. (ii) Another cost-friendly approach is called “instruction tuning” or “in-context learning”, which formulates each downstream task as a textual instruction to the LLM to obtain a task-specific answer, sometimes with one or multiple examples for reference. Representative methods include NLGraph [Wang *et al.*, 2023a], InstructGLM [Ye *et al.*, 2023], and GraphGPT [Tang *et al.*, 2023]. (iii) An LLM can assist in training one or multiple additional GNNs as subordinates for graph-related tasks [Chen *et al.*, 2024; Huang *et al.*, 2024]. For example, GIANT [Chien *et al.*, 2022] utilizes BERT to directly generate numerical features for downstream graphs, while TAPE [He *et al.*, 2024] generates the node prediction results along with explanation paragraphs as the enhanced node textual attributes.

**Knowledge-based perspective.** Apart from the readily available textual attributes, several knowledge-aware models highlight the required graph knowledge during the adaptation of LLMs, as LLM-based graph models place more emphasis on downstream task adaptation. Microscopically, InstructGLM [Ye *et al.*, 2023] and GraphGPT [Tang *et al.*, 2023] integrate *context* information (Subsection 3.4) into the

LLM by conducting neighbor sampling on every node for instruction tuning (link prediction and graph-text matching, respectively). Macroscopically, LLM-GNN [Chen *et al.*, 2024] leverages the *cluster* pattern (Subsection 4.3), allowing the LLM to generate more reliable pseudo-labels for downstream training by selecting nodes closer to the cluster centers. Different from the methods above that directly inject graph knowledge into the LLM, Graph-ToolFormer [Zhang, 2024] fine-tunes an LLM by invoking different pre-trained graph models for the required knowledge, such as Graph-Bert [Zhang *et al.*, 2020] for node-level reasoning and K-means for community detection. However, the utilization of knowledge-specific prompts for LLM adaptation has yet to receive much attention.

## 7 Future Directions

**Deeper and wider graph knowledge discovery.** So far, SSL on graphs has resorted to methods from other domains, which often fall short of learning sufficient structural knowledge of graphs. Therefore, it is crucial to introduce graph-specific paradigms to uncover deeper generalizable knowledge hidden in the graph structure. In addition, there is a need for wider exploration of new knowledge types that are beneficial to generalizable graph representations.

**Theoretical guarantee.** The theoretical foundation of graph self-supervised pre-training is still lagging behind the empirical methods instead of guiding them. Theoretical principles are essential as they can serve as blueprints for pretext design. For instance, the graph information bottleneck [Wu *et al.*, 2020] has guided the design of numerous instance discrimination methods [Xu *et al.*, 2021a; Suresh *et al.*, 2021; Wei *et al.*, 2022; Wu *et al.*, 2023]. More self-supervised theoretical frameworks, such as multi-view frameworks [Tsai *et al.*, 2021] and energy-based frameworks [Kim and Ye, 2022], hold great potential for guiding more powerful pre-training methods for graph foundation models.

**Combining different pretexts effectively.** The success of different pretexts strongly depends on the downstream tasks [Jin *et al.*, 2022]. Therefore, to achieve a more generalizable and powerful graph learner, efforts should be made to effectively leverage these pretexts. Researchers are striving for developing different parameter search algorithms [Jin *et al.*, 2022; Ju *et al.*, 2023; Fan *et al.*, 2024] and knowledge distillation algorithms [Wu *et al.*, 2022] to address this issue. However, there is still a lack of research on effectively combining multiple graph pretexts.

**Bridging methodologies and applications.** It is often the case that pretexts and downstream adaptation strategies should cater to real-world factors, such as robustness, fairness, and explainability. There are some progresses such as attacks and defenses on graph contrastive models [Xu *et al.*, 2022a; Feng *et al.*, 2022; Zhang *et al.*, 2023], representation debiasing [Fan *et al.*, 2021; Wang *et al.*, 2022], and task-agnostic explanations [Xie *et al.*, 2022a; Wang *et al.*, 2023b], but they are limited to certain pretexts or frameworks. Future research should work towards designing pretexts that are generalizable to a wide range of real-world scenarios.

