Unlocking the Potential of Black-box Pre-trained GNNs for Graph Few-shot Learning

Qiannan Zhang[1], Shichao Pei[2], Yuan Fang[3*], Xiangliang Zhang[4*]
[1]Cornell University, USA
[2]University of Massachusetts Boston, USA
[3]Singapore Management University, Singapore
[4]University of Notre Dame, USA
qiz4005@med.cornell.edu, shichao.pei@umb.edu, yfang@smu.edu.sg, xzhang33@nd.edu

Abstract

Few-shot learning has emerged as an important problem on graphs to combat label scarcity, which can be approached by current trends in pre-trained graph neural networks (GNNs) and meta-learning. Recent efforts integrate both paradigms in a white-box setting, leaving the more realistic black-box setting largely underexplored, where the parameters and gradients in the pre-trained GNNs are inaccessible. In this paper, we study the critical problem: Leveraging black-box pre-trained GNNs for graph few-shot learning. Despite its appeal, two key issues hinder the unlocking of its potential: the inherent task gap between pre-training and downstream stages, which can introduce irrelevant knowledge and undermine the generalizability of a pre-trained black-box GNN on downstream tasks; and the inaccessibility of parameters and gradients, which limits the model's adaptation to novel tasks. To effectively leverage the black-box pre-trained GNNs and improve generalization, we propose a lightweight graph meta-learner to extract relevant knowledge from a black-box pre-trained GNN, meanwhile harnessing knowledge from related tasks for rapid adaptation on novel tasks. Furthermore, we prune the graph meta-learner to enhance its generalization on novel tasks. Extensive experiments on real-world datasets for few-shot node classification validate the effectiveness of our proposed method in the black-box setting.

Introduction

Graph learning models (Chen et al. 2018; Kipf and Welling 2017; Xu et al. 2019; Wu et al. 2020) have become pervasive in various real-world applications, ranging from social network analysis (Tang and Liu 2010) to drug discovery (Gaudelet et al. 2021). Despite its popularity, conventional supervised training on graphs usually necessitates a large volume of annotated data to improve model performance on a specific task. Thus, when a task has limited labels due to high annotation expense, model performance is often degraded. To tackle label scarcity, two research trends for few-shot learning on graphs have emerged and gained significant traction (Wang et al. 2020; Bose et al. 2019; Huang and Zitnik 2020; Ding et al. 2021). *First*, a growing trend on pre-trained GNNs (Hu et al. 2020b; Jiang et al. 2021; Liu et al. 2023a) aims to harness the knowledge acquired with various self-supervised pre-training strategies on large-scale label-free graphs, thus reducing the dependence on labeling for the downstream tasks. *Second*, meta-learning emerges as a promising paradigm for achieving rapid adaptation to novel tasks with few-shot annotations, provided that the model has been "meta-trained" on a series of related tasks (Hospedales, Antoniou, and Micaelli 2020).

However, each paradigm has its drawbacks thus hindering their effective application in real-world scenarios. On the one hand, pre-trained GNNs mostly rely on intrinsic information from graph topology and features to capture the general knowledge within graphs. Yet the lack of task-specific knowledge at pre-training may inhibit the pre-trained GNNs from generalizing well to downstream few-shot tasks. On the other hand, meta-learning aims to acquire meta-knowledge from related meta-training tasks, assuming these tasks are independent and identically distributed (i.i.d.), drawn from the same task distribution as the meta-testing tasks. However, the i.i.d. assumption may not hold for graph data where nodes are interconnected, such that the information gained from meta-training tasks may not perform effectively on meta-testing tasks. Here, general graph knowledge from pre-trained GNNs might serve as a useful complement for task adaptation in meta-learning. Given their respective drawbacks, the lack of integration between the two paradigms can limit the ability to tackle data scarcity.

To fully capitalize on the advantages offered by both trends, some endeavors are conducted to combine pre-trained models and meta-learning in a *white-box setting* where the parameters and gradients in pre-trained models are *accessible* (Sablayrolles et al. 2019). One straightforward approach applied on molecular graphs meta-trains GNNs initialized with the pre-trained parameters (Guo et al. 2021; Wang et al. 2021). Recent work (Tan et al. 2023) also gains insights from the success of prompt techniques in language models (Liu et al. 2023b) and meta-learns continuous prompts in the embedding space based on pre-trained GNNs. However, the white-box assumption may not be practical in some real-world scenarios. The prevalence of pre-trained models inspires the Model-as-a-Service (MaaS) (Brown et al. 2020; Sun et al. 2021), which has been the new norm of the interplay between the cloud infrastructure and edge devices. In this case, the pre-trained model is hosted on a remote server or accessed via cloud services.

*Corresponding Authors.

And service providers usually encapsulate the model parameters and expose APIs to avoid model reverse-engineering and protect proprietary. The *black-box setting* disables back-propagation due to unavailable parameters and gradients, offering a promising mechanism to leverage pre-trained models and secure service infrastructure. Moreover, operating in a black-box setting with no need for complex parameter sharing or model fine-tuning allows for more efficient and scalable deployment of pre-trained models in various existing workflows and applications. Despite its appeal, the utilization of black-box pre-trained GNNs for graph few-shot learning is largely underexplored. Existing studies require fine-tuning the parameters of pre-trained GNNs or optimizing continuous prompts by back-propagation in pre-trained models, both necessitating access to parameters and gradients and violating the black-box setting.

The research gap motivates us to investigate the novel setting, where the black-box pre-trained GNNs take a graph as input and only output the representation of nodes. In this work, we aim to design an effective strategy to employ the black-box pre-trained GNNs for graph few-shot learning, yet inevitably face two challenges, of which the first is **(C1) how to effectively utilize the black-box pre-trained GNNs.** A naive way (Tan et al. 2022b) is fine-tuning a classifier using the output node representations for downstream tasks, yet the lack of task-specific knowledge prevents the generalization on downstream few-shot tasks. Also, due to the objective gap between pre-training and downstream tasks, task-irrelevant knowledge in pre-trained GNNs tends to mislead the prediction and harm the effectiveness of few-shot learning. Therefore, devising a learnable module for bridging the black-box pre-trained GNNs and downstream tasks to leverage downstream task-specific knowledge is a primary and vital aim. Meanwhile, the module is expected to be equipped with the ability to extract task-relevant knowledge from the black-box pre-trained GNNs to further alleviate the impact of task-irrelevant knowledge. If possible to design such a module, the module can be meta-trained on a series of related tasks and fast adapted to novel tasks. Thus, the second challenge is **(C2) how to design a learnable module to improve the generalization on novel tasks.** Meta-learning assumes there is a large number of diverse meta-training tasks. Nevertheless, a sufficient variety of meta-training tasks can be absent in the real-world graph data, leading the model to memorize meta-training tasks and limiting its generalization ability on the meta-testing tasks (Yao et al. 2021). Provided with insufficient meta-training tasks, determining the optimal size of the learnable module poses a complex dilemma. A compact module might prove inadequate for acquiring knowledge about task adaptation, while a larger one would lead to memorization and overfitting. Thus, seeking a solution to improve generalization in this scenario deserves more thorough exploration.

In light of this, we propose a novel framework called Graph **Meta**-learning with **B**lack-box **P**re-trained GNNs (**Meta-BP**) for few-shot learning on graphs. It aims to unify the two paradigms of pre-training and meta-learning in a *black-box setting*. To address the challenge **(C1)**, we introduce a lightweight learnable module called graph meta-learner to extract minimal sufficient information from the black-box pre-trained GNN to exclude task-irrelevant knowledge, and further adapt the knowledge to suit each individual task. Thus, the graph meta-learner not only capitalizes on task-relevant knowledge from the black-box pre-trained GNN but also the meta-knowledge from related meta-training tasks. Note that the reason for using a lightweight graph meta-learner is to reduce the dependency on computational resources in the real-world scenario, where users of black-box models may not have enough resources to perform large-scale model training. Then, to improve the generalization and tackle the challenge **(C2)**, inspired by the lottery ticket hypothesis (Frankle and Carbin 2019), we propose to optimize the size of the graph meta-learner, and extract a subnetwork from the meta-learner for fast adaptation on the novel tasks in meta-testing. To conform to the black-box setting, only the lightweight graph meta-learner is updated, while the pre-trained GNN is kept frozen and disables the back-propagation in both meta-training and -testing.

