FEW-SHOT NODE PROMPT TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite the powerful representation ability of GNNs, recent works have demonstrated that the performance of GNNs can severely degrade when the number of labeled nodes is limited in training data. *Few-shot Node Classification* is one of the problems with an extreme shortage of node labels and has drawn growing attention lately. The current modus operandi, i.e., meta-learning, has succeeded in transferring the structural knowledge learned from *base classes* with abundant labeled nodes to few-shot *novel classes*. However, for real-world scenarios, it is often the case that all the classes on the graph have limited labeled nodes, thus meta-learning cannot be directly deployed. In this work, we generalize the few-shot node classification by removing the assumption that there exist abundant labeled nodes for the base classes. In the meantime, we propose a novel *Few-shot Node Prompt Tuning* method to effectively elicit substantial prior knowledge in the input graph for solving few-shot node classification tasks without labeled base classes. Specifically, we fix a pretrained graph transformer as the encoder and inject virtual nodes as soft prompts in the embedding space to bridge the gap of training objectives between the pretexts and downstream few-shot node classification tasks. Such prompts are small tensors and can be efficiently optimized with a simple classifier corresponding to the few labeled nodes. Since a single pretrained encoder is shared across different tasks, the proposed method retains the efficiency and potential for the model ensemble. Extensive experiments on four prevalent node classification datasets show that the proposed method, FS-NPT, is an efficient and effective way to tackle the general few-shot node classification problem. Our implementation is released[1].

1 INTRODUCTION

With the rising of deep learning, various Graph Neural Networks (GNNs) (Kipf & Welling, 2017; Hamilton et al., 2017; Veličković et al., 2018; Xu et al., 2019) have been proposed for effective graph representation learning. However, recent works (Zhang et al., 2018; Ding et al., 2020; Wang et al., 2020b) find that the performance of those GNNs will degrade severely when the number of labeled nodes is insufficient. Such a label scarcity issue prevailingly exists in large graphs where the cost for manual data collection and labeling is overwhelming. This challenge has led to a proliferation of studies (Huang & Zitnik, 2020; Lan et al., 2020; Wang et al., 2022) that try to learn fast-adaptable GNNs for unseen tasks with extremely scarce ground-truth labels, i.e., *Few-Shot Node Classification* (FSNC) tasks. Conventionally, if the target FSNC task contains N classes with K labeled nodes per class, the problem is denoted as an N-way K-shot node classification task, where the K labeled nodes are termed as a *support set*, and the unlabeled nodes are termed as a *query set* for evaluation.

The current modus operandi, i.e., *meta-learning*, has become a predominate and successful paradigm to tackle the issue of label shortage on graphs (Zhang et al., 2018; Ding et al., 2020; Huang & Zitnik, 2020; Wang et al., 2022). Besides the target node classes (termed as *novel classes*) with few labeled nodes, meta-learning based methods assume the existence of a set of *base classes*, which is disjoint with the novel classes set and has substantial labeled nodes in each class to sample a number of meta-tasks, or episodes, to train the GNN model while emulating the target N-way K-shot task structure. This emulation-based training has been proved helpful for fast adaptation to target FSNC tasks (Lan et al., 2020; Wang et al., 2020b). Despite astonishing breakthroughs having been made, Tan et al. (2022) firstly points out that those meta-learning based methods suffer from the piecemeal graph

[1] https://github.com/Anonymous-submit-23/FS-NPT.git

knowledge issue, which implies that only a small portion of nodes are involved in each episode, thus hindering the generalizability of the learned GNN models regarding unseen novel classes. Moreover, the assumption of the existence of disjoint base and novel classes could be untenable for real-world graphs where there might be no base class (e.g., all the classes on the given graph only contain a few labeled nodes, and we want to do node classification on all classes). In a nutshell, meta-learning is a successful method for FSNC tasks, but also has its own limitations in effectiveness and applicability.

Based on these limitations in the existing efforts, in this work, we first relax the assumption in the traditional definition of FSNC tasks to cover the scenarios where there could be no base classes at all (e.g., all the classes on the given graph only contain a few labeled nodes, and node classification is performed over all classes, or the number of base classes is very small). Then, to facilitate sufficient training for obtaining a powerful graph encoder, we choose *Graph Transformers* (GTs) (Zhang et al., 2020; Chen et al., 2022) as the encoder to learn representative node embeddings. Recently, large transformer-based (Vaswani et al., 2017) models have thrived in various domains, such as languages (Devlin et al., 2018), images (Dosovitskiy et al., 2020), as well as graphs (Hu et al., 2020b). Compared to traditional GNNs, GTs have a much larger magnitude of parameter numbers, which have shown unique advantages in modeling graph data and acquiring structural knowledge (Zhang et al., 2020; Chen et al., 2022). Furthermore, pretrained in an unsupervised manner, GTs can learn from a large number of unlabeled nodes by enforcing the model to learn from pre-defined pretext tasks (e.g. masked link restoration, masked node recovery, etc.) (Zhang et al., 2020; Hu et al., 2020b). In other words, no node label information is needed for obtaining pretrained GTs enriched with topological and semantic knowledge. Thus, the pretraining does not need labeled nodes from base classes. However, following Tan et al. (2022), we conduct an experiment in Section 4.3, which shows that directly transferring node embedding from GTs and fine-tuning another classifier on the support set will lead to unsatisfactory performance on the corresponding FSNC task. This is because directly transferring node embeddings neglects the inherent gap between the training objective of the pretexts and that of the downstream FSNC tasks. Also, naive fine-tuning with the few labeled nodes will lead to severe overfitting. Both these two factors can render the transferred node embeddings sub-optimal for target FSNC tasks. Accordingly, to elicit the learned substantial graph knowledge from GTs with only a few labels from each target task, we propose a method, *Few-shot Node Prompt Tuning* (FS-NPT), that can efficiently modulate the GTs to customize the pretrained node embeddings for different FSNC tasks.

With the recent developments in natural language processing (NLP), *prompting* has become a new fashion to adapt large-scale transformer-based language models to new few-shot or zero-shot tasks (Liu et al., 2021a). It refers to prepending language instructions to the input text so that those language models can better *understand* the new task and give more customized *replies*. However, such a technique cannot be straightforwardly applied to GTs due to the significant disparity between the two types of data: graphs and texts. Given the symbolic graph data, it is infeasible and counter-intuitive to manually design semantic prompts like human languages for each target FSNC task. Inspired by more recent works (Lester et al., 2021; Jia et al., 2022), instead of manually devising prompts in the raw graph data space (e.g., nodes and edges), we propose to inject different continuous vectors as task-specific virtual nodes in the node embedding space to function as *soft prompts* to elicit the substantial knowledge contained in the learned GTs. In the fine-tuning phase, these prompts can be optimized via the few-shot labeled nodes from the support set in each FSNC task. Such a simple tuning with virtual node prompts can modulate the learned node embeddings according to arbitrary FSNC tasks. We have conducted extensive experiments to show the effectiveness of the proposed FS-NPT method, and the results further exhibit the great potential of prompt initialization and ensemble in terms of both accuracy and efficiency.

We hope our proposed FS-NPT can provide a new promising path forward. Our contributions are:

Problem Generalization. We relax the assumption in conventional *few-shot node classification* (FSNC) tasks to cover scenarios where there are no base classes with substantial labeled nodes.