**Better graph foundation framework design.** Existing LLM-based graph models reveal a preference for textual attributes. Although some LLM-based research employs knowledge-aware prompting for compensation of structural knowledge, LLM-based graph models are not the optimal solution. The key direction and challenge for designing large graph foundation models lie in the native support for extracting and leveraging various types of graph knowledge, along with the corresponding large-scale self-supervised pretexts and adaptation strategies. Many microscopic tasks have been satisfactorily handled thanks to the rich text information of nodes. As the global structure provides a broader scope to discover the untapped potential of billion-scale parameters, we should more deeply investigate the *macroscopic knowledge* for constructing large graph models.

## References

- [Bardes *et al.*, 2022] Adrien Bardes, Jean Ponce, et al. VI-CReg: Variance-invariance-covariance regularization for self-supervised learning. In *ICLR*, 2022.
- [Bielak *et al.*, 2022] Piotr Bielak, Tomasz Kajdanowicz, et al. Graph Barlow Twins: A self-supervised representation learning framework for graphs. *KBS*, 2022.
- [Bo *et al.*, 2023] Deyu Bo, Yuan Fang, et al. Graph contrastive learning with stable and scalable spectral encoding. In *NeurIPS*, 2023.
- [Chen and Kou, 2023] Jialu Chen and Gang Kou. Attribute and structure preserving graph contrastive learning. In *AAAI*, 2023.
- [Chen *et al.*, 2020] Ting Chen, Simon Kornblith, et al. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.
- [Chen *et al.*, 2022] Jingfan Chen, Guanghui Zhu, et al. Towards self-supervised learning on graphs with heterophily. In *CIKM*, 2022.
- [Chen *et al.*, 2023a] Han Chen, Ziwen Zhao, et al. CSGCL: Community-strength-enhanced graph contrastive learning. In *IJCAI*, 2023.
- [Chen *et al.*, 2023b] Zhaoliang Chen, Zhihao Wu, et al. Dual low-rank graph autoencoder for semantic and topological networks. In *AAAI*, 2023.
- [Chen *et al.*, 2024] Zhikai Chen, Haitao Mao, et al. Label-free node classification on graphs with large language models (LLMs). In *ICLR*, 2024.
- [Cheng *et al.*, 2023] Jiashun Cheng, Man Li, et al. Wiener graph deconvolutional network improves graph self-supervised learning. In *AAAI*, 2023.
- [Chien *et al.*, 2022] Eli Chien, Wei-Cheng Chang, et al. Node feature extraction by self-supervised multi-scale neighborhood prediction. In *ICLR*, 2022.
- [Cui *et al.*, 2020] Ganqu Cui, Jie Zhou, et al. Adaptive graph encoder for attributed graph embedding. In *KDD*, 2020.
- [Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, et al. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.
- [Ding *et al.*, 2023] Kaize Ding, Yancheng Wang, et al. Eliciting structural and semantic global knowledge in unsupervised graph contrastive learning. In *AAAI*, 2023.
- [Dong *et al.*, 2022] Wei Dong, Junsheng Wu, et al. Node representation learning in graph via node-to-neighbourhood mutual information maximization. In *CVPR*, 2022.
- [Fan *et al.*, 2021] Wei Fan, Kunpeng Liu, et al. Fair graph auto-encoder for unbiased graph representations with wasserstein distance. In *ICDM*, 2021.
- [Fan *et al.*, 2023] Xiaolong Fan, Maoguo Gong, et al. Maximizing mutual information across feature and topology views for representing graphs. *TKDE*, 2023.
- [Fan *et al.*, 2024] Tianyu Fan, Lirong Wu, et al. Decoupling weighing and selecting for integrating multiple graph pre-training tasks. In *ICLR*, 2024.
- [Fang *et al.*, 2023] Taoran Fang, Yunchao Zhang, et al. Universal prompt tuning for graph neural networks. In *NeurIPS*, 2023.
- [Fatemi *et al.*, 2021] Bahare Fatemi, Layla El Asri, et al. SLAPS: Self-supervision improves structure learning for graph neural networks. In *NeurIPS*, 2021.
- [Feng *et al.*, 2022] Shengyu Feng, Baoyu Jing, et al. Adversarial graph contrastive learning with information regularization. In *WWW*, 2022.
- [Gong *et al.*, 2023] Xumeng Gong, Cheng Yang, et al. MAGCL: Model augmentation tricks for graph contrastive learning. In *AAAI*, 2023.