Our contributions are summarized as follows:

(1) To our best knowledge, **Meta-BP** is the first work to address few-shot learning on graphs by integrating meta-learning with pre-trained GNNs in the black-box setting.

(2) We design a graph meta-learner to extract the minimal sufficient information that excludes task-irrelevant knowledge to achieve effective utilization of black-box pre-trained GNNs. Then, we propose to extract a subnetwork from the graph meta-learner to improve the generalization on novel tasks.

(3) We conduct extensive experiments on real-world graphs and show the effectiveness of Meta-BP on few-shot node classification with black-box pre-trained GNNs.

Related Work

Graph Pre-training

Graph pre-training aims to obtain graph representations by employing carefully crafted pre-training tasks, thereby mitigating the expense of annotating data for downstream tasks. A line of research on graph pre-training involves utilizing self-generated targets derived from both graph topology and features to facilitate the training of GNNs (Hu et al. 2020b; Jin et al. 2020). In addition, contrastive techniques come into play by enhancing the alignment among analogous graph instances while distancing the dissimilar ones (Sun et al. 2019; You et al. 2020; Qiu et al. 2020; Liu et al. 2024). For example, DGI (Velickovic et al. 2019) focuses on maximizing mutual information between local and global graph representations. GMI (Peng et al. 2020) introduces graphical mutual information as a means to incorporate topological insights. Despite the rich information it may preserve, the application of pre-trained GNNs to downstream tasks usually entails fine-tuning the whole parameter set (Guo et al. 2021; Wang et al. 2021) or prompt-tuning a set of prompts (Tan et al. 2023; Sun et al. 2022, 2023), leading to a violation of the black-box setting. A recent work (Tan et al. 2022b) leverages node representation from pre-trained GNNs directly with a classifier, yet it fails to handle the irrelevant

knowledge within pre-trained GNNs and is unable to effectively utilize the pre-trained models.

Graph Meta-learning

As a prevalent paradigm for few-shot learning, meta-learning has diverged into optimization-based approaches (Finn, Abbeel, and Levine 2017; Li et al. 2017), and metric-based methods (Li et al. 2019; Yoon, Seo, and Moon 2019). To counter label scarcity in graph learning, recent studies explore varied meta-knowledge to transfer, including initializing graph representations (Kipf and Welling 2017; Wang et al. 2020; Bose et al. 2019; Huang and Zitnik 2020; Zhang et al. 2022a; Wang et al. 2023; Liu et al. 2022; Pei et al. 2023; Li et al. 2024) and understanding metric space properties (Ding et al. 2020; Lu et al. 2022; Qu et al. 2020; Yao et al. 2020; Tan et al. 2022a). Existing methods mostly employ various GNNs as the base learner. Nevertheless, training these GNNs from scratch entails high computational resources and often leads to overfitting (Chen et al. 2019). The generalization of these models is also hampered by the lack of general knowledge about the graph. A few works (Guo et al. 2021; Wang et al. 2021; Zhang et al. 2023; Tan et al. 2023) leverage pre-trained models to complement meta-learning, yet necessitate access to gradients in pre-trained models and are not suited to the black-box setting.

Preliminary

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ denote a graph where $\mathcal{V}$ represents a set of nodes and $\mathcal{E}$ denotes a set of edges. Nodes possess features $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times d}$ with a dimension of d and share the labels $\mathcal{Y} = \{y_1, y_2, \ldots, y_{|\mathcal{Y}|}\}$. In this paper, we center on node classification, assuming there are disjoint node labels $\mathcal{Y} = \{\mathcal{Y}_{tr}, \mathcal{Y}_{ts}\}$, where $\mathcal{Y}_{tr}$ and $\mathcal{Y}_{ts}$ denote the training and testing classes respectively and $\mathcal{Y}_{tr} \cap \mathcal{Y}_{ts} = \emptyset$. We follow the conventions of few-shot learning, considering $\mathcal{Y}_{ts}$ as novel classes with limited labeled nodes and $\mathcal{Y}_{tr}$ as base classes with sufficient labels. Hence, the objective is to develop a graph learning model $\mathcal{M}$ such that it can accurately predict node labels in $\mathcal{Y}_{ts}$ given few-shot annotations during meta-testing, after sufficient meta-training iterations with $\mathcal{Y}_{tr}$. We seek to take advantage of a black-box pre-trained GNN $\mathbf{f}_{pre}$ that provides general knowledge about the graph to complement meta-learning. And $\mathbf{f}_{pre}$ can be any GNN endowed with diverse pre-training strategies (Jin et al. 2020). Note that our focus is to leverage the existing pre-trained models, rather than crafting novel pre-training strategies.

Episodic Training More specifically, we adopt an episodic training paradigm (Finn, Abbeel, and Levine 2017) where training and testing data are framed as a series of related N-way K-shot meta-tasks denoted as $\mathcal{T}_{tr}$ and $\mathcal{T}_{ts}$. Each meta-task τ is generated by randomly sampling N different classes $\mathcal{Y}_\tau$, where $\mathcal{Y}_\tau \subset \mathcal{Y}$ and $|\mathcal{Y}_\tau| = N$. Upon that, K and J labeled nodes are sampled to form the support set Ω^s and the query set Ω^q of each meta-task. The optimization-based meta-learning adopted in this work involves minimizing the loss on the query set Ω^q (meta-update in the outer loop), using the parameters obtained by minimizing the loss on the support set Ω^s (inner-update in the inner loop). The learned model then facilitates fast adaptation to novel classes.

Proposed Model

To reap the advantages of both meta-learning and pre-trained GNNs in a black-box setting, we propose a lightweight graph meta-learner to extract the relevant knowledge from the black-box pre-trained GNN for downstream meta-tasks without accessing its parameters and gradients, and meanwhile, obtain the knowledge of how to adapt to novel meta-tasks via meta-training with related meta-tasks. However, extracting relevant knowledge from pre-trained GNNs for downstream meta-tasks is non-trivial. The knowledge learned from pre-trained models often contains task-irrelevant details that are not directly applicable to downstream meta-tasks or even detrimental to the inference on these tasks. To mitigate the irrelevant interference originating from the pre-trained model, the graph meta-learner aims to extract approximately *minimal sufficient information* specifically tailored to the meta-tasks. Therefore, we meta-optimize the graph meta-learner using information bottleneck (Tishby, Pereira, and Bialek 2000) to obtain the node representation with only relevant knowledge from pre-trained GNN yet sufficient information for label prediction. To further enhance generalization on novel meta-tasks, we prune the graph meta-learner and extract a subnetwork from the learned graph meta-learner before transferring it to the meta-testing tasks. The subnetwork enables more effective fast adaptation on novel tasks. The minimum sufficient information extraction and meta-learner pruning are integrated along with the meta-optimization of graph meta-learner in an end-to-end framework. Next, we elaborate on the design of Meta-BP, with the overall framework shown in Figure 1.

