HeterGP: Bridging Heterogeneity in Graph Neural Networks with Multi-View Prompting

Fengyu Yan[1,2†], **Xiaobao Wang**[1,2†], **Dongxiao He**[1], **Longbiao Wang**[1], **Jianwu Dang**[3], **Di Jin**[1*]

[1]Tianjin Key Laboratory of Cognitive Computing and Application, College of Intelligence and Computing, Tianjin University, Tianjin, China,

[2]Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen, China,

[3]Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China,

{fengyuyan, wangxiaobao, hedongxiao, longbiao_wang}@tju.edu.cn, jdang@jaist.ac.jp, jindi@tju.edu.cn

Abstract

The challenges tied to unstructured graph data are manifold, primarily falling into node, edge, and graph-level problem categories. Graph Neural Networks (GNNs) serve as effective tools to tackle these issues. However, individual tasks often demand distinct model architectures, and training these models typically requires abundant labeled data, a luxury often unavailable in practical settings. Recently, various "prompt tuning" methodologies have emerged to empower GNNs to adapt to multi-task learning with limited labels. The crux of these methods lies in bridging the gap between pre-training tasks and downstream objectives. Nonetheless, a prevalent oversight in existing studies is the homophily-centric nature of prompt tuning frameworks, disregarding scenarios characterized by high heterogeneity. To remedy this oversight, we introduce a novel prompting strategy named HeterGP tailored for highly heterophilic scenarios. Specifically, we present a dual-view approach to capture both homophilic and heterophilic information, along with a prompt graph design that encompasses token initialization and insertion patterns. Through extensive experiments conducted in a few-shot context encompassing node and graph classification tasks, our method showcases superior performance in highly heterophilic environments compared to state-of-the-art prompt tuning techniques.

Introduction

Graph Neural Networks (GNNs) find broad applications across diverse domains. For example, anomaly detection leverages node classification (Ding et al. 2019; Wang et al. 2023), recommendation systems benefit from link prediction (He et al. 2020; Jin et al. 2021), and the domains of molecules (Kim et al. 2022) and drug discovery (Liu et al. 2023a) witness promising applications of graph classification and generation. Recent studies have predominantly concentrated on specific tasks, primarily relying on supervised graph training methodologies. However, numerous real-world scenarios suffer from a scarcity of annotated data (Ding et al. 2024), posing significant challenges to the deployment of supervised graph training techniques.

To address these issues, recent work has drawn inspiration from pre-training models in NLP, which have demonstrated excellent performance on downstream tasks (Qiu et al. 2020; Devlin et al. 2019; Radford et al. 2018). Efforts have been made to design pre-training methods for graphs. However, unlike the alignment of upstream and downstream objectives in NLP, there's often a significant gap between pre-training and downstream tasks in graph-based applications (Sun et al. 2023). For instance, using masked edges as a pre-training objective can result in negative transfer when applied to graph-level classification tasks.

To address the inconsistency between upstream and downstream task objectives and to improve the performance of pre-trained models on different downstream tasks, several studies have explored graph prompt learning methods, inspired by prompt-tuning in NLP (Brown et al. 2020). These methods include the All-in-One approach (Sun et al. 2023), which unifies upstream and downstream tasks by extracting subgraphs and adding multiple learnable nodes to bridge the gap between pre-training models and various downstream tasks. GPPT (Sun et al. 2022) and GraphPrompt (Liu et al. 2023c) construct pre-training tasks centered on link prediction, unifying various downstream tasks into link prediction tasks. They employ category representation vectors and subgraph readout to handle the diversity of downstream tasks.

While the aforementioned methods have shown promising results, they heavily rely on the assumption of graph homophily (most neighboring nodes belong to the same categories and possess similar features). There are mainly two aspects that reflect this assumption, the pre-training process and prompt tuning. As shown in Figure 1(a), most pre-training tasks assume that the similarity between adjacent nodes is greater than that between non-adjacent nodes. Similarly, in the process of prompt tuning, only neighboring nodes are used to obtain subgraph information. In addition, more similar nodes are connected during prompt graph initializing and inserting pattern procedure in the existing methods, shown in Figure 1(b). However, many real-world graph networks deviate from the homophily assumption (Yu et al. 2024), displaying low homophily or even heterophily (most neighboring nodes belong to different categories and exhibit dissimilar features). Such scenarios do not align with the characteristics of graphs with high heterogeneity, leading

[†]These authors contributed equally.

[*]Corresponding author

to limited effectiveness in prompt tuning. Although there is existing research on pre-training for heterophily (Liu et al. 2023b), it difficult to align the objectives of the downstream prompt tuning step.

Addressing the challenges posed by strong heterophily in graphs during the former kinds of prompt-tuning approaches involves several difficulties. Firstly, under traditional message-passing mechanisms, simply using K-hop neighboring node information as in All-in-One (Sun et al. 2023) is insufficient to effectively represent subgraph features in low homophily scenarios. Balancing homophily and heterophily for subsequent prompt tuning calculations is a key challenge. Secondly, after obtaining sampled subgraph information, designing a graph prompt that effectively captures features in both homophilic and heterophilic scenarios using prompt-tuning methods is a significant challenge.