- [Grover *et al.*, 2019] Aditya Grover, Aaron Zweig, et al. Graphite: Iterative generative modeling of graphs. In *ICML*, 2019.
- [Gui *et al.*, 2023] Anchun Gui, Jinqiang Ye, et al. G-Adapter: Towards structure-aware parameter-efficient transfer learning for graph transformer networks. *arXiv:2305.10329*, 2023.
- [Hamilton *et al.*, 2017] Will Hamilton, Zhitao Ying, et al. Inductive representation learning on large graphs. In *NIPS*, 2017.
- [Han *et al.*, 2021] Xueting Han, Zhenhuan Huang, et al. Adaptive transfer learning on graph neural networks. In *KDD*, 2021.
- [Hasanzadeh *et al.*, 2019] Arman Hasanzadeh, Ehsan Hajiramezani, et al. Semi-implicit graph variational autoencoders. In *NeurIPS*, 2019.
- [Hassani and Khasahmadi, 2020] Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *ICML*, 2020.
- [He *et al.*, 2020] Kaiming He, Haoqi Fan, et al. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.
- [He *et al.*, 2022] Kaiming He, Xinlei Chen, et al. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.
- [He *et al.*, 2023] Dongxiao He, Jitao Zhao, et al. Contrastive learning meets homophily: two birds with one stone. In *ICML*, 2023.
- [He *et al.*, 2024] Xiaoxin He, Xavier Bresson, et al. Harnessing explanations: LLM-to-LM interpreter for enhanced text-attributed graph representation learning. In *ICLR*, 2024.
- [Ho *et al.*, 2020] Jonathan Ho, Ajay Jain, et al. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.
- [Hou *et al.*, 2022] Zhenyu Hou, Xiao Liu, et al. GraphMAE: Self-supervised masked graph autoencoders. In *KDD*, 2022.
- [Hou *et al.*, 2023] Zhenyu Hou, Yufei He, et al. GraphMAE2: A decoding-enhanced masked self-supervised graph learner. In *WWW*, 2023.

- [Houlsby *et al.*, 2019] Neil Houlsby, Andrei Giurgiu, et al. Parameter-efficient transfer learning for NLP. In *ICML*, 2019.
- [Hu *et al.*, 2019a] Weihua Hu, Bowen Liu, et al. Strategies for pre-training graph neural networks. *arXiv:1905.12265*, 2019.
- [Hu *et al.*, 2019b] Ziniu Hu, Changjun Fan, et al. Unsupervised pre-training of graph convolutional networks. In *ICLR Workshop (RLGM)*, 2019.
- [Hu *et al.*, 2020] Ziniu Hu, Yuxiao Dong, et al. GPT-GNN: Generative pre-training of graph neural networks. In *KDD*, 2020.
- [Hu *et al.*, 2021] Yang Hu, Haoxuan You, et al. Graph-MLP: Node classification without message passing in graph. *arXiv:2106.04051*, 2021.
- [Hu *et al.*, 2022] Edward Hu, Yelong Shen, et al. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022.
- [Huang *et al.*, 2023] Qian Huang, Hongyu Ren, et al. PRODIGY: Enabling in-context learning over graphs. In *NeurIPS*, 2023.
- [Huang *et al.*, 2024] Xuanwen Huang, Kaiqiao Han, et al. Can GNN be good adapter for LLMs? In *WWW*, 2024.
- [Hwang *et al.*, 2020] Dasol Hwang, Jinyoung Park, et al. Self-supervised auxiliary learning with meta-paths for heterogeneous graphs. In *NeurIPS*, 2020.
- [Inae *et al.*, 2023] Eric Inae, Gang Liu, et al. Motif-aware attribute masking for molecular graph pre-training. In *NIPS Workshop (GLFrontiers)*, 2023.
- [Jiang *et al.*, 2024] Xinke Jiang, Zidi Qin, et al. Incomplete graph learning via attribute-structure decoupled variational auto-encoder. In *WSDM*, 2024.
- [Jiao *et al.*, 2020] Yizhu Jiao, Yun Xiong, et al. Sub-graph contrast for scalable self-supervised graph representation learning. In *ICDM*, 2020.