Method Proposed. We propose a simple yet effective method, *Few-shot Node Prompt Tuning* (FS-NPT), that directly injects virtual nodes in the embedding space to function as prompts to customize the pretrained node embeddings for each FSNC task. Such prompts can be learned through back-propagation and do not require any human involvement. Since only a small prompt tensor and a simple classifier are retrained, and the single pretrained GT is recycled for all downstream FSNC tasks, our method considerably reduces the per-task storage and computation cost.

Comprehensive Experiment. We conduct extensive experiments on the four most widely used real-world datasets to show the effectiveness and applicability of our proposed method. We find that, without any assumption for base classes, FS-NPT can still outperform all the existing methods even if they are given labels of nodes in base classes. Further analysis also indicates that the proposed FS-NPT method can considerably benefit from prompt initialization and ensemble.

2 PROBLEM FORMULATION

In this work, we focus on few-shot node classification (FSNC) on a single graph. Formally, given an attributed network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{X}) = (\boldsymbol{A}, \boldsymbol{X})$, where $\mathcal{V}$ denotes the set of vertices $\{v_1, v_2, ..., v_V\}$, $\mathcal{E}$ denotes the set of edges $\{e_1, e_2, ..., e_M\}$, $\boldsymbol{X} = [\boldsymbol{x}_1; \boldsymbol{x}_2; ...; \boldsymbol{x}_V]$ denotes all the node features, and $\boldsymbol{A} = \{0, 1\}^{V \times V}$ is the adjacency matrix representing the network structure. Specifically, $\boldsymbol{A}_{j,k} = 1$ indicates that there is an edge between node v_j and node v_k; otherwise, $\boldsymbol{A}_{j,k} = 0$. The few-shot node classification problem assumes the existence of a series of node classification tasks, $\mathcal{T} = \{\mathcal{T}_i\}_{i=1}^{I}$, where $\mathcal{T}_i$ denotes the given dataset of a task, I denotes the number of such tasks. Those tasks are formed from target *novel classes* (i.e. $\mathbb{C}_{novel}$), where only a few labeled nodes are available per class. Traditional FSNC tasks assume there exists a disjoint set of *base classes* (i.e. $\mathbb{C}_{base}$, $\mathbb{C}_{base} \cap \mathbb{C}_{novel} = \varnothing$) on the graph where substantial labeled nodes are accessible during training. Then, the traditional few-shot node classification problem can be defined as follows:

Definition 1 ***Traditional Few-shot Node Classification:*** *Given an attributed graph $\mathcal{G} = (\boldsymbol{A}, \boldsymbol{X})$ with a divided node label space $\mathbb{C} = \{\mathbb{C}_{base}, \mathbb{C}_{novel}\}$, substantial labeled nodes for $\mathbb{C}_{base}$ are available, and the number of base classes, i.e. $|\mathbb{C}_{base}|$, is sufficiently large for sampling meta-tasks. On the contrary, only few-shot labeled nodes (support set $\mathbb{S}$) are available for the target novel classes, $\mathbb{C}_{novel}$. The task is to predict the labels for unlabeled nodes (query set $\mathbb{Q}$) from $\mathbb{C}_{novel}$. If the support set in each target (test) task has N novel classes with K labeled nodes, then we term this task a traditional N-way K-shot node classification task.*

However, the assumption of the existence of a disjoint set of base classes with substantial labeled nodes could be untenable for real-world graphs. For example, all the classes on a given graph may only have a few labeled nodes, and node classification is performed over all classes, or the number of base classes is very small. Considering this limitation, in this paper, we generalize the definition of FSNC such that it has no assumption for base classes. It is formulated as follows:

Definition 2 ***General Few-shot Node Classification:*** *Given an attributed graph $\mathcal{G} = (\boldsymbol{A}, \boldsymbol{X})$ with a target node label space $\mathbb{C}$, only few-shot labeled nodes (support set $\mathbb{S}$) are available for each class $c \in \mathbb{C}$. The task is to predict the labels for unlabeled nodes (query set $\mathbb{Q}$) from $\mathbb{C}$. If the support set in each target (test) task has N novel classes with K labeled nodes, then we term this task a general N-way K-shot node classification task.*

Our paper is the first to work on this more realistic yet challenging problem formulation of FSNC tasks. Without the assumption that there exists abundant labeled nodes from the base classes, the meta-learning based existing methods cannot be directly deployed. However, to explicitly demonstrate the advantages of our model, in Section 4, we compare our model trained under the general FSNC setting and the existing methods trained under the traditional setting, and find that the proposed method still can significantly outperform all the baselines even if our method is trained with much less labeled data.

3 METHODOLOGY

3.1 PRELIMINARY: GRAPH TRANSFORMERS

Graph transformers (GTs) (Zhang et al., 2020; Rong et al., 2020; Chen et al., 2022) are transformer (Vaswani et al., 2017)-based Graph Neural Networks (GNNs) without relying on convolution or aggregation operations. Following BERT (Devlin et al., 2018) for large-scale natural language modeling, a D-layer GT is used to project the node attribute $\boldsymbol{x}_j$ of each node v_j ($\forall j \in \mathbb{N}, 1 \leq j \leq V$) into the embeddings $\boldsymbol{e}_j$. GTs usually have a much larger number of parameters than traditional

GNNs and are often trained in a self-supervised manner, without the need for substantial gold-labeled nodes. For the sake of generality, we choose two simplest and most universally-used pretext tasks, *node attribute reconstruction* and *structure recovery*, to pretrain the GT encoder (Zhang et al., 2020; Chen et al., 2022). An exhaustive discussion of methods for pretraining GTs is out of the scope of this paper, please see more details for GT pretraining in Appendix B. Then, with a pre-trained GT, each node v_j, or say the graph $\mathcal{G}$, is projected into a F-dimensional embedding space. Both node attribute and topology (or position) structure are considered:

$$\boldsymbol{E}^0 = [\boldsymbol{e}_1^0; ...; \boldsymbol{e}_j^0; ...; \boldsymbol{e}_V^0] = Embed(\mathcal{G}) = Embed(\boldsymbol{X}, \boldsymbol{A}) \in \mathbb{R}^{V \times F}, \tag{1}$$

where n is the number of nodes in the given graph $\mathcal{G}$ and V is the embedding size. Then, the node embeddings $\boldsymbol{E}^{d-1}$ computed by the $d-1$-th layer are fed into the following transformer layer L^d ($\forall d \in \mathbb{N}, 1 \leq d \leq D$) to get more high-level node representations, which can be formulated as:

$$\boldsymbol{E}^d = [\boldsymbol{e}_1^d; ...; \boldsymbol{e}_j^d; ...; \boldsymbol{e}_V^d] = L^d(\boldsymbol{E}^{d-1}) \in \mathbb{R}^{V \times F}. \tag{2}$$

Conventionally, to adapt the pretrained GT to different downstream tasks, further fine-tuning of the GT on the corresponding datasets (Zhang et al., 2020; Rong et al., 2020; Chen et al., 2022) is performed. However, according to our experiments in Section 4, this vanilla approach suffers from the following limitations when applied to FSNC tasks: **(1)** The number of labeled nodes for each FSNC task is very limited (usually less than 5), making the fine-tuned GT highly overfit on them and hard to generalize to query set. **(2)** This method neglects the inherent gap between the training objective of the pretext tasks and that of the downstream FSNC tasks, rendering the transferred node embeddings sub-optimal for the target FSNC tasks. **(3)** For every new task, all the parameters of GT models need to be updated, making the model hard to converge and greatly raising the cost to apply GTs to real-world applications. This work is the first to propose a simple yet effective and efficient *prompting* method for GTs to tackle the three aforementioned limitations.