Designing Graph Meta-learner

To bridge the black-box pre-trained GNN $\mathbf{f}_{pre}$ and downstream meta-tasks, we design a simple yet effective graph meta-learner to capture the general knowledge from $\mathbf{f}_{pre}$ to benefit the meta-tasks. Formally, we denote the graph meta-learner as $\mathrm{GML}(\cdot)$, which works by:

$$\mathbf{Z} = \mathrm{GML}(\mathbf{O}, \mathcal{G}; \phi) \tag{1}$$

where $\mathcal{G}$ denotes the given graph and $\mathrm{GML}(\cdot)$, parameterized by ϕ, obtains the representations $\mathbf{Z}$ for nodes in $\mathcal{G}$. In this work, without contradicting the black-box setting, we assume that the black-box $\mathbf{f}_{pre}$ can output last-layer node representations denoted as $\mathbf{O}$, following $\mathbf{O} = \mathbf{f}_{pre}(\mathcal{G})$.

To account for both graph topology and features, we summarize the neighborhood information by neighbor abstraction. Specifically, taking the initial feature $\mathbf{x}_v$ of node v as input, the last-layer node representations generated by $\mathbf{f}_{pre}$ can be obtained, denoted as $\mathbf{h}_v$. Then the neighbor abstraction for node v considering graph adjacency can be simply computed as $\mathbf{h}_{\mathcal{N}_v} = \frac{1}{|\mathcal{N}_v|} \sum_{u \in \mathcal{N}_v} \mathbf{h}_u$, where $\mathcal{N}_v$ depicts the neighboring nodes of v. It is important to note that the pre-trained GNN remains unchanged throughout and we only leverage the output node representations from the pre-trained GNN. Neighbor abstractions are computed solely

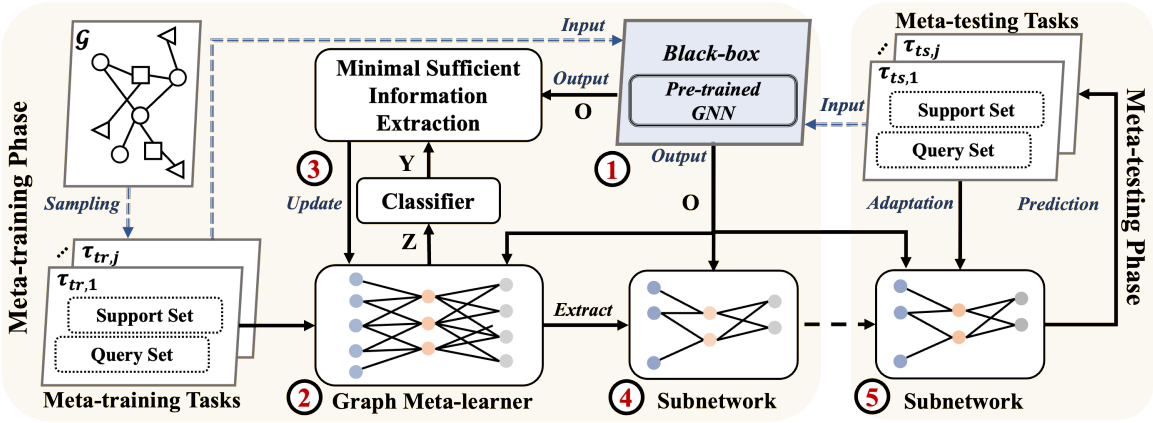

Figure 1: The step-by-step illustration of **Meta-BP**. (**1**) The black-box pre-trained GNN outputs node representations for subsequent components while remaining inaccessible itself; (**2**) Graph meta-learner built on (1) exploits both graph pre-training and meta-learning; (**3**) Graph meta-learner learns the representations $\mathbf{Z}$ to capture minimal sufficient information from the pre-trained GNN tailored to the meta-tasks; (**4**) A subnetwork is derived from the graph meta-learner during meta-training to improve generalization; (**5**) The subnetwork is anticipated to rapidly adapt to the meta-testing tasks.

once in advance, eliminating the need for adjacency multiplication during model training. Moreover, although leveraging the node representations output from pre-trained GNN layers, the method adheres to the black-box setting, since the internal parameters or internal updating mechanisms of the pre-trained model remain inaccessible.

Then we adopt linear transformation layers to introduce trainable parameters for fusing the node representation and neighbor abstraction for node v as follows:

$$\mathbf{z}_v = \sigma([\mathbf{h}_v || \mathbf{h}_{\mathcal{N}_v}]\mathbf{W}) \tag{2}$$

where $\mathbf{z}_v$ represents information extracted from the node and its neighborhood for depicting node v, $\mathbf{W}$ is a randomly initialized weight matrix, σ signifies the activation function, and $||$ denotes concatenation operation. The prediction based on $\mathrm{GML}(\cdot)$ goes as follows:

$$y_v = f_{\mathrm{C}}(\mathbf{z}_v) \quad \text{or} \quad \mathbf{Y} = f_{\mathrm{C}}(\mathbf{Z}) \tag{3}$$

where $f_{\mathrm{C}}(\cdot)$ refers to a standard classifier, and $\mathbf{Y}$ denotes the predicted node labels.

Extracting Minimal Sufficient Information

To prevent task-irrelevant knowledge in the pre-trained GNN from interfering with downstream tasks, our objective is to extract relevant knowledge from the pre-trained GNN for the downstream meta-tasks. To accomplish this, the graph meta-learner is designed to extract the minimal sufficient information tailored to few-shot node classification tasks during meta-training, i.e., to extract relevant information preserved by input variable $\mathbf{O}$ about the output variable $\mathbf{Y}$. Thus, we utilize the prediction $\mathbf{Y}$ to implicitly identify the relevant and irrelevant information within the output $\mathbf{O}$ of the pre-trained GNN. It means that an optimal representation mapping of $\mathbf{O}$ would capture the relevant features while compressing $\mathbf{O}$ by discarding the irrelevant parts that do not contribute to the prediction $\mathbf{Y}$. The graph meta-learner is thus expected to learn node representations $\mathbf{Z}$ defined in Eq. (1) as the optimal mapping of $\mathbf{O}$ w.r.t. the prediction $\mathbf{Y}$.

We adopt the information bottleneck principle (Tishby, Pereira, and Bialek 2000; Tishby and Zaslavsky 2015) and describe the relatedness between the output $\mathbf{O}$ and the node representations $\mathbf{Z}$ using mutual information $I(\mathbf{O};\mathbf{Z})$. To discard task-irrelevant knowledge from pre-trained GNN, the output $\mathbf{Z}$ from graph meta-learner is regarded as a minimal knowledge (simplest mapping) from the pre-trained GNN, yet sufficient to adapt well to meta-tasks. In other words, we can minimize the mutual information $I(\mathbf{O};\mathbf{Z})$ to obtain the simplest mapping under the constraint on $I(\mathbf{Z};\mathbf{Y})$. Namely, finding node representations $\mathbf{Z}$ with minimal sufficient information from the pre-trained GNN is formulated as the minimization of the following Lagrangian:

$$\mathcal{L}_I = \min_{\mathbf{Z}\sim\mathrm{GML}(\cdot)} I(\mathbf{O};\mathbf{Z}) - \beta I(\mathbf{Z};\mathbf{Y}) \tag{4}$$

subject to the Markov chain $\mathbf{Y} \rightarrow \mathbf{O} \rightarrow \mathbf{Z}$. Here, β is a tradeoff parameter between the complexity of the representations $I(\mathbf{O};\mathbf{Z})$ and the amount of preserved relevant information $I(\mathbf{Z};\mathbf{Y})$. The preserved relevant information $\mathbf{Z}$ from $\mathrm{GML}(\cdot)$ contributes to diminishing the uncertainty in $\mathbf{Y}$: $I(\mathbf{Z};\mathbf{Y}) = H(\mathbf{Y}) - H(\mathbf{Y}|\mathbf{Z})$, where H denotes Shannon entropy. Then Eq. (4) can be rewritten as follows:

$$\mathcal{L}_I = \min_{\mathbf{Z}\sim\mathrm{GML}(\cdot)} I(\mathbf{O};\mathbf{Z}) - \beta H(\mathbf{Y}) + \beta H(\mathbf{Y}|\mathbf{Z}) \tag{5}$$

where $H(\mathbf{Y})$ is a constant and $H(\mathbf{Y}|\mathbf{Z})$ is regarded as a cross-entropy loss denoted as $\mathcal{L}_{CE}$ for node classification by classifier $f_{\mathrm{C}}(\cdot)$. $\mathcal{L}_I$ thus encourages the graph meta-learner to leverage only relevant knowledge from $\mathbf{f}_{pre}$ that is tailored for downstream node classification. We then elaborate on the estimation of $I(\mathbf{O};\mathbf{Z})$.