- [Jin *et al.*, 2020] Wei Jin, Tyler Derr, et al. Self-supervised learning on graphs: Deep insights and new direction. *arXiv:2006.10141*, 2020.
- [Jin *et al.*, 2021] Ming Jin, Yizhen Zheng, et al. Multi-scale contrastive siamese networks for self-supervised graph representation learning. In *IJCAI*, 2021.
- [Jin *et al.*, 2022] Wei Jin, Xiaorui Liu, et al. Automated self-supervised learning for graphs. In *ICLR*, 2022.
- [Jin *et al.*, 2023] Bowen Jin, Gang Liu, et al. Large language models on graphs: A comprehensive survey. *arXiv:2312.02783*, 2023.
- [Ju *et al.*, 2023] Mingxuan Ju, Tong Zhao, et al. Multi-task self-supervised graph neural networks enable stronger task generalization. In *ICLR*, 2023.
- [Kim and Oh, 2021] Dongkwan Kim and Alice Oh. How to find your friendly neighborhood: Graph attention design with self-supervision. In *ICLR*, 2021.
- [Kim and Ye, 2022] Beomsu Kim and Jong Chul Ye. Energy-based contrastive learning of visual representations. In *NeurIPS*, 2022.
- [Kim *et al.*, 2022] Dongki Kim, Jinheon Baek, et al. Graph self-supervised learning with accurate discrepancy learning. In *NeurIPS*, 2022.
- [Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Variational graph auto-encoders. In *NIPS Workshop (BDL)*, 2016.
- [Lee *et al.*, 2022] Namkyeong Lee, Junseok Lee, et al. Augmentation-free self-supervised learning on graphs. In *AAAI*, 2022.
- [Li *et al.*, 2020] Jia Li, Jianwei Yu, et al. Dirichlet graph variational autoencoder. In *NeurIPS*, 2020.
- [Li *et al.*, 2021a] Jia Li, Mengzhou Liu, et al. Mask-GVAE: Blind denoising graphs via partition. In *WWW*, 2021.
- [Li *et al.*, 2021b] Pengyong Li, Jun Wang, et al. Pairwise half-graph discrimination: A simple graph-level self-supervised strategy for pre-training graph neural networks. In *IJCAI*, 2021.
- [Li *et al.*, 2022a] Bolian Li, Baoyu Jing, et al. Graph communal contrastive learning. In *WWW*, 2022.
- [Li *et al.*, 2022b] Jintang Li, Wangbin Sun, et al. Link prediction with contextualized self-supervision. *TKDE*, 2022.
- [Li *et al.*, 2023a] Jintang Li, Wangbin Sun, et al. Over-smoothing: A nightmare for graph contrastive learning? *arXiv:2306.02117*, 2023.
- [Li *et al.*, 2023b] Jintang Li, Ruofan Wu, et al. What’s behind the mask: Understanding masked graph modeling for graph autoencoders. In *KDD*, 2023.
- [Li *et al.*, 2023c] Wenzhi Li, Changdong Wang, et al. HomoGCL: Rethinking homophily in graph contrastive learning. In *KDD*, 2023.
- [Li *et al.*, 2023d] Wenzhi Li, Changdong Wang, et al. Towards effective and robust graph contrastive learning with graph autoencoding. *TKDE*, 2023.
- [Li *et al.*, 2023e] Xiang Li, Tiandi Ye, et al. SeeGera: Self-supervised semi-implicit graph variational auto-encoders with masking. In *WWW*, 2023.
- [Li *et al.*, 2024] Shengrui Li, Xueting Han, et al. AdapterGNN: Parameter-efficient fine-tuning improves generalization in gnn. In *AAAI*, 2024.
- [Lin *et al.*, 2023] Lu Lin, Jinghui Chen, et al. Spectral augmentation for self-supervised learning on graphs. In *ICLR*, 2023.
- [Liu *et al.*, 2021] Jiahong Liu, Menglin Yang, et al. Enhancing hyperbolic graph embeddings via contrastive learning. In *NIPS Workshop (SSL)*, 2021.
- [Liu *et al.*, 2022a] Nian Liu, Xiao Wang, et al. Revisiting graph contrastive learning from the perspective of graph spectrum. In *NeurIPS*, 2022.