3.2 Few-shot Node Prompt Tuning

Since the GT encoder is pretrained on the given graph in a self-supervised manner, it does not require any label information from base classes. Then, in this section, we introduce the proposed *Few-shot Node Prompt Tuning* (FS-NPT) method which effectively utilizes the limited few labeled nodes in the *support set* $\mathbb{S}$ from the target label space $\mathbb{C}$ to customize the node representations from the pretrained GT for each specific FSNC task.

We introduce an extra set of p randomly initialized continuous parameters with the same embedding size F, *i.e. prompt*, denoted as $\boldsymbol{P} = [\boldsymbol{p}_1; ...; \boldsymbol{p}_p; ...; \boldsymbol{p}_P], (\boldsymbol{p}_p \in \mathbb{R}^F)$. Our prompt tuning strategy is simple to implement. We fix the pretrained weights of the GT encoder during fine-tuning while keeping the prompt parameter $\boldsymbol{P}$ trainable, and we concatenate this prompt with the pretrained node embedding right after the embedding layer and feed it to the first transformer layer of the GT. The injected prompts can be viewed as task-specific *virtual nodes* that help modulate the pretrained node representations and elicit the learned substantial knowledge from the pretrained GT for different target FSNC tasks. In such a manner, our approach allows the frozen large transformer layers to update the intermediate-layer node representations for different tasks, as contextualized by those virtual nodes (more detailed discussion about the effect from those virtual nodes on target FSNC tasks is presented in Section 4.3 and 4.5):

$$[\boldsymbol{E}^1 || \boldsymbol{Z}^1] = L^1([\boldsymbol{E}^0 || \boldsymbol{P}]) \in \mathbb{R}^{(V+P) \times F}, \tag{3}$$

where $||$ denotes the concatenation operator. Then we feed the learned node representations and the prompt to the following transformer layers L^d ($\forall d \in \mathbb{N}, 2 \leq d \leq D$), which is formulated as:

$$[\boldsymbol{E}^d || \boldsymbol{Z}^d] = L^d([\boldsymbol{E}^{d-1} || \boldsymbol{Z}^1]) \in \mathbb{R}^{(V+P) \times F}. \tag{4}$$

With the modulated node representations, we can get the predicted label for any node v_j by applying a simple classifier, f_ψ (e.g. SVM, Logistic Regression, shallow MLP, etc.):

$$y = f_\psi(\boldsymbol{e}_j^d). \tag{5}$$

Then, for each target N-way K-shot FSNC task $\mathcal{T}_i = \{\mathbb{S}_i, \mathbb{Q}_i\}$, we can predict labels for all the few labeled nodes in the support set $\mathbb{S}_i$, and calculate the Cross-entropy loss, $\mathcal{L}_{CE}$, to update the prompt parameters $\boldsymbol{P}$ and the simple classifier f_ψ. This optimization procedure can be formulated as:

$$\boldsymbol{P}, \psi = \arg\min_{\boldsymbol{P}, \psi} \mathcal{L}_{CE}(\mathbb{S}_i; \boldsymbol{P}, \psi). \tag{6}$$

Finally, following the same procedure, we use the fine-tuned prompt $\boldsymbol{P}$, classifier f_ψ, and node representations from the pretrained GT to predict labels for unlabeled nodes in the query set $\mathbb{Q}_i$. It is notable that the parameters of the pretrained GT are frozen throughout the node prompt tuning process and are recycled for all downstream FSNC tasks. In other words, to adapt to a new FSNC task, we only need to train a small prompt tensor $\boldsymbol{P}$ to modulate the intermediate-layer node representations to be customized by the few labeled nodes, which is computationally similar to training a very shallow MLP. This signifies that the proposed FS-NPT method requires a low per-task storage and computation cost to retain its effectiveness.

4 Experimental Study

4.1 Experimental Settings

We conduct systematic experiments to compare the proposed FS-NPT method with the baselines on the few-shot node classification task. In this work, we consider two categories of baselines, i.e., *meta-learning* based methods and *graph contrastive learning* (GCL) based self-supervised learning methods (Tan et al., 2022). For meta-learning, we test typical methods including: **Meta-GNN** (Zhou et al., 2019), **G-Meta** (Huang & Zitnik, 2020), **GPN** (Ding et al., 2020), **AMM-GNN** (Wang et al., 2020b), and **TENT** (Wang et al., 2022). For GCL based self-supervised learning, the chosen pretraining methods contain: **MVGRL** (Hassani & Khasahmadi, 2020), **GraphCL** (You et al., 2020), **GRACE** (Zhu et al., 2020), and **BGRL** (Thakoor et al., 2021). For these GCL based methods and the proposed FS-NPT, we choose Logistic Regression as the classifier f_ψ. We describe the details of these models in Appendix A. For comprehensive studies, we report the results of those methods on four prevalent real-world graph datasets: `CoraFull` (Bojchevski & Günnemann, 2018), `ogbn-arxiv` (Hu et al., 2020a), `Cora` (Yang et al., 2016), `CiteSeer` (Yang et al., 2016). Specifically, each dataset is a graph that contains a considerable number of node classes. This ensures that the evaluation consists of various tasks for a more comprehensive evaluation. A more detailed description of those datasets is provided in Appendix D, with their statistics and class splits in Table 6 in Appendix C. It is notable that our method does not have any assumption for base classes, but this assumption is necessary for all the baselines. For explicit comparison, we still provide base classes for all the baselines (**our method does not use any label information from base classes**), and we compare our method with them under various N-way K-shot settings.

4.2 Comparable Study

Table 1: The overall comparison between the proposed FS-NPT method and meta-learning or GCL pretraining based methods under different settings. Accuracy (↑) and confident interval (↓) are in %. The best results are **bold**, and the second best results in each category of methods are underlined. OOM denotes the out-of-memory issue. More results are included in Table 8 in Appendix H.

Dataset	`CoraFull`		`Ogbn-arxiv`		`CiteSeer`		`Cora`	
Setting	5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot	2-way 1-shot	2-way 5-shot	2-way 1-shot	2-way 5-shot
Meta-GNN	55.33±2.43	70.50±2.02	27.14±1.94	31.52±1.71	56.14±2.62	67.34±2.10	65.27±2.93	72.51±1.91
GPN	52.75±2.32	72.82±1.88	37.81±2.34	50.50±2.13	53.10±2.39	63.09±2.50	62.61±2.71	67.39±2.33
AMM-GNN	58.77±2.32	75.61±1.78	33.92±1.80	48.94±1.87	54.53±2.51	62.93±2.42	65.23±2.67	82.30±2.07
G-Meta	60.44±2.48	75.84±1.70	31.48±1.70	47.16±1.73	55.15±2.68	64.53±2.35	67.03±3.22	80.05±1.98
TENT	55.44±2.08	70.10±1.73	48.26±1.73	61.38±1.72	62.75±3.23	72.95±2.13	53.05±2.78	62.15±2.13
MVGRL	59.91±2.39	76.76±1.63	OOM	OOM	64.45±2.77	80.25±1.82	71.17±3.04	89.91±1.44
GraphCL	64.20±2.56	83.74±1.46	OOM	OOM	**73.51±3.09**	**92.38±1.24**	73.50±3.18	**92.35±1.30**
GRACE	66.69±2.26	84.06±1.43	OOM	OOM	69.85±2.75	85.93±1.57	69.13±2.69	88.68±1.37
BGRL	43.83±2.11	70.44±1.62	36.76±1.74	53.44±0.36	54.32±1.63	70.50±2.11	60.14±2.33	79.86±1.92
FS-NPT (**Ours.**)	**68.50±2.13**	**84.56±2.15**	**50.40±1.97**	**74.91±1.87**	70.60±2.15	86.23±1.75	**84.50±1.94**	90.50±1.55