Mutual Information Estimation. The computation of mutual information struggles with scaling to large sample sizes

or high-dimensional data (Paninski 2003; Gao, Ver Steeg, and Galstyan 2015). Inspired by the recent advance (Belghazi et al. 2018), we employ a neural estimator to accurately estimate $I(\mathbf{O};\mathbf{Z})$, and integrate it alongside the meta-optimization process. Specifically, mutual information is equivalent to KL divergence between the joint distribution and the product of the marginals, derived as follows:

$$\begin{aligned} &I(\mathbf{O};\mathbf{Z}) = D_{KL}(\mathbb{P}_{OZ}||\mathbb{P}_O \otimes \mathbb{P}_Z) \\ &\geq \sup_{T\in\mathcal{F}} \mathbb{E}_{\mathbb{P}_{OZ}}[T] - \log(\mathbb{E}_{\mathbb{P}_O\otimes\mathbb{P}_Z}[e^T]) = I_\Theta(\mathbf{O};\mathbf{Z}) \end{aligned} \tag{6}$$

where $\mathbb{P}_O$ and $\mathbb{P}_Z$ refer to marginals of $\mathbf{O}$ and $\mathbf{Z}$ and $\mathbb{P}_{OZ}$ denotes their joint distribution. $I_\Theta(\mathbf{O};\mathbf{Z})$ is the lower bound of $I(\mathbf{O};\mathbf{Z})$ obtained following the Donsker-Varadhan representation (Donsker and Varadhan 1983). And $\mathcal{F}$ can be any class of functions $T : \Omega \to \mathbb{R}$ satisfying the integrability constraints. By implementing T as a deep neural network, i.e., the neural estimator, lower bound $I_\Theta(\mathbf{O};\mathbf{Z})$ can be defined as a neural information measure:

$$I_\Theta(\mathbf{O};\mathbf{Z}) = \sup_{\theta\in\Theta} \mathbb{E}_{\mathbb{P}_{OZ}}[T_\theta] - \log(\mathbb{E}_{\mathbb{P}_O\otimes\mathbb{P}_Z}[e^{T_\theta}]) \tag{7}$$

where T_θ is a neural network parameterized by $\theta \in \Theta$.

Practically, the expectations can be estimated using empirical samples from $\mathbf{O}$ and $\mathbf{Z}$ as follows:

$$I(\mathbf{O};\mathbf{Z}) \approx \frac{1}{b}\sum_{i=1}^{b} T_\theta(\mathbf{o}_i,\mathbf{z}_i) - \log(\frac{1}{b}\sum_{i=1}^{b} e^{T_\theta(\mathbf{o}_i,\overline{\mathbf{z}}_i)}). \tag{8}$$

where $\{(\mathbf{o}_i,\mathbf{z}_i), ..., (\mathbf{o}_b,\mathbf{z}_b)\}$ are drawn to simulate the joint distribution $\mathbb{P}_{OZ}$ and $\{\overline{\mathbf{z}}_1, ..., \overline{\mathbf{z}}_b\}$ are randomly sampled from $\mathbf{Z}$ to simulate the marginal and b denotes the number of empirical node samples to estimate expectations. And $T_\theta(\mathbf{o},\mathbf{z})$ takes the concatenation of $\mathbf{o}$ and $\mathbf{z}$ as input. The neural estimator is well aligned with meta-optimization, as to be discussed in Optimization.

Pruning Graph Meta-learner

While the graph meta-learner extracts minimal sufficient information to diminish the influence of task-irrelevant knowledge from the pre-trained GNN, Meta-BP still adopts dense layers for the graph meta-learner to accommodate the learning of diverse meta-tasks. Due to the common over-parameterization of DNNs (Denil et al. 2013; Han et al. 2015), the graph meta-learner might still face the challenge of overfitting the insufficient meta-training tasks and lead to inferior generalization when adapted to novel tasks. Therefore, to better learn meta-training tasks for task-specific knowledge acquisition, we propose enhancing the generalization by pruning redundant/unnecessary weights from the graph meta-learner, thus finding a sparse subnetwork that holds on fast adaptation on meta-testing tasks. Inspired by the Lottery Ticket Hypothesis (LTH) (Frankle and Carbin 2019) and its capability to improve the generalization, we aim to find a subnetwork of $\mathrm{GML}(\cdot;\phi)$ by learning its model weights and binary masks (the subnetwork) together. The binary mask $\mathbf{m}^*$, which describes the subnetwork such that $|\mathbf{m}^*|$ is less than the model capacity $c\cdot|\phi|$, works as follows:

$$\begin{aligned} \mathbf{m}^* = \underset{\mathbf{m}}{\operatorname{argmin}} &\frac{1}{n}\sum_{i=1}^{n}(\mathcal{L}_{CE}(f_{\mathrm{C}}(\mathrm{GML}(\mathbf{o}_i,\mathcal{G};\phi\odot\mathbf{m})),\hat{y}_i) - C) \\ &\text{subject to} \quad |\mathbf{m}^*| \leq c\cdot|\phi|, \end{aligned} \tag{9}$$

where $C = \mathcal{L}_{CE}(f_{\mathrm{C}}(\mathrm{GML}(\mathbf{o}_i,\mathcal{G};\phi)),\hat{y}_i)$ and $c\cdot|\phi| \ll |\phi|$. c denotes the model capacity ratio in %. To enable backpropagation, $\mathbf{m}^*$ is obtained via a continuous learnable mask $\mathbf{s}$. Concretely, $\mathbf{m}$ is determined as top-c scores of $\mathbf{s}$ where $\mathbf{s}$ can be updated through gradient descent using Straight-through Estimator (Bengio, Léonard, and Courville 2013; Ramanujan et al. 2020). Note that it is not necessary to prune T_θ because only the subnetwork of $\mathrm{GML}(\cdot;\phi)$ is transferred to the meta-testing phase. Then we discuss the learning of the subnetwork in the subsequent discussion.

Optimization

To benefit from the relevant information in the pre-trained GNN, as well as the meta-knowledge of task adaptation based on meta-training tasks, we perform minimal sufficient information extraction along with the meta-optimization process. Specifically, given a batch of meta-training tasks $\{\tau_1,\tau_2,\ldots,\tau_B\}$, $\mathcal{L}_I$ in Eq. (5) can be derived combined with Eq. (8) as follows:

$$\begin{aligned} \mathcal{L}_I = \min_{\theta,\phi} \frac{1}{B}\sum_{j=1}^{B}\{&\frac{1}{b}\sum_{i=1}^{b} T_\theta(\mathbf{o}_i,\mathbf{z}_i) - \log(\frac{1}{b}\sum_{i=1}^{b} e^{T_\theta(\mathbf{o}_i,\overline{\mathbf{z}}_i)}) \\ &+ \frac{\beta}{b}\sum_{i=1}^{b}\mathcal{L}_{CE}(f_{\mathrm{C}}(\mathrm{GML}(\mathbf{o}_i,\mathcal{G};\phi_j)),\hat{y}_i)\}. \end{aligned} \tag{10}$$

where $\phi_j = \phi - \lambda\nabla\mathcal{L}_{CE}^j$, with λ denoting the inner update rate and $\mathcal{L}_{CE}^j$ computed on support set Ω_j^s of each meta-task τ_j. We use the query set Ω_j^q of size b as the empirical samples. In the inner loop, $\mathrm{GML}(\cdot;\phi)$ is updated in a task-specific manner regarding each meta-task while $T_\theta(\cdot)$ keeps the same across the batch of meta-tasks, thus ensuring consistent mutual information estimation across all meta-tasks.