- [Liu *et al.*, 2022b] Yixin Liu, Ming Jin, et al. Graph self-supervised learning: A survey. *TKDE*, 2022.

- [Liu *et al.*, 2023a] Jiawei Liu, Cheng Yang, et al. Towards graph foundation models: A survey and beyond. *arXiv:2310.11829*, 2023.
- [Liu *et al.*, 2023b] Zemin Liu, Xingtong Yu, et al. Graph-Prompt: Unifying pre-training and downstream tasks for graph neural networks. In *WWW*, 2023.
- [Liu *et al.*, 2024] Hao Liu, Jiarui Feng, et al. One for All: Towards training one graph model for all classification tasks. In *ICLR*, 2024.
- [Lu *et al.*, 2021] Yuanfu Lu, Xunqiang Jiang, et al. Learning to pre-train graph neural networks. In *AAAI*, 2021.
- [Luong and Singh, 2023] Kha-Dinh Luong and Ambuj Singh. Fragment-based pretraining and finetuning on molecular graphs. In *NeurIPS*, 2023.
- [Mo *et al.*, 2022] Yujie Mo, Liang Peng, et al. Simple unsupervised graph representation learning. In *AAAI*, 2022.
- [Navarin *et al.*, 2018] Nicolò Navarin, Dinh V Tran, et al. Pre-training graph neural networks with kernels. *arXiv:1811.06930*, 2018.
- [Oord *et al.*, 2018] Aaron van den Oord, Yazhe Li, et al. Representation learning with contrastive predictive coding. *arXiv:1807.03748*, 2018.
- [Pan *et al.*, 2018] Shirui Pan, Ruiqi Hu, et al. Adversarially regularized graph autoencoder for graph embedding. In *IJCAI*, 2018.
- [Park *et al.*, 2019] Jiwoong Park, Minsik Lee, et al. Symmetric graph convolutional autoencoder for unsupervised graph representation learning. In *ICCV*, 2019.
- [Peng *et al.*, 2020] Zhen Peng, Wenbing Huang, et al. Graph representation learning via graphical mutual information maximization. In *WWW*, 2020.
- [Peng *et al.*, 2022] Zhen Peng, Yixiang Dong, et al. A new self-supervised task on graphs: Geodesic distance prediction. *Information Sciences*, 2022.
- [Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, et al. DeepWalk: Online learning of social representations. In *KDD*, 2014.
- [Qiu *et al.*, 2020] Jiezhong Qiu, Qibin Chen, et al. GCC: Graph contrastive coding for graph neural network pre-training. In *KDD*, 2020.
- [Radford *et al.*, 2018] Alec Radford, Karthik Narasimhan, et al. Improving language understanding by generative pre-training. *OpenAI*, 2018.
- [Rong *et al.*, 2020] Yu Rong, Yatao Bian, et al. Self-supervised graph transformer on large-scale molecular data. In *NeurIPS*, 2020.
- [Shiao *et al.*, 2023a] William Shiao, Zhichun Guo, et al. Link prediction with non-contrastive learning. In *ICLR*, 2023.
- [Shiao *et al.*, 2023b] William Shiao, Uday Singh Saini, et al. CARL-G: Clustering-accelerated representation learning on graphs. In *KDD*, 2023.
- [Skenderi *et al.*, 2023] Geri Skenderi, Hang Li, et al. Graph-level representation learning with joint-embedding predictive architectures. *arXiv:2309.16014*, 2023.
- [Sun *et al.*, 2020a] Fanyun Sun, Jordan Hoffmann, et al. InfoGraph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In *ICLR*, 2020.
- [Sun *et al.*, 2020b] Ke Sun, Zhouchen Lin, et al. Multi-stage self-supervised learning for graph convolutional networks on graphs with few labeled nodes. In *AAAI*, 2020.
- [Sun *et al.*, 2022a] Li Sun, Zhongbao Zhang, et al. A self-supervised mixed-curvature graph neural network. In *AAAI*, 2022.
- [Sun *et al.*, 2022b] Mingchen Sun, Kaixiong Zhou, et al. GPPT: Graph pre-training and prompt tuning to generalize graph neural networks. In *KDD*, 2022.
- [Sun *et al.*, 2023a] Xiangguo Sun, Hong Cheng, et al. All in One: Multi-task prompting for graph neural networks. In *KDD*, 2023.