Table 1 presents the performance comparison of all the methods on the few-shot node classification task. Specifically, we present results under four different few-shot settings to exhibit a more comprehensive comparison: 5-way 1-shot, 5-way 5-shot, 2-way 1-shot, and 2-way 5-shot. We choose the average classification accuracy and the 95% confidence interval over 5 repetitions with different random seeds as the evaluation metrics. For each repetition, we sample 100 meta-test tasks for evaluation and calculate the evaluation metrics. From Table 1, we obtain the following observations:

- Even **without any label information from base classes**, the proposed method, FS-NPT, can consistently outperform meta-learning based methods and outperform GCL based self-supervised learning methods in most cases. This demonstrates the considerable superiority of the proposed FS-NPT in terms of accuracy. The pretrained GT has learned substantial prior knowledge and the injected virtual node prompts effectively elicit the knowledge for different downstream FSNC tasks.
- Generally speaking, for both GCL and the proposed GT, **self-supervised pretraining can outperform meta-learning based method**. However, one most recent pretraining method, BGRL, when transferred for downstream FSNC tasks, shows surprisingly frustrating performance. This further validates **the impact from the gap of training objective** between pretexts and target FSNC tasks. The pretext of BGRL minimizes the *Mean Square Error* of the original node representation and its slightly perturbed counterpart but does not enforce the model to discriminate between different nodes as the other GCL baselines do. The objective of this pretext deviates more from the downstream FSNC tasks, thus leading to worse results. We further show the impact of this in ablation studies (see Section 4.3).
- Compared to all the baselines, the proposed FS-NPT method is **more robust to extremely scarce label scenarios**, i.e, the number of labeled nodes in the support set K equals 1. **The performance degradation resulting from decreasing the number of shots** K is significant for all the methods. More detailed results can be found in Appendix F.1. Smaller K makes the encoder or the classifier more prone to overfitting, thus leading to worse generalization to query sets. In contrast, the proposed method injects virtual nodes into the model, which have separate learnable embeddings for different FSNC tasks. This implies that our method implicitly performs adaptable data augmentation for the few labeled nodes, which makes our framework more robust to tasks with extremely scarce labeled nodes. Further analysis and explanation are given in Section 4.5.

4.3 Ablation Study

Table 2: Ablation study on `Cora` and `Ogbn-arxiv` datasets to analyze the effectiveness of different components in our method.

Encoder	Frozen	Prompt	Cora		Ogbn-arxiv			
			2-way 1-shot	2-way 5-shot	2-way 1-shot	2-way 5-shot	5-way 1-shot	5-way 5-shot
GCN			52.12±2.62	57.93±2.23	57.62±2.31	64.11±2.65	26.68±1.57	27.90±1.45
GCN	✓		68.43±2.94	78.20±2.83	65.21±2.86	77.10±2.46	38.47±1.77	51.46±1.69
GT			67.50±2.24	79.42±1.89	63.00±2.35	79.84±1.98	40.73±2.65	55.35±1.88
GT	✓		75.50±2.54	84.94±1.74	53.64±2.62	73.64±2.33	31.64±2.45	52.36±2.04
GT		✓	77.85±1.99	85.43±1.84	71.82±2.58	82.73±2.14	36.36±2.74	65.45±2.31
GT	✓	✓	**84.50±1.94**	**90.50±1.55**	**82.00±1.77**	**87.27±1.64**	**50.40±1.97**	**74.91±1.87**

In this subsection, we conduct ablation studies to investigate the effectiveness of different components in our framework. We present the results of experiments on the `Cora` and `Ogbn-arxiv` datasets, under different N-way K-shot settings (similar results can be observed on the other datasets and settings). For the GCN baseline, following the common practice (Zhou et al., 2019; Ding et al., 2020), we pretrain a 2-layer GCN using all the data from base classes with Cross-Entropy Loss. Specifically, Frozen means during fine-tuning, the GNN encoder is fixed, and only the classifier is fine-tuned. Prompt refers to the proposed node prompt tuning method. The results are shown in Table 3, from which we draw the following conclusions:

- Our simple implementation of GT can **consistently achieve much better results** than traditional GNNs such as GCN. This is because the GT has a much larger number of parameters, making it capable of learning more complex relations among nodes. Besides, pretrained with the two pretext tasks, i.e., *node attribute reconstruction* and *structure recovery*, in a self-supervised manner, GT can learn much more transferable graph patterns compared to those meta-learning based methods.
- **Freezing the GNN encoder** during fine-tuning on the downstream FSNC tasks consistently leads to better results. This shows that fine-tuning the graph encoder on the few labeled nodes would in turn impair the quality of the learned node embedding.

- The proposed method, FS-NPT, which contains a frozen pretrained GT encoder with prompt tuning can provide **the best performance**. This implies that the introduced virtual node prompt can help the model better modulate the learned substantial graph knowledge for each FSNC task while avoiding impairing the pretrained node embeddings.

4.4 Classification over All Classes on Graph Datasets

As aforementioned in this paper, the proposed FS-NPT method does not require any assumption on the availability of a separate base class set disjoint with the target few-shot classes. It means that, given few-shot labeled nodes from each class, FS-NPT can perform node classification over all the classes in the graph. We conduct this experiment on `Cora` and `CiteSeer` datasets and list the results in Table 3. From the table, we can see that the original GT cannot perform well due to the shortage of supervision. But the proposed FS-GPT can improve the performance significantly, especially for larger shot numbers, K.

Table 3: The FSNC results of the vanilla GT and the proposed FS-NPT methods over all the node classes on `Cora` and `CiteSeer` datasets.

Dataset	`Cora`		`CiteSeer`	
Setting	7-way 1-shot	7-way 5-shot	6-way 1-shot	6-way 5-shot
GT	34.55	61.30	21.94	43.64
FS-NPT	**50.13**	**74.55**	**29.36**	**63.51**

4.5 Interpretation of Virtual Nodes as Prompt

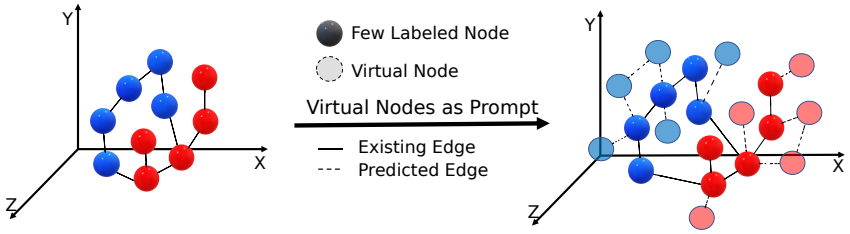


Figure 1: The illustration of effect from introduced virtual nodes under 2-way node classification setting. Different colors indicate different node classes.