In addition, we extract the subnetwork from $\mathrm{GML}(\cdot;\phi)$ during meta-optimization based on Eq. (9) as follows:

$$\begin{aligned} \mathcal{L}_S = \min_{\mathbf{m}} \frac{1}{B}\sum_{j=1}^{B}\{&\frac{1}{b}\sum_{i=1}^{b}(\mathcal{L}_{CE}(f_{\mathrm{C}}(\mathrm{GML}(\mathbf{o}_i,\mathcal{G};\phi_j\odot\mathbf{m})),\hat{y}_i) \\ &- C_j)\}. \end{aligned} \tag{11}$$

where $C_j = \mathcal{L}_{CE}(f_{\mathrm{C}}(\mathrm{GML}(\mathbf{o}_i,\mathcal{G};\phi_j)),\hat{y}_i)$, evaluated on the query set Ω_j^q of τ_j. Likewise, $\mathbf{m}$ remains unchanged during the inner loop for the batch of meta-tasks. Ultimately, the meta-optimization objective is formulated as follows:

$$\mathcal{L}_{Meta} = \mathcal{L}_I + \alpha\mathcal{L}_S \tag{12}$$

where α is the tradeoff weight. Consequently, $\mathrm{GML}(\cdot;\phi)$ undergoes optimization-based meta-training with Eq. (12) for a sufficient number of iterations. Following this, only the extracted subnetwork is utilized for meta-testing tasks as the model initialization for fast adaptation.

Experiments

Experimental Setup

Dataset. We leverage four real-world graph datasets for experimental evaluation following previous works (Zhou

Methods	Cora	Computers	Cora-full		OGBN-arxiv	
	2-way	3-way	5-way	10-way	5-way	10-way
			1-shot			
GCN	$55.21_{(\pm 5.64)}$	$37.33_{(\pm 3.91)}$	$43.75_{(\pm 2.92)}$	$31.26_{(\pm 3.29)}$	$54.17_{(\pm 5.31)}$	$37.80_{(\pm 5.44)}$
GraphSage	$58.33_{(\pm 5.22)}$	$39.98_{(\pm 5.17)}$	$44.26_{(\pm 2.64)}$	$32.54_{(\pm 3.53)}$	$52.33_{(\pm 5.29)}$	$35.20_{(\pm 5.87)}$
GMI	$60.25_{(\pm 4.32)}$	$62.28_{(\pm 4.92)}$	$56.81_{(\pm 1.42)}$	$40.98_{(\pm 1.72)}$	$55.92_{(\pm 4.23)}$	$38.74_{(\pm 4.92)}$
DGI	$61.56_{(\pm 4.46)}$	$64.52_{(\pm 5.03)}$	$56.52_{(\pm 1.53)}$	$40.36_{(\pm 1.76)}$	$55.63_{(\pm 5.02)}$	$39.82_{(\pm 5.11)}$
PN	$52.60_{(\pm 5.23)}$	$47.63_{(\pm 5.23)}$	$48.25_{(\pm 1.80)}$	$35.65_{(\pm 1.98)}$	$53.26_{(\pm 3.94)}$	$34.67_{(\pm 4.07)}$
MAML	$53.66_{(\pm 4.92)}$	$66.05_{(\pm 5.16)}$	$58.38_{(\pm 2.01)}$	$38.72_{(\pm 2.19)}$	$55.21_{(\pm 4.13)}$	$39.26_{(\pm 4.21)}$
Meta-SGC	$57.72_{(\pm 5.99)}$	$67.40_{(\pm 6.79)}$	$61.34_{(\pm 4.53)}$	$41.29_{(\pm 4.06)}$	$54.90_{(\pm 5.08)}$	$41.00_{(\pm 5.40)}$
GPN	$57.22_{(\pm 4.20)}$	$63.78_{(\pm 5.27)}$	$54.36_{(\pm 2.29)}$	$43.27_{(\pm 1.92)}$	$54.22_{(\pm 5.22)}$	$37.42_{(\pm 5.18)}$
G-Meta	$62.24_{(\pm 4.93)}$	$67.22_{(\pm 5.61)}$	$55.21_{(\pm 2.15)}$	$46.23_{(\pm 1.79)}$	$52.73_{(\pm 5.15)}$	$41.29_{(\pm 5.37)}$
TLP	$61.11_{(\pm 3.14)}$	$65.05_{(\pm 5.40)}$	$61.28_{(\pm 2.41)}$	$48.12_{(\pm 1.53)}$	$53.94_{(\pm 4.36)}$	$39.75_{(\pm 4.23)}$
TENT	$61.25_{(\pm 5.15)}$	$66.24_{(\pm 5.24)}$	$62.42_{(\pm 2.16)}$	$47.95_{(\pm 1.88)}$	$\underline{57.32}_{(\pm 4.91)}$	$\underline{42.56}_{(\pm 4.17)}$
TEG	$\underline{63.14}_{(\pm 4.43)}$	$\underline{67.58}_{(\pm 5.11)}$	$\underline{63.73}_{(\pm 2.08)}$	$\underline{48.36}_{(\pm 2.03)}$	$57.09_{(\pm 5.37)}$	$41.89_{(\pm 4.74)}$
Meta-BP	$\mathbf{66.38}_{(\pm 5.01)}$	$\mathbf{69.35}_{(\pm 4.28)}$	$\mathbf{66.05}_{(\pm 1.46)}$	$\mathbf{51.41}_{(\pm 1.91)}$	$\mathbf{59.06}_{(\pm 4.20)}$	$\mathbf{43.78}_{(\pm 4.39)}$
			3-shot			
GCN	$61.98_{(\pm 4.77)}$	$54.58_{(\pm 7,89)}$	$50.23_{(\pm 3.25)}$	$35.25_{(\pm 3.41)}$	$60.28_{(\pm 5.77)}$	$42.60_{(\pm 6.11)}$
GraphSage	$65.39_{(\pm 6.32)}$	$52.63_{(\pm 5.72)}$	$51.64_{(\pm 2.63)}$	$36.36_{(\pm 2.69)}$	$59.22_{(\pm 6.23)}$	$40.17_{(\pm 6.39)}$
GMI	$62.79_{(\pm 4.01)}$	$65.36_{(\pm 4.86)}$	$61.92_{(\pm 1.49)}$	$51.22_{(\pm 1.79)}$	$64.74_{(\pm 5.04)}$	$43.28_{(\pm 5.26)}$
DGI	$63.52_{(\pm 4.18)}$	$67.43_{(\pm 4.91)}$	$60.37_{(\pm 1.56)}$	$49.25_{(\pm 1.88)}$	$65.96_{(\pm 5.31)}$	$44.56_{(\pm 5.71)}$
PN	$62.31_{(\pm 5.21)}$	$60.22_{(\pm 5.19)}$	$53.72_{(\pm 1.98)}$	$37.69_{(\pm 2.31)}$	$61.73_{(\pm 4.09)}$	$42.67_{(\pm 4.65)}$
MAML	$56.77_{(\pm 4.38)}$	$69.26_{(\pm 5.25)}$	$63.18_{(\pm 2.67)}$	$52.64_{(\pm 3.02)}$	$62.22_{(\pm 4.19)}$	$52.60_{(\pm 4.72)}$
Meta-SGC	$59.64_{(\pm 5.48)}$	$70.91_{(\pm 7.29)}$	$67.31_{(\pm 2.53)}$	$54.82_{(\pm 3.51)}$	$65.85_{(\pm 5.52)}$	$51.73_{(\pm 5.61)}$
GPN	$64.28_{(\pm 4.22)}$	$68.82_{(\pm 5.11)}$	$62.85_{(\pm 1.42)}$	$50.75_{(\pm 1.94)}$	$62.23_{(\pm 4.67)}$	$46.68_{(\pm 4.99)}$
G-Meta	$62.47_{(\pm 4.63)}$	$72.08_{(\pm 5.89)}$	$69.16_{(\pm 1.91)}$	$54.21_{(\pm 2.63)}$	$63.15_{(\pm 5.28)}$	$51.32_{(\pm 5.93)}$
TLP	$\underline{73.38}_{(\pm 5.29)}$	$73.28_{(\pm 4.41)}$	$64.53_{(\pm 1.74)}$	$52.76_{(\pm 2.45)}$	$62.58_{(\pm 5.59)}$	$43.16_{(\pm 5.14)}$
TENT	$65.43_{(\pm 4.36)}$	$71.32_{(\pm 5.73)}$	$67.59_{(\pm 1.60)}$	$55.21_{(\pm 2.27)}$	$66.37_{(\pm 5.01)}$	$52.07_{(\pm 5.46)}$
TEG	$68.28_{(\pm 4.57)}$	$\underline{73.54}_{(\pm 5.26)}$	$\underline{69.75}_{(\pm 1.72)}$	$\underline{55.83}_{(\pm 1.73)}$	$\underline{66.49}_{(\pm 4.87)}$	$\underline{53.64}_{(\pm 5.12)}$
Meta-BP	$\mathbf{75.29}_{(\pm 4.21)}$	$\mathbf{75.14}_{(\pm 4.16)}$	$\mathbf{72.98}_{(\pm 1.86)}$	$\mathbf{57.79}_{(\pm 2.16)}$	$\mathbf{69.03}_{(\pm 5.18)}$	$\mathbf{55.98}_{(\pm 5.03)}$