- [Sun *et al.*, 2023b] Xiangguo Sun, Jiawen Zhang, et al. Graph prompt learning: A comprehensive survey and beyond. *arXiv:2311.16534*, 2023.
- [Sun *et al.*, 2024] Wangbin Sun, Jintang Li, et al. Rethinking and simplifying bootstrapped graph latents. In *WSDM*, 2024.
- [Suresh *et al.*, 2021] Susheel Suresh, Pan Li, et al. Adversarial graph augmentation to improve graph contrastive learning. In *NeurIPS*, 2021.
- [Tan *et al.*, 2023a] Qiaoyu Tan, Ninghao Liu, et al. S2GAE: Self-supervised graph autoencoders are generalizable learners with graph masking. In *WSDM*, 2023.
- [Tan *et al.*, 2023b] Yanchao Tan, Zihao Zhou, et al. WalkLM: A uniform language model fine-tuning framework for attributed graph embedding. In *NeurIPS*, 2023.
- [Tang *et al.*, 2022] Mingyue Tang, Carl Yang, et al. Graph auto-encoder via neighborhood Wasserstein reconstruction. In *ICLR*, 2022.
- [Tang *et al.*, 2023] Jiabin Tang, Yuhao Yang, et al. GraphGPT: Graph instruction tuning for large language models. *arXiv:2310.13023*, 2023.
- [Thakoor *et al.*, 2022] Shantanu Thakoor, Corentin Tallec, et al. Large-scale representation learning on graphs via bootstrapping. In *ICLR*, 2022.
- [Tian *et al.*, 2023] Yijun Tian, Kaiwen Dong, et al. Heterogeneous graph masked autoencoders. In *AAAI*, 2023.
- [Tsai *et al.*, 2021] Yao-Hung Hubert Tsai, Yue Wu, et al. Self-supervised learning from a multi-view perspective. In *ICLR*, 2021.
- [Veličković *et al.*, 2019] Petar Veličković, William Fedus, et al. Deep graph infomax. In *ICLR*, 2019.
- [Vincent *et al.*, 2008] Pascal Vincent, Hugo Larochelle, et al. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008.

- [Wan *et al.*, 2021] Sheng Wan, Shirui Pan, et al. Contrastive and generative graph convolutional networks for graph-based semi-supervised learning. In *AAAI*, 2021.
- [Wang *et al.*, 2017] Chun Wang, Shirui Pan, et al. MGAE: Marginalized graph autoencoder for graph clustering. In *CIKM*, 2017.
- [Wang *et al.*, 2020] Xiao Wang, Meiqi Zhu, et al. AM-GCN: Adaptive multi-channel graph convolutional networks. In *KDD*, 2020.
- [Wang *et al.*, 2022] Ruijia Wang, Xiao Wang, et al. Uncovering the structural fairness in graph contrastive learning. In *NeurIPS*, 2022.
- [Wang *et al.*, 2023a] Heng Wang, Shangbin Feng, et al. Can language models solve graph problems in natural language? *arXiv:2305.10037*, 2023.
- [Wang *et al.*, 2023b] Jihong Wang, Minnan Luo, et al. Empower post-hoc graph explanations with information bottleneck: A pre-training and fine-tuning perspective. In *KDD*, 2023.
- [Wang *et al.*, 2023c] Jinpeng Wang, Ziyun Zeng, et al. MIS-Rec: Pre-training and transferring multi-modal interest-aware sequence representation for recommendation. In *MM*, 2023.
- [Wang *et al.*, 2024] Liang Wang, Xiang Tao, et al. Rethinking graph masked autoencoders through alignment and uniformity. In *AAAI*, 2024.
- [Wei *et al.*, 2022] Chunyu Wei, Jian Liang, et al. Contrastive graph structure learning via information bottleneck for recommendation. In *NeurIPS*, 2022.
- [Winter *et al.*, 2021] Robin Winter, Frank Noé, et al. Permutation-invariant variational autoencoder for graph-level representation learning. In *NeurIPS*, 2021.
- [Wu *et al.*, 2020] Tailin Wu, Hongyu Ren, et al. Graph information bottleneck. In *NeurIPS*, 2020.