Different from the pioneer works in NLP (Brown et al., 2020; Liu et al., 2021a), where prompts consist of human languages that are naturally interpretable to humans. For graph data, we inject the virtual nodes in the node embedding space rather than the raw graph space (e.g., nodes and edges). Hence, it is hard to interpret the effect of the proposed node prompts. To better understand the behavior of the virtual node prompts, we leverage Walkpooling (Pan et al., 2021), which is the state-of-the-art for the link prediction task, to predict the links that are the most likely to exist between the virtual nodes and the few labeled nodes from the support set of each task. We train the model on the whole graph dataset and use it to predict the most possible links between the virtual nodes and the existing ones. Specifically, we only consider potential links with at least one virtual node as their vertex. Under the 2-way few-shot node classification setting, we initialize half of the virtual node prompts as the prototype vector of the first class, and the other half of the virtual node as the prototype vector of the second class. As indicated in Fig. 1, after the convergence of prompt tuning, we notice that the vertices of the most possible links are always connecting existing nodes with virtual prompt nodes from the same classes. This implies that the introduced virtual node prompts can learn node representations semantically similar to those from the same class, **thus helping push node representations from the same classes closer**. To further validate this, on `Cora` dataset, we calculate the average *cosine* similarities and L_2 distances (normalized by the longest distance of any pair of nodes) for virtual nodes and existing nodes from the two novel classes. As presented in Table 4, we can see that the virtual nodes and existing nodes from the same classes have smaller L_2 distances and larger *cosine* similarities. We give further analyses of the effect from different numbers of such virtual nodes in Fig. 5 in Appendix F.2.

Table 4: The L_2 distance and *cosine* similarity scores between prompting nodes and real nodes from two novel classes on `Cora` dataset. P denotes virtual node prompts, N denotes existing nodes.

Metrics	L_2	*Cosine*	L_2	*Cosine*
Nodes	P in class-1		P in class-2	
N in class-1	0.0622	0.7625	0.6725	0.2344
N in class-2	0.6520	0.8627	0.0548	0.1165

4.6 NODE REPRESENTATIONS CLUSTERING ON NOVEL CLASSES

In this section, we analyze the quality of the learned node representations from different training strategies. Particularly, we leverage two prevalent clustering evaluation metrics: *Normalized Mutual Information* (NMI) and *adjusted random index* (ARI), on learned node embeddings clustered based on K-Means. Also, we deploy t-SNE to visualize them and compare them with those learned by baseline methods on the `CoraFull` dataset. We choose nodes from 5 randomly selected novel classes for visualization. The results are presented in Table 7 and Fig. 3 in Appendix E. We observe that the proposed FS-NPT method enhances the quality of the node representations of GTs, and achieves the best clustering performance on novel classes. Also, based on the visualization, we discover that a vanilla GT without prompt tuning cannot learn node embeddings that are discriminative enough compared to strong baselines like TENT and GRACE. However, when equipped with the proposed FS-NPT, a GT can learn highly discriminative node embeddings. This also authenticates that the added prompts can help elicit more customized knowledge for each downstream FSNC task.

4.7 PROMPT INITIALIZATION

The results in Section 4.5 imply that carefully initializing the virtual node prompts as prototype representations could be beneficial to the virtual node prompt tuning process. Intuitively, given a test node, an ideal model should produce an output node embedding that is close to the corresponding class prototype representations. Initializing the prompts with prototype embeddings that enumerate the node embedding space might give the model some hints about the target categories, thus helping improve the optimization process. Specifically, we initialize equal portions of virtual prompt nodes to all novel node classes as their prototype representations. The corresponding results are given in Table 5. We can observe that such a simple initialization can consistently enhance the performance on target downstream FSNC tasks. Note that even though promising improvement is observed by this prototype-based prompt initialization strategy, we still choose random initialization as the default option for the other experiments for its simplicity and generality.

Table 5: The accuracy scores of FS-NPT on `Cora` and `Ogbn-arxiv` datasets. Init. indicates the prototype-based prompt initialization strategy described in Section 4.7. Method without Init. means the prompts are randomly initialized. The best results are **bold**. MV refers to majority voting.

FS-GPT			`Cora`		`Ogbn-arxiv`			
Init.	**Ensemble**		2-way 1-shot	2-way 5-shot	2-way 1-shot	2-way 5-shot	5-way 1-shot	5-way 5-shot
			84.50	90.50	82.00	87.27	50.40	74.91
✓			85.25	91.30	83.05	88.34	51.06	75.86
✓	✓	**Avg.**	85.50	89.64	82.50	89.65	48.82	75.65
		Best	86.24	92.38	**85.45**	90.63	**53.68**	78.82
		MV	**87.63**	**92.52**	84.72	**91.50**	52.15	**79.60**

4.8 PROMPT ENSEMBLE

Lester et al. (2021) has demonstrated the efficiency of prompt for model ensembling. Since the large transformer backbone is frozen after pretraining and can always be reused, only the small prompt tensors require extra space to store. Furthermore, during inference, only one pass is needed with a specially-designed batch with replicated original data and variously initialized prompts (Lester et al., 2021; Jia et al., 2022). Given such advantages, we investigate the effectiveness of enabling prompt ensembling for FS-NPT. Concretely, to facilitate the ensembling with the prompt initialization strategy, we add independently sampled Gaussian noise tensors to 5 prompts for each FSNC task. Each prompt contains virtual nodes initialized as node class prototypes. We use simple majority voting to compute final predictions from the ensemble. Table 5 shows that the ensembled model can outperform the average or even the best single prompt counterpart.

4.9 EFFECTIVENESS ACCORDING TO THE SCALE OF MODEL

In Fig. 2, we present the accuracy of the proposed framework against the scale of the GT encoder as a heat map. We consider the width (embedding size F) and the depth (the number of transformer layers D) of the GT. The results shown are from the `Cora` dataset under the 2-way 1-shot setting and we observe similar trends on other datasets under different settings. We have the following findings:

- **As the depth increases, the final accuracy decreases.** This is because we only inject the virtual node prompt before the first transformer layer. The effect vanishes as the prompt goes deeper. On the other hand, it shows that naively increasing the depth of the GT encoder is not necessarily helpful for the downstream FSNC tasks, and the proposed FS-NPT method is effective and necessary for better adaptation.
- **As the width increases, the final accuracy increases.** Larger embedding size implies the input node embeddings contain richer semantic and topological knowledge, and the injected prompt can learn to modulate the node embeddings more precisely.

5 Background and Related Work

Few-shot Node Classification. Recently, episodic meta-learning (Finn et al., 2017) has become the most dominant paradigm for FSNC tasks. It trains the GNN encoders by explicitly emulating the test environment for few-shot learning (Zhou et al., 2019; Ding et al., 2020). However, those methods rely on the assumption that there exist *base classes* with substantial labeled nodes per class to sample episodes. So the existing methods cannot work for the general FSNC problem defined in this paper. A more detailed review is given in Appendix I.

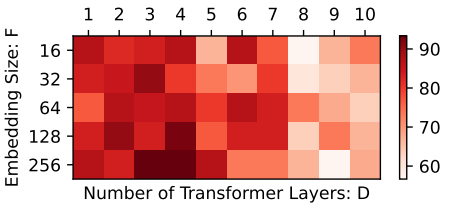


Figure 2: The 2-way 1-shot accuracy (%) of the proposed FS-NPT according to the scale of the GT encode on the `Cora` dataset.