Table 1: Few-shot node classification accuracy (%) on multiple datasets.

et al. 2019; Wu et al. 2022), including Cora (Yang, Cohen, and Salakhudinov 2016), Amazon Computers (Zhang et al. 2022b), Cora-full (Bojchevski and Günnemann 2018), and OGBN-arxiv (Hu et al. 2020a). For dataset splitting (train/-val/test), we used ratios of 3/2/2 for Cora, 4/3/3 for Computers, 25/20/25 for Cora-Full, and 20/10/10 for OGBN-Arxiv.

Baselines. To validate the effectiveness of our proposed Meta-BP, we compare it with baselines in three groups: 1) Graph Neural Networks including **GCN** (Kipf and Welling 2017) and **GraphSage** (Hamilton, Ying, and Leskovec 2017); 2) Graph pre-training including **GMI** (Peng et al. 2020), and **DGI** (Velickovic et al. 2019); 3) Graph few-shot learning models such as conventional **PN** (Snell, Swersky, and Zemel 2017) and **MAML** (Zhou et al. 2019); Besides, we compare Meta-BP to **Meta-SGC** (Zhou et al. 2019), **GPN** (Liu et al. 2019), **G-Meta** (Huang and Zitnik 2020), **TLP** (Tan et al. 2022b), **TENT** (Wang et al. 2022), and **TEG** (Kim et al. 2023). Note that baseline approaches except TLP focus on the *white-box setting* and are unable to perform in the black-box setting.

Reproducibility Details. Following common practice in few-shot learning, we perform random class splitting to form the training, testing, and validation classes for each run. During each run, model performance is evaluated on 500 random few-shot tasks considering small support sets, obtaining the average accuracy per run. To account for the randomness of class splitting, we conduct four random runs for each N-way K-shot problem and keep class splits consistent across all models. Average scores and standard deviations across runs are reported. We implement Meta-BP in PyTorch with an NVIDIA Tesla V100 GPU and use a two-layer DGI of 256 hidden units as the black-box pre-trained GNN, while most baselines, which cannot handle the black-box setting, are allowed for GNN parameter updating. Dimensions of the learnable transformation layer in GML upon node representations are determined via a grid search over $\{4, 8, 32, 64, 128\}$. The neural estimator is established as a two-layer MLP with 64 units. β is 1.0 for the information bottleneck regularization and α is 0.1 for meta-optimization. Learning rates of all models are searched from {0.01, 0.005, 0.001, 0.0005, 0.0001}. MAML-based approaches including Meta-BP adopt two fast updates with a step size of 0.05, except that on Amazon Computers it applies 0.01 as the step size. Implementation can be found at https://github.com/repograph/metabp.

Experimental Results

Overall Performance. The performance of Meta-BP and baselines are presented in Table 1, where the best results are highlighted in **bold** and the best baseline results are underlined. From the results, we find that Meta-BP achieves the best performance in all settings. Other observations are discussed as follows. First, GCN and GraphSage generally exhibit inferior performance compared to graph pre-training models such as GMI and DGI. It can be attributed to their training from scratch, whereas pre-trained parameters could convey useful knowledge about the graph. Second, graph few-shot learning methods consistently outperform GCN

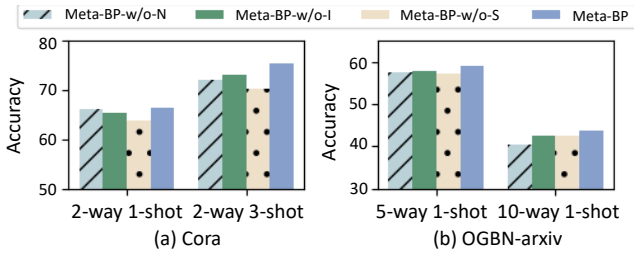

Figure 2: Few-shot node classification on Cora and OGBN-arxiv with different variants of Meta-BP.

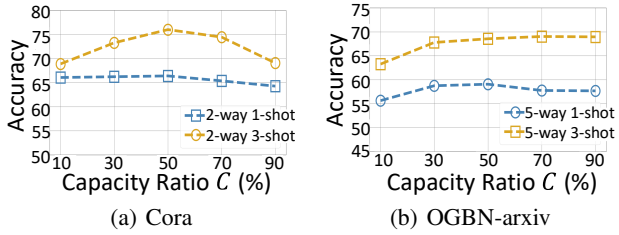

Figure 3: Few-shot node classification on Cora and OGBN-arxiv with varying capacity ratios.

Methods	Cora-full			
	5-way 1-shot	5-way 3-shot	10-way 1-shot	10-way 3-shot
Meta-BP-GMI	$66.13_{(\pm 1.48)}$	$75.58_{(\pm 1.23)}$	$51.88_{\pm(1.44)}$	$62.28_{(\pm 1.86)}$
Meta-BP-BGRL	$65.92_{(\pm 1.62)}$	$73.64_{(\pm 1.42)}$	$51.36_{(\pm 1.65)}$	$59.47_{(\pm 1.94)}$
Meta-BP-DGI	$66.05_{(\pm 1.46)}$	$72.98_{(\pm 1.86)}$	$51.41_{(\pm 1.91)}$	$57.79_{(\pm 2.16)}$

Table 2: The accuracy (%) of few-shot node classification with different pre-trained models.

and GraphSage. This suggests that few-shot approaches acquire meta-knowledge regarding task adaptation, contrasting with conventional supervised training. Last, Meta-BP achieves the best performance in all cases. This underscores Meta-BP's ability to leverage pertinent information from black-box pre-trained GNN and also task-specific knowledge for rapid adaptation. The proposed minimum sufficient information extraction and the graph meta-learner pruning work jointly to promote the inference on novel classes. Note that only TLP and Meta-BP operate in a black-box setting, requiring no access to parameters or gradients in the pre-trained models. However, Meta-BP outperforms TLP, which struggles to handle task-irrelevant information from pre-trained GNNs and task-specific knowledge. Considering class splitting randomness, experimental evaluation utilizing various class partitioning offers a better assessment of the performances.