- [Wu *et al.*, 2022] Lirong Wu, Yufei Huang, et al. Automated graph self-supervised learning via multi-teacher knowledge distillation. *arXiv:2210.02099*, 2022.
- [Wu *et al.*, 2023] Junran Wu, Xueyuan Chen, et al. SEGA: Structural entropy guided anchor view for graph contrastive learning. In *ICML*, 2023.
- [Xia *et al.*, 2022a] Jun Xia, Lirong Wu, et al. ProGCL: Rethinking hard negative mining in graph contrastive learning. In *ICML*, 2022.
- [Xia *et al.*, 2022b] Jun Xia, Lirong Wu, et al. SimGRACE: A simple framework for graph contrastive learning without data augmentation. In *WWW*, 2022.
- [Xia *et al.*, 2022c] Jun Xia, Yanqiao Zhu, et al. A survey of pretraining on graphs: Taxonomy, methods, and applications. *arXiv:2202.07893*, 2022.
- [Xiang *et al.*, 2023] Weilai Xiang, Hongyu Yang, et al. Denoising diffusion autoencoders are unified self-supervised learners. In *ICCV*, 2023.
- [Xie *et al.*, 2022a] Yaochen Xie, Sumeet Katariya, et al. Task-agnostic graph explanations. In *NeurIPS*, 2022.
- [Xie *et al.*, 2022b] Yaochen Xie, Zhao Xu, et al. Self-supervised learning of graph neural networks: A unified review. *TPAMI*, 2022.
- [Xie *et al.*, 2022c] Yaochen Xie, Zhao Xu, et al. Self-supervised representation learning via latent graph prediction. In *ICML*, 2022.
- [Xu *et al.*, 2021a] Dongkuan Xu, Wei Cheng, et al. InfoGCL: Information-aware graph contrastive learning. In *NeurIPS*, 2021.
- [Xu *et al.*, 2021b] Minghao Xu, Hang Wang, et al. Self-supervised graph-level representation learning with local and global structure. In *ICML*, 2021.
- [Xu *et al.*, 2022a] Jiarong Xu, Yang Yang, et al. Unsupervised adversarially robust representation learning on graphs. In *AAAI*, 2022.
- [Xu *et al.*, 2022b] Xinyi Xu, Cheng Deng, et al. Group contrastive self-supervised learning on graphs. *TPAMI*, 2022.
- [Yan *et al.*, 2024] Pengwei Yan, Kaisong Song, et al. Empowering dual-level graph self-supervised pretraining with motif discovery. In *AAAI*, 2024.
- [Yang *et al.*, 2021] Longqi Yang, Liangliang Zhang, et al. Graph adversarial self-supervised learning. In *NeurIPS*, 2021.
- [Yang *et al.*, 2022a] Haoran Yang, Hongxu Chen, et al. Dual space graph contrastive learning. In *WWW*, 2022.
- [Yang *et al.*, 2022b] Yaming Yang, Ziyu Guan, et al. Self-supervised heterogeneous graph pre-training based on structural clustering. In *NeurIPS*, 2022.
- [Yang *et al.*, 2023] Run Yang, Yuling Yang, et al. Directional diffusion models for graph representation learning. In *NeurIPS*, 2023.
- [Ye *et al.*, 2023] Ruosong Ye, Caiqi Zhang, et al. Natural language is all a graph needs. *arXiv:2308.07134*, 2023.
- [You *et al.*, 2020a] Yuning You, Tianlong Chen, et al. Graph contrastive learning with augmentations. In *NeurIPS*, 2020.
- [You *et al.*, 2020b] Yuning You, Tianlong Chen, et al. When does self-supervision help graph convolutional networks? In *ICML*, 2020.
- [You *et al.*, 2021] Yuning You, Tianlong Chen, et al. Graph contrastive learning automated. In *ICML*, 2021.
- [You *et al.*, 2022] Yuning You, Tianlong Chen, et al. Bringing your own view: Graph contrastive learning without prefabricated data augmentations. In *WSDM*, 2022.
- [Yu *et al.*, 2023a] Xingtong Yu, Zhenghao Liu, et al. Generalized graph prompt: Toward a unification of pre-training and downstream tasks on graphs. *arXiv:2311.15317*, 2023.