Graph Transformer. Graph Transformers (GTs) (Zhang et al., 2020; Rong et al., 2020; Chen et al., 2022) are a new category of GNNs that are based on transformers (Vaswani et al., 2017). Usually, GTs contain two parts: A embedding network that projects the raw graphs to the embedding space. Then, a transformer-based network is involved to learn the complicated relationships among embeddings. GTs have a larger magnitude of parameter numbers to contain richer knowledge. Typically, GTs are trained in a self-supervised manner with pre-designed pretexts, such as node attribute reconstruction (Zhang et al., 2020) and structure recovery (Chen et al., 2022). GTs have shown promising performance for general transfer learning, but there is no existing work succeeding in adapting the huge GTs to few-shot scenarios where the labels are extremely scarce.

Learning with Prompts. In NLP, prompting (Liu et al., 2021a; Lester et al., 2021) has become a new fashion recently for adapting huge language models to different downstream tasks by prepending task descriptions to the input texts. However, this requires human involvement to design descriptions in human words, making it costly and hard to deploy on symbolic graphs. More recently, some works (Lester et al., 2021; Jia et al., 2022) find that by introducing external learnable parameters as soft prompts, those language models can fast adapt to new tasks without human labor. However, no work has been done for symbolic graph data due to the significant uniqueness of graphs that the instances on graphs, nodes, are not i.i.d. Our work is the first to provide a prompt-based method for learning representative node embeddings on graphs, especially, under the extreme few-shot setting.

6 Conclusion

In this paper, we first extend the traditional FSNC problem setting to a more realistic yet challenging formulation, where the assumption for the existence of base classes may not necessarily hold. To tackle this problem, we propose FS-NPT as an efficient solution to modulate pretrained GTs according to different FSNC tasks. Concretely, a GT encoder is pretrained on the whole graph in a self-supervised manner by two widely-used pretext tasks. Then, virtual nodes are involved in the embedding space to customize the learned node embeddings from GT for different FSNC tasks. We conduct rigorous empirical studies to validate the effectiveness of the proposed FS-NPT method on four widely-used real-world graph datasets. The results also demonstrate the great potential for prompt initialization and ensemble.

In broader terms, this work is the first to investigate the general FSNC tasks and explore an effective method to introduce prompt tuning for symbolic graphs under such an extreme shortage of label scenarios. We hope our work can provide a novel path forwarding the learning with scarce supervision on graphs, and an alternative way to fine-tune large GTs for the target domain.

REFERENCES

Aleksandar Bojchevski and Stephan Günnemann. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. In *ICLR*, 2018.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Dexiong Chen, Leslie O'Bray, and Karsten Borgwardt. Structure-aware transformer for graph representation learning. In *International Conference on Machine Learning*, pp. 3469–3489. PMLR, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Kaize Ding, Jianling Wang, Jundong Li, Kai Shu, Chenghao Liu, and Huan Liu. Graph prototypical networks for few-shot learning on attributed networks. In *CIKM*, 2020.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Pires, Zhaohan Guo, Mohammad Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *NeurIPS*, 2020.

William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive Representation Learning on Large Graphs. In *NeurIPS*, pp. 1024–1034, 2017.

Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *International Conference on Machine Learning*, pp. 4116–4126. PMLR, 2020.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *NeurIPS*, 2020a.

Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of The Web Conference 2020*, pp. 2704–2710, 2020b.

Kexin Huang and Marinka Zitnik. Graph meta learning via local subgraphs. In *NeurIPS*, 2020.

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. *arXiv preprint arXiv:2203.12119*, 2022.

Yizhu Jiao, Yun Xiong, Jiawei Zhang, Yao Zhang, Tianqi Zhang, and Yangyong Zhu. Sub-graph contrast for scalable self-supervised graph representation learning. In *2020 IEEE international conference on data mining (ICDM)*, pp. 222–231. IEEE, 2020.

Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*, 2017.

Lin Lan, Pinghui Wang, Xuefeng Du, Kaikai Song, Jing Tao, and Xiaohong Guan. Node classification on graphs with few-shot novel labels via meta transformed network embedding. *Advances in Neural Information Processing Systems*, 33:16520–16531, 2020.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021a.

Zemin Liu, Yuan Fang, Chenghao Liu, and Steven CH Hoi. Relative and absolute location embedding for few-shot node classification on graph. In *AAAI*, 2021b.

Yujie Mo, Liang Peng, Jie Xu, Xiaoshuang Shi, and Xiaofeng Zhu. Simple unsupervised graph representation learning. AAAI, 2022.

Liming Pan, Cheng Shi, and Ivan Dokmanić. Neural link prediction with walk pooling. In *International Conference on Learning Representations*, 2021.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *Proceedings of the 31st Conference on Neural Information Processing Systems*, 2017.

Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. Self-supervised graph transformer on large-scale molecular data. *Advances in Neural Information Processing Systems*, 33:12559–12571, 2020.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017.

Zhen Tan, Kaize Ding, Ruocheng Guo, and Huan Liu. A simple yet effective pretraining strategy for graph few-shot learning. *arXiv preprint arXiv:2203.15936*, 2022.

Shantanu Thakoor, Corentin Tallec, Mohammad Gheshlaghi Azar, Remi Munos, Petar Veličković, and Michal Valko. Bootstrapped representation learning on graphs. In *ICLR Workshop on Geometrical and Topological Representation Learning*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. In *ICLR*, 2018.

Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia. Microsoft academic graph: When experts are not enough. *Quantitative Science Studies*, 2020a.

Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*, 2019.

Ning Wang, Minnan Luo, Kaize Ding, Lingling Zhang, Jundong Li, and Qinghua Zheng. Graph few-shot learning with attribute matching. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, 2020b.

Song Wang, Kaize Ding, Chuxu Zhang, Chen Chen, and Jundong Li. Task-adaptive few-shot node classification. *arXiv preprint arXiv:2206.11972*, 2022.

Zhihao Wen, Yuan Fang, and Zemin Liu. Meta-inductive node classification across graphs. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *Proceedings of the 2019 International Conference on Learning Representations*, 2019.

Zhilin Yang, William Cohen, and Ruslan Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pp. 40–48. PMLR, 2016.

Huaxiu Yao, Chuxu Zhang, Ying Wei, Meng Jiang, Suhang Wang, Junzhou Huang, Nitesh Chawla, and Zhenhui Li. Graph few-shot learning via knowledge transfer. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, 2020.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *NeurIPS*, 2020.

Jiawei Zhang, Haopeng Zhang, Congying Xia, and Li Sun. Graph-bert: Only attention is needed for learning graph representations. *arXiv preprint arXiv:2001.05140*, 2020.

Shengzhong Zhang, Ziang Zhou, Zengfeng Huang, and Zhongyu Wei. Few-shot classification on graphs with structural regularized gcns. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.

Fan Zhou, Chengtai Cao, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Ji Geng. Meta-gnn: On few-shot node classification in graph meta-learning. In *CIKM*, 2019.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131*, 2020.

A Baseline Description

In this section, we provide further details about the baselines used in our experiments.