Ablation Study. We analyze the effectiveness of different components in Meta-BP, aiming to answer the following questions: (RQ1) Do neighbor abstractions improve graph meta-learner by utilizing topological information? (RQ2) Does minimal sufficient information extraction facilitate few-shot node classification? (RQ3) How does subnetwork extraction affect the few-shot performance? We conduct ablation studies with three variants of Meta-BP, including: (a) **Meta-BP-w/o-N** that only makes use of the node information without neighbor abstractions from the black-box pre-trained GNN; (b) **Meta-BP-w/o-I** that does not apply minimal sufficient information extraction; (c) **Meta-BP-w/o-S** that utilizes a full set of parameters of GML for meta-testing, rather than the extracted subnetwork. The performance of the variants is depicted in Figure 2, from which we find the answers to the above questions. RQ1: The inferior performance of Meta-BP-w/o-N compared to Meta-BP suggests that incorporating graph structural information via neighbor abstractions enhances few-shot node classification. RQ2: Meta-BP-w/o-I performs worse compared to Meta-BP, implying that extracting relevant information from the black-box pre-trained GNN helps obtain more accurate node representations for classification tasks. RQ3: Meta-BP-w/o-S is outperformed by Meta-BP, showing the advantages of pruning graph meta-learner, which leads to enhanced generalizability on novel tasks.

Impact of Varying Capacity Ratios. We analyze the impact of different capacity ratio values c, employed in the subnetwork extraction from graph meta-learner. As shown in Figure 3, we observe that excessively small values of c can lead to inferior performance probably due to insufficient model capacity. As the capacity ratio c grows, the model performance initially increases and then decreases. It suggests that extracting a subnetwork from GML with an appropriate capacity ratio can enhance model performance by avoiding potential overfitting on the meta-training tasks.

Impact of Pre-trained Models. Meta-BP aims to integrate meta-learning with flexible black-box pre-trained GNNs. In our experiments, We employ DGI as the pre-trained GNN. Here, we examine the impact of various pre-trained GNNs to validate Meta-BP's effectiveness. Table 2 presents the performance of Meta-BP associated with black-box GMI (Peng et al. 2020), BGRL (Thakoor et al. 2021) and DGI (Velickovic et al. 2019), respectively. It is shown that Meta-BP with different pre-training strategies performs differently, which depends on the effectiveness of pre-training strategies and demonstrates the capability and versatility of Meta-BP in effectively learning from different pre-trained GNNs and extracting relevant knowledge for few-shot node classification. It is important to emphasize that our work underscores the ability to learn from black-box pre-trained GNNs, focusing on the utilization of any existing pre-trained models.

Conclusion

In this paper, we study graph few-shot learning and explore the integration of meta-learning and black-box pre-trained GNNs. Specifically, we devise a graph meta-learner to bridge the pre-trained GNN and downstream tasks to enable effective utilization of the black-box pre-trained GNN. We then optimize the graph meta-learner to extract only relevant knowledge from the pre-trained GNN to facilitate the downstream few-shot node classification tasks. Furthermore, we introduce pruning to graph meta-learner to enhance adaptation ability on novel tasks. Extensive experiments validate the effectiveness of our proposed framework.

Acknowledgments

This research was supported by the Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 1 grant (22-SIS-SMU-054). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Ministry of Education, Singapore.

References

Belghazi, M. I.; Baratin, A.; Rajeshwar, S.; Ozair, S.; Bengio, Y.; Courville, A.; and Hjelm, D. 2018. Mutual information neural estimation. In *ICML*.

Bengio, Y.; Léonard, N.; and Courville, A. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Bojchevski, A.; and Günnemann, S. 2018. Deep Gaussian Embedding of Graphs: Unsupervised Inductive Learning via Ranking. In *ICLR*.

Bose, A. J.; Jain, A.; Molino, P.; and Hamilton, W. L. 2019. Meta-graph: Few shot link prediction via meta learning. *arXiv preprint arXiv:1912.09867*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. In *NeurIPS*.

Chen, H.; Yin, H.; Wang, W.; Wang, H.; Nguyen, Q. V. H.; and Li, X. 2018. PME: projected metric embedding on heterogeneous networks for link prediction. In *KDD*.

Chen, W.-Y.; Liu, Y.-C.; Kira, Z.; Wang, Y.-C. F.; and Huang, J.-B. 2019. A Closer Look at Few-shot Classification. In *ICLR*.

Denil, M.; Shakibi, B.; Dinh, L.; Ranzato, M.; and De Freitas, N. 2013. Predicting parameters in deep learning. In *NeurIPS*.

Ding, K.; Wang, J.; Li, J.; Shu, K.; Liu, C.; and Liu, H. 2020. Graph prototypical networks for few-shot learning on attributed networks. In *CIKM*.

Ding, K.; Zhou, Q.; Tong, H.; and Liu, H. 2021. Few-shot Network Anomaly Detection via Cross-network Meta-learning. In *WWW*.

Donsker, M. D.; and Varadhan, S. S. 1983. Asymptotic evaluation of certain Markov process expectations for large time. IV. *Communications on pure and applied mathematics*.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*.

Frankle, J.; and Carbin, M. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *ICLR*.

Gao, S.; Ver Steeg, G.; and Galstyan, A. 2015. Efficient estimation of mutual information for strongly dependent variables. In *AISTATS*.

Gaudelet, T.; Day, B.; Jamasb, A. R.; Soman, J.; Regep, C.; Liu, G.; Hayter, J. B.; Vickers, R.; Roberts, C.; Tang, J.; et al. 2021. Utilizing graph machine learning within drug discovery and development. *Briefings in bioinformatics*.

Guo, Z.; Zhang, C.; Yu, W.; Herr, J.; Wiest, O.; Jiang, M.; and Chawla, N. V. 2021. Few-Shot Graph Learning for Molecular Property Prediction. In *WWW*.

Hamilton, W. L.; Ying, R.; and Leskovec, J. 2017. Inductive representation learning on large graphs. In *NeurIPS*.

Han, S.; Pool, J.; Tran, J.; and Dally, W. 2015. Learning both weights and connections for efficient neural network. In *NeurIPS*.

Hospedales, T.; Antoniou, A.; and Micaelli, P. 2020. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*.

Hu, W.; Fey, M.; Zitnik, M.; Dong, Y.; Ren, H.; Liu, B.; Catasta, M.; and Leskovec, J. 2020a. Open graph benchmark: Datasets for machine learning on graphs. In *NeurIPS*.

Hu, W.; Liu, B.; Gomes, J.; Zitnik, M.; Liang, P.; Pande, V.; and Leskovec, J. 2020b. Strategies For Pre-training Graph Neural Networks. In *ICLR*.

Huang, K.; and Zitnik, M. 2020. Graph Meta Learning via Local Subgraphs. In *NeurIPS*.

Jiang, X.; Jia, T.; Fang, Y.; Shi, C.; Lin, Z.; and Wang, H. 2021. Pre-training on large-scale heterogeneous graph. In *KDD*.

Jin, W.; Derr, T.; Liu, H.; Wang, Y.; Wang, S.; Liu, Z.; and Tang, J. 2020. Self-supervised Learning on Graphs: Deep Insights and New Direction. arXiv:2006.10141.

Kim, S.; Lee, J.; Lee, N.; Kim, W.; Choi, S.; and Park, C. 2023. Task-equivariant graph few-shot learning. In *KDD*.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

Li, H.; Eigen, D.; Dodge, S.; Zeiler, M.; and Wang, X. 2019. Finding task-relevant features for few-shot learning by category traversal. In *CVPR*.