- [Yu *et al.*, 2023b] Yue Yu, Xiao Wang, et al. Provable training for graph contrastive learning. In *NeurIPS*, 2023.

- [Zbontar *et al.*, 2021] Jure Zbontar, Li Jing, et al. Barlow Twins: Self-supervised learning via redundancy reduction. In *ICML*, 2021.
- [Zhang *et al.*, 2020a] Jiawei Zhang, Haopeng Zhang, et al. Graph-Bert: Only attention is needed for learning graph representations. *arXiv:2001.05140*, 2020.
- [Zhang *et al.*, 2020b] Tianqi Zhang, Yun Xiong, et al. Comdgi: Community detection oriented deep graph infomax. In *CIKM*, 2020.
- [Zhang *et al.*, 2021a] Hengrui Zhang, Qitian Wu, et al. From canonical correlation analysis to self-supervised graph neural networks. In *NeurIPS*, 2021.
- [Zhang *et al.*, 2021b] Shichang Zhang, Ziniu Hu, et al. Motif-driven contrastive learning of graph representations. In *WWW Workshop (SSL)*, 2021.
- [Zhang *et al.*, 2021c] Zaixi Zhang, Qi Liu, et al. Motif-based graph self-supervised learning for molecular property prediction. In *NeurIPS*, 2021.
- [Zhang *et al.*, 2022a] Jiying Zhang, Xi Xiao, et al. Fine-tuning graph neural networks via graph topology induced optimal transport. In *IJCAI*, 2022.
- [Zhang *et al.*, 2022b] Rui Zhang, Li Miao, et al. A benchmark and comprehensive survey on knowledge graph entity alignment via representation learning. *VLDB J*, 2022.
- [Zhang *et al.*, 2022c] Yifei Zhang, Hao Zhu, et al. COSTA: Covariance-preserving feature augmentation for graph contrastive learning. In *KDD*, 2022.
- [Zhang *et al.*, 2023] Hangfan Zhang, Jinghui Chen, et al. Graph contrastive backdoor attacks. In *ICML*, 2023.
- [Zhang *et al.*, 2024] Shengzhong Zhang, Wenjie Yang, et al. StructComp: Substituting propagation with structural compression in training graph contrastive learning. In *ICLR*, 2024.
- [Zhang, 2024] Jiawei Zhang. Graph-ToolFormer: To empower LLMs with graph reasoning ability via prompt augmented by chatgpt. *arXiv:2304.11116*, 2024.
- [Zhao *et al.*, 2023] Wenting Zhao, Gongping Xu, et al. Deep graph structural infomax. In *AAAI*, 2023.
- [Zhao *et al.*, 2024a] Haihong Zhao, Aochuan Chen, et al. All in one and one for all: A simple yet effective method towards cross-domain graph pretraining. *arXiv:2402.09834*, 2024.
- [Zhao *et al.*, 2024b] Ziwen Zhao, Yuhua Li, et al. Masked graph autoencoder with non-discrete bandwidths. In *WWW*, 2024.
- [Zheng *et al.*, 2022] Yizhen Zheng, Shirui Pan, et al. Rethinking and scaling up graph contrastive learning: An extremely efficient approach with group discrimination. In *NeurIPS*, 2022.
- [Zhu and Koniusz, 2022] Hao Zhu and Piotr Koniusz. Generalized Laplacian Eigenmaps. In *NeurIPS*, 2022.
- [Zhu *et al.*, 2020] Yanqiao Zhu, Yichen Xu, et al. Deep graph contrastive representation learning. In *ICML Workshop (GRL+)*, 2020.
- [Zhu *et al.*, 2021a] Hao Zhu, Ke Sun, et al. Contrastive Laplacian Eigenmaps. In *NeurIPS*, 2021.
- [Zhu *et al.*, 2021b] Qi Zhu, Carl Yang, et al. Transfer learning of graph neural networks with ego-graph information maximization. In *NeurIPS*, 2021.
- [Zhu *et al.*, 2021c] Yanqiao Zhu, Yichen Xu, et al. Graph contrastive learning with adaptive augmentation. In *WWW*, 2021.
- [Zhu *et al.*, 2024] Yun Zhu, Yaoke Wang, et al. GraphControl: Adding conditional control to universal graph pre-trained models for graph domain transfer learning. In *WWW*, 2024.