Meta-learning based methods:

- **ProtoNet** (Snell et al., 2017): ProtoNet learns a prototype for each class in meta-tasks by averaging the embeddings of samples in this class. Then it conducts classification on query instances based on their distances to prototypes.
- **MAML** (Finn et al., 2017): MAML first optimizes model parameters according to the gradients calculated on the support instances for several steps. Then it meta-updates parameters based on the loss of query instances calculated with the parameters updated on support instances.
- **Meta-GNN** (Zhou et al., 2019): Meta-GNN combines GNNs with the MAML strategy to apply meta-learning to graph-structured data. Specifically, Meta-GNN learns node embeddings with GNNs, while updating and meta-updating the GNN parameters based on the MAML strategy.
- **G-Meta** (Huang & Zitnik, 2020): G-Meta extracts a subgraph for each node to learn the node representation with GNNs. Then it conducts the classification on query nodes based on the MAML strategy to update and meta-update the parameters of GNNs.
- **GPN** (Ding et al., 2020): GPN proposes to learn node importance for each node in meta-tasks to select more beneficial nodes for classification. Then GPN utilizes PN to learn node prototypes via averaging node embeddings in a weighted manner.
- **AMM-GNN** (Wang et al., 2020b): AMM-GNN proposes to extend MAML with an attribute matching mechanism. Specifically, the node embeddings will be adjusted according to the embeddings of nodes in the entire meta-task in an adaptive manner.
- **TENT** (Wang et al., 2022): TENT reduces the variance among different meta-tasks for better generalization performance. In particular, TENT learns node and class representations by conducting node-level and class-level adaptations. It also incorporates task-level adaptations that maximize the mutual information between the support set and the query set.

Self-supervised GCL Pretraining methods:

- **MVGRL** (Hassani & Khasahmadi, 2020): MVGRL learns node and graph-level representations by contrasting the representations of two structural views of graphs, which include first-order neighbors and a graph diffusion. It utilizes a Jensen-Shannon Divergence based contrastive loss L_{JSD}.
- **GraphCL** (You et al., 2020): GraphCL proposes to leverage combinations of different transformations in GCL to facilitate GNNs with generalizability, transferability, and robustness without sophisticated architectures. It also uses L_{JSD} as the objective.
- **GRACE** (Zhu et al., 2020): GRACE proposes a hybrid scheme for generating different graph views on both structure and attribute levels. GRACE further provides theoretical justifications behind the motivation. It proposes a variant of Information Noise Contrastive Estimation $L_{InfoNCE}$ as the contrastive loss.
- **BGRL** (Thakoor et al., 2021): BGRL leverages the concept of BYOL (Grill et al., 2020) and applies it to graph-structured data by enforcing the agreement between positive views without any explicitly designs on negative views. Especially, it uses Mean Squared Error L_{MSE} between positive views as the final loss.

B Implementation Detail

B.1 General Settings

All experiments are implemented using PyTorch Paszke et al. (2017). We run all experiments on a single 80GB Nvidia A100 GPU.

B.2 Implementation of the Simplified GT

For the sake of generality, we try to keep the used GT encoder very simple and easy to transfer to other complicated architectures. Specifically, we use 1-layer MLP to project the raw graph, including node attributes and structural positions into the embedding space. We use simple summation to merge those embeddings together and feed them into the following transformer layers. We use the transformer module released by huggingface (Wolf et al., 2019). We perform a grid search like Section 4.9 to get the width and depth of the GT.

For pretraining, we utilize two prevailing pretext tasks, *node attribute reconstruction* and *structure recovery*, to train the GT encoder in a self-supervised manner. Concretely, for *node attribute reconstruction* pretext, given a node, we minimize the *Mean Square Error* (MSE) between the original node attributes and the reconstructed version via a fully connected layer and the learned node embedding from the GT. For *structure recovery* pretext, given any pair of nodes, we try to predict if there is a link between them and compare the result with the ground truth by an MSE loss. To accommodate a larger graph, similar to Jiao et al. (2020); Mo et al. (2022), we adopt the mini-batch strategy to sample a portion of nodes with their subgraphs (based on PPR) in each epoch for pretraining.

C Statistics of Benchmark Datasets

Table 6: Statistics of benchmark node classification datasets. $\mathbb{C}_{train}$ denotes the base classes for training, $\mathbb{C}_{dev}$ and $\mathbb{C}_{test}$ denote novel classes for validation and test respectively.

Dataset	# Nodes	# Edges	# Features	$\lvert\mathbb{C}\rvert$	$\lvert\mathbb{C}_{train}\rvert$	$\lvert\mathbb{C}_{dev}\rvert$	$\lvert\mathbb{C}_{test}\rvert$
`CoraFull`	19,793	63,421	8,710	70	40	15	15
`Ogbn-arxiv`	169,343	1,166,243	128	40	20	10	10
`Cora`	2,708	5,278	1,433	7	3	2	2
`CiteSeer`	3,327	4,552	3,703	6	2	2	2

D DESCRIPTION OF DATASETS

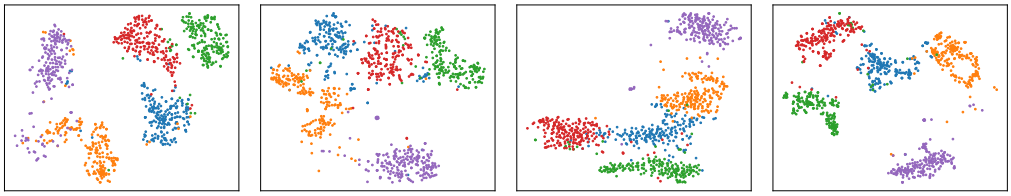
In this section, we provide detailed descriptions of the benchmark datasets used in our experiments. All the datasets are public and available on both PyTorch-Geometric (Fey & Lenssen, 2019) and DGL (Wang et al., 2019).

- **CoraFull** (Bojchevski & Günnemann, 2018) is a citation network that extends the prevalent small Cora network. Specifically, it is achieved from the entire citation network, where nodes are papers, and edges denote the citation relations. The classes of nodes are obtained based on the paper topic. For this dataset, we use 40/15/15 node classes for $\mathbb{C}_{train}$/$\mathbb{C}_{dev}$/$\mathbb{C}_{test}$.
- **Ogbn-arxiv** (Hu et al., 2020a) is a directed citation network that consists of CS papers from MAG (Wang et al., 2020a). Here nodes represent CS arXiv papers, and edges denote the citation relations. The classes of nodes are assigned based on the 40 subject areas of CS papers in arXiv. For this dataset, we use 20/10/10 node classes for $\mathbb{C}_{train}$/$\mathbb{C}_{dev}$/$\mathbb{C}_{test}$.
- **Cora** (Yang et al., 2016) is a citation network dataset where nodes mean paper and edges mean citation relationships. Each node has a predefined feature with 1433 dimensions. The dataset is designed for the node classification task. The task is to predict the category of a certain paper. For this dataset, we use 3/2/2 node classes for $\mathbb{C}_{train}$/$\mathbb{C}_{dev}$/$\mathbb{C}_{test}$.
- **CiteSeer** (Yang et al., 2016) is also a citation network dataset where nodes mean scientific publications and edges mean citation relationships. Each node has a predefined feature with 3703 dimensions. The dataset is designed for the node classification task. The task is to predict the category of a certain publication. For this dataset, we use 2/2/2 node classes for $\mathbb{C}_{train}$/$\mathbb{C}_{dev}$/$\mathbb{C}_{test}$.

E NODE REPRESENTATION CLUSTERING RESULTS

Table 7: The overall NMI (↑) and ARI (↑) scores of baselines and proposed FS-NPT on `CoraFull` and `CiteSeer` datasets. The best and second best results are **bold** and underlined, respectively.