Li, Z.; Zhang, H.; Zhang, Q.; Kou, Z.; and Pei, S. 2024. Learning from Novel Knowledge: Continual Few-shot Knowledge Graph Completion. In *CIKM*.

Li, Z.; Zhou, F.; Chen, F.; and Li, H. 2017. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*.

Liu, J.; Yang, C.; Lu, Z.; Chen, J.; Li, Y.; Zhang, M.; Bai, T.; Fang, Y.; Sun, L.; Yu, P. S.; et al. 2023a. Towards graph foundation models: A survey and beyond. *arXiv preprint arXiv:2310.11829*.

Liu, L.; Zhou, T.; Long, G.; Jiang, J.; and Zhang, C. 2019. Learning to propagate for graph meta-learning. In *NeurIPS*.

Liu, P.; Yuan, W.; Fu, J.; Jiang, Z.; Hayashi, H.; and Neubig, G. 2023b. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*.

Liu, Y.; Huang, L.; Cao, B.; Li, X.; Giunchiglia, F.; Feng, X.; and Guan, R. 2024. A simple but effective approach for unsupervised few-shot graph classification. In *WWW*.

Liu, Y.; Li, M.; Li, X.; Giunchiglia, F.; Feng, X.; and Guan, R. 2022. Few-shot node classification on attributed networks with graph meta-learning. In *SIGIR*.

Lu, B.; Gan, X.; Yang, L.; Zhang, W.; Fu, L.; and Wang, X. 2022. Geometer: Graph Few-Shot Class-Incremental Learning via Prototype Representation. In *KDD*.

Paninski, L. 2003. Estimation of entropy and mutual information. *Neural computation*.

Pei, S.; Kou, Z.; Zhang, Q.; and Zhang, X. 2023. Few-shot low-resource knowledge graph completion with multi-view task representation generation. In *KDD*.

Peng, Z.; Huang, W.; Luo, M.; Zheng, Q.; Rong, Y.; Xu, T.; and Huang, J. 2020. Graph representation learning via graphical mutual information maximization. In *WWW*.

Qiu, J.; Chen, Q.; Dong, Y.; Zhang, J.; Yang, H.; Ding, M.; Wang, K.; and Tang, J. 2020. Gcc: Graph contrastive coding for graph neural network pre-training. In *KDD*.

Qu, M.; Gao, T.; Xhonneux, L.-P. A.; and Tang, J. 2020. Few-shot Relation Extraction via Bayesian Meta-learning on Relation Graphs. In *ICML*.

Ramanujan, V.; Wortsman, M.; Kembhavi, A.; Farhadi, A.; and Rastegari, M. 2020. What's hidden in a randomly weighted neural network? In *CVPR*.

Sablayrolles, A.; Douze, M.; Schmid, C.; Ollivier, Y.; and Jégou, H. 2019. White-box vs black-box: Bayes optimal strategies for membership inference. In *International Conference on Machine Learning*, 5558–5567. PMLR.

Snell, J.; Swersky, K.; and Zemel, R. 2017. Prototypical networks for few-shot learning. In *NeurIPS*.

Sun, F.-Y.; Hoffmann, J.; Verma, V.; and Tang, J. 2019. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. *arXiv preprint arXiv:1908.01000*.

Sun, M.; Zhou, K.; He, X.; Wang, Y.; and Wang, X. 2022. Gppt: Graph pre-training and prompt tuning to generalize graph neural networks. In *KDD*.

Sun, X.; Zhang, J.; Wu, X.; Cheng, H.; Xiong, Y.; and Li, J. 2023. Graph prompt learning: A comprehensive survey and beyond. *arXiv preprint arXiv:2311.16534*.

Sun, Y.; Wang, S.; Feng, S.; Ding, S.; Pang, C.; Shang, J.; Liu, J.; Chen, X.; Zhao, Y.; Lu, Y.; et al. 2021. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *arXiv preprint arXiv:2107.02137*.

Tan, Z.; Ding, K.; Guo, R.; and Liu, H. 2022a. Graph few-shot class-incremental learning. In *WSDM*.

Tan, Z.; Guo, R.; Ding, K.; and Liu, H. 2023. Virtual Node Tuning for Few-shot Node Classification. In *KDD*.

Tan, Z.; Wang, S.; Ding, K.; Li, J.; and Liu, H. 2022b. Transductive Linear Probing: A Novel Framework for Few-Shot Node Classification. In *LoG*.

Tang, L.; and Liu, H. 2010. Graph mining applications to social network analysis. *Managing and mining graph data*.

Thakoor, S.; Tallec, C.; Azar, M. G.; Munos, R.; Veličković, P.; and Valko, M. 2021. Bootstrapped representation learning on graphs. In *ICLR 2021 Workshop on Geometrical and Topological Representation Learning*.

Tishby, N.; Pereira, F. C.; and Bialek, W. 2000. The information bottleneck method. *arXiv preprint physics/0004057*.

Tishby, N.; and Zaslavsky, N. 2015. Deep learning and the information bottleneck principle. In *2015 IEEE information theory workshop (ITW)*.

Velickovic, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2019. Deep graph infomax. In *ICLR*.

Wang, N.; Luo, M.; Ding, K.; Zhang, L.; Li, J.; and Zheng, Q. 2020. Graph Few-shot Learning with Attribute Matching. In *CIKM*.

Wang, S.; Ding, K.; Zhang, C.; Chen, C.; and Li, J. 2022. Task-adaptive few-shot node classification. In *KDD*.

Wang, S.; Dong, Y.; Ding, K.; Chen, C.; and Li, J. 2023. Few-shot node classification with extremely weak supervision. In *WSDM*.

Wang, Y.; Abuduweili, A.; Yao, Q.; and Dou, D. 2021. Property-aware relation networks for few-shot molecular property prediction. In *NeurIPS*.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Philip, S. Y. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*.

Wu, Z.; Zhou, P.; Wen, G.; Wan, Y.; Ma, J.; Cheng, D.; and Zhu, X. 2022. Information Augmentation for Few-shot Node Classification. In *IJCAI*.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How powerful are graph neural networks? In *ICLR*.

Yang, Z.; Cohen, W.; and Salakhudinov, R. 2016. Revisiting semi-supervised learning with graph embeddings. In *ICML*.

Yao, H.; Huang, L.-K.; Zhang, L.; Wei, Y.; Tian, L.; Zou, J.; Huang, J.; et al. 2021. Improving generalization in meta-learning via task augmentation. In *ICML*.

Yao, H.; Zhang, C.; Wei, Y.; Jiang, M.; Wang, S.; Huang, J.; Chawla, N.; and Li, Z. 2020. Graph few-shot learning via knowledge transfer. In *AAAI*.

Yoon, S. W.; Seo, J.; and Moon, J. 2019. Tapnet: Neural network augmented with task-adaptive projection for few-shot learning. In *ICML*.

You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; and Shen, Y. 2020. Graph contrastive learning with augmentations. In *NeurIPS*.

Zhang, Q.; Pei, S.; Yang, Q.; Zhang, C.; Chawla, N. V.; and Zhang, X. 2023. Cross-domain few-shot graph classification with a reinforced task coordinator. In *AAAI*.

Zhang, Q.; Wu, X.; Yang, Q.; Zhang, C.; and Zhang, X. 2022a. Few-shot Heterogeneous Graph Learning via Cross-domain Knowledge Transfer. In *KDD*.

Zhang, S.; Liu, Y.; Sun, Y.; and Shah, N. 2022b. Graph-less neural networks: Teaching old mlps new tricks via distillation. In *ICLR*.

Zhou, F.; Cao, C.; Zhang, K.; Trajcevski, G.; Zhong, T.; and Geng, J. 2019. Meta-gnn: On few-shot node classification in graph meta-learning. In *CIKM*.