Dataset	`CoraFull`		`CiteSeer`	
Metrics	**NMI**	**ARI**	**NMI**	**ARI**
AMM-GNN	0.6247	0.5087	0.2090	0.1781
G-Meta	0.5003	0.3702	0.1913	0.1502
Meta-GNN	0.5534	0.4196	0.1317	0.1171
GPN	0.6001	0.4599	0.2119	0.2087
TENT	0.5760	0.4652	0.0930	0.0811
MVGRL	0.6227	0.4788	0.2554	0.2232
GraphCL	<u>0.7023</u>	0.5628	<u>0.5579</u>	<u>0.5890</u>
GRACE	0.6781	<u>0.5856</u>	0.2663	0.2778
BGRL	0.5137	0.4382	0.2051	0.1875
GT	0.5225	0.3864	0.3452	0.3189
FS-NPT	**0.7768**	**0.6427**	**0.5998**	**0.6331**



(a) TENT (b) GRACE (c) GT (d) FS-NPT

Figure 3: The t-SNE visualization results on `CoraFull` (5-way) dataset.

F Parameter Study

In this section, we experiment on important sensitive parameters for the proposed FS-NPT method, including different N-way K-shot settings and different numbers of virtual nodes per prompt.

F.1 N-way K-shot settings

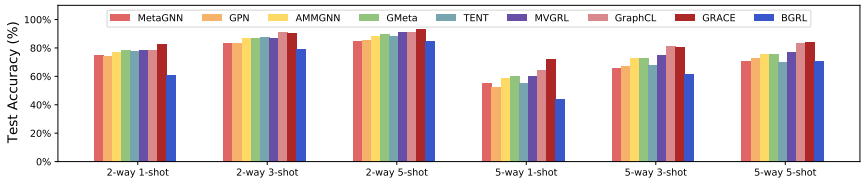


Figure 4: The accuracy (%) under different N-way K-shot settings on the `CoraFull` dataset.

F.2 Number of Virtual Nodes

The number of virtual nodes P is an important hyper-parameter to tune. On the four benchmark datasets, we give the results under the 2-way 5-shot setting. Specifically, we use a parameter α to control the number of virtual nodes. For a N-way K-shot FSNC task, we define $\alpha = \frac{P}{N \cdot K}$. Larger α means more virtual nodes are introduced. From the results shown in Fig. 5, we find that when $\alpha = 1$, the proposed FS-NPT can give the best performance. Therefore, we choose $P = \alpha \times (N \cdot K) = N \cdot K$ as the default number of virtual nodes per prompt.

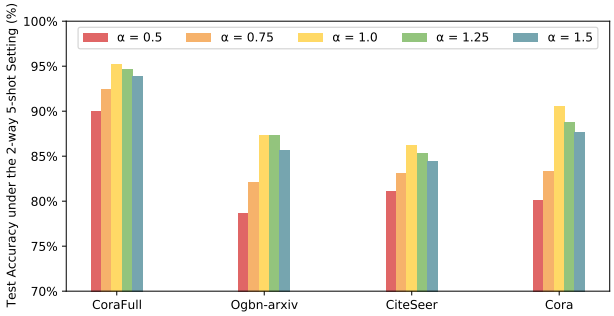


Figure 5: The test accuracy (%) under different α values on four benchmark datasets.

G Box-plot Illustration

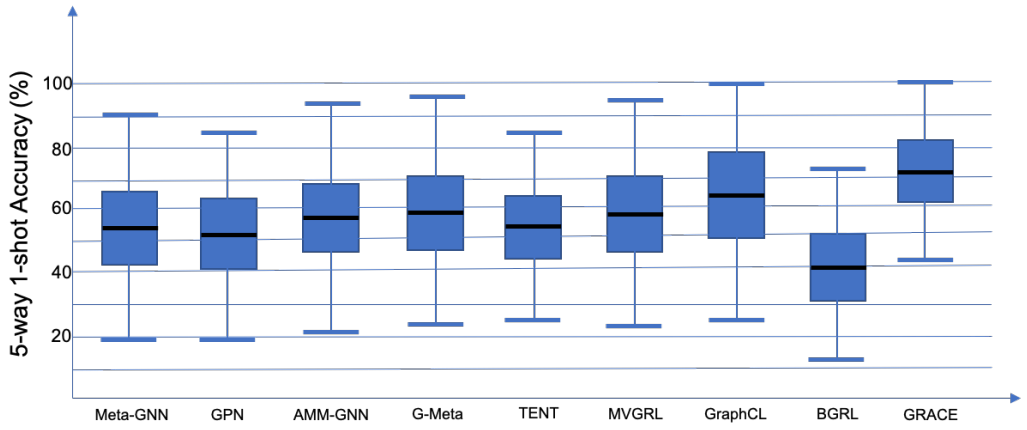


Figure 6: The box-plot illustrates the performance of state-of-the-art FSNC methods on the `CoraFull` dataset under the 5-way 1-shot setting

H More Results on CoraFull and Ogbn-arxiv datasets

Table 8: More comparison results between the proposed FS-NPT method and meta-learning or GCL pretraining based methods under different settings. Accuracy (↑) and confident interval (↓) are in %. The best results are **bold**, and the second best results in each category of methods are underlined. OOM denotes the out-of-memory issue.

Dataset	`CoraFull`		`Ogbn-arxiv`	
Setting	2-way 1-shot	2-way 5-shot	2-way 1-shot	2-way 5-shot
Meta-GNN	75.28±3.85	84.59±2.89	62.52±3.41	70.15±2.68
GPN	74.29±3.47	85.58±2.53	64.00±3.71	76.78±3.5
AMM-GNN	77.29±3.40	88.66±2.06	64.68±3.13	78.42±2.71
G-Meta	78.23±3.41	89.49±2.04	63.06±3.32	76.56±2.89
TENT	77.75±3.29	88.20±2.61	70.30±2.85	81.35±2.77
MVGRL	78.81±3.32	91.03±1.80	OOM	OOM
GraphCL	78.50±3.26	91.30±2.11	OOM	OOM
GRACE	80.71±3.01	91.68±1.57	OOM	OOM
BGRL	61.08±2.64	85.03±2.25	59.91±2.36	76.75±1.86
FS-NPT (**Ours.**)	**85.50±3.14**	**95.24±1.87**	**82.00±2.35**	**87.27±1.95**

I A more Detailed Review for Few-shot Node Classification

The task of few-shot node classification targets learning a model to assign labels for unlabeled nodes on graphs with only a few labeled nodes per class for training. Recently, episodic meta-learning (Finn et al., 2017) has become the most dominant paradigm to deal with such label shortages for FSNC tasks. It trains the GNN encoders by explicitly emulating the test environment for few-shot learning. For instance, Meta-GNN (Zhou et al., 2019) applies MAML (Finn et al., 2017) to learn directions for optimization with limited labels. GPN (Ding et al., 2020) adopts Prototypical Networks (Snell et al., 2017) to perform the classification based on the distance between the node feature and the prototypes. MetaTNE (Lan et al., 2020) and RALE (Liu et al., 2021b) also use episodic meta-learning to enhance the adaptability of the learned GNN encoder and achieve similar results. Furthermore, G-Meta (Huang & Zitnik, 2020), GFL-KT (Yao et al., 2020) and MI-GNN (Wen et al., 2021) use meta-learning to transfer knowledge when other auxiliary graphs exist. TNT (Wang et al., 2022) further takes variance among different meta-tasks into account. All those methods rely on the assumption that there exist *base classes* with substantial labeled nodes per class to sample episodes, thus not working for the general FSNC problem defined in this paper.