To address these obstacles, we propose the Heterophilic Graph Prompt (HeterGP), which amalgamates heterogeneity in GNNs through multi-view prompting. Recent studies have showcased notable efficacy in utilizing random walks for acquiring homophilic and heterophilic information (Jin et al. 2022). Our strategy confronts the initial challenge by adopting this efficient method to gather insights on both homophily and heterophily. To tackle the subsequent challenge, where a singular prompt proves insufficient for managing both information types, we draw inspiration from NLP practices (Wei et al. 2024) that embrace multi-prompt collaborative frameworks. We introduce two distinct prompt types to individually process homophilic and heterophilic information, thereby transcending the restriction of exclusively considering homophilic information of a single type. Subsequently, these two streams of information are merged, and an attention mechanism is harnessed to discern the significance of harmonizing the two information types. Our approach undergoes rigorous testing across diverse datasets and varied downstream tasks, showcasing robust performance in scenarios involving both homophily and heterophily, as well as in few-shot settings. Our contributions can be outlined as follows:

- We are the first to propose a unique information collection method designed to unify graph tasks that exhibit diverse structural homogeneity and heterogeneity.
- We propose an innovative graph prompt-tuning strategy that unifies the format of graph prompts in both homophilic and heterophilic scenarios through a multi-view design.
- We conduct extensive comparisons of HeterGP against state-of-the-art methods across a range of homophilic and heterophilic datasets in multiple tasks, demonstrating the effectiveness of HeterGP.

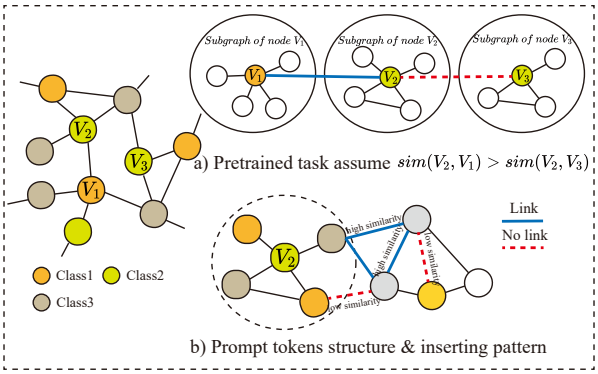

Figure 1: The existing methods include homophilic assumption. (a) Higher similarity is assumed among adjacent nodes during the pre-training process, using K-hop subgraphs to unify tasks and represent node features, and (b) the insertion patterns of the prompt graph also assume that highly similar nodes are connected.

Related Work

Graph Pre-training In NLP, pre-trained models have broad utility across tasks (Wang et al. 2022b). In graph scenarios with limited labeled data, the need for generalization emphasizes the significance of graph pre-trained models. Pre-training relies on self-supervised methods utilizing existing graph data. For edge-level pre-training, techniques like link prediction (Long et al. 2022; Kipf and Welling 2016) are foundational. LaGraph (Xie, Xu, and Ji 2022) reconstructs masked edges to enhance robust representations. Contrastive learning at the node level is a key strategy, with GRACE (Zhu et al. 2020), GCA (Zhu et al. 2021), and HomoGCL (Li et al. 2023) refining this approach. Approaches such as BGRL (Thakoor et al. 2021) treat nodes in augmented graphs as positive samples, training the online encoder to predict the target encoder for potent node representations. Conversely, DGI (Velickovic et al. 2019) emphasizes maximizing mutual information between node and global representations, using augmented graphs as negative samples to enhance node comprehension.

Graph Prompt-tuning Derived from NLP, prompt tuning methods adeptly connect pre-training tasks to targeted downstream goals. Unlike NLP contexts, merging tasks and domains poses unique challenges in the domain of graph prompt frameworks. Among existing methods, GPPT (Sun et al. 2022) converts link prediction pre-training into a node classification task, employing task tokens and structure tokens within the prompt. Similarly, GPF (Fang et al. 2023) augments original nodes with additional learnable features. In GraphPrompt (Liu et al. 2023c), a pre-training task emphasizing increased similarity between neighboring nodes is utilized, then adapted into node and graph classification tasks through a trainable weight matrix in the prompt, adjusting representations post-readout. The All-in-One (Sun et al. 2023) approach integrates prompts comprising nodes as vectors of new learnable features introduced within subgraphs, linked to original graph nodes with high similarity. Conversely, the One-for-All (Liu et al. 2024) incorporates prompts consisting of newly added nodes, their initial representations derived by feeding natural language descriptions into LLM. While the aforementioned methodologies tackle the multi-task graph learning challenge using prompts, they often neglect the diversity inherent in graph structures.

Problem Statement

Given a graph defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}) \in \mathbb{G}$, where $\mathcal{V} = \{v_1, v_2, ..., v_{|\mathcal{V}|}\}$ is the set of nodes and $\mathcal{E} = \{e_1, e_2, ..., e_{|\mathcal{V}|}\}$ is the set of edges. The features of the nodes are defined as $\boldsymbol{X} \in \mathbb{R}^{|\mathcal{V}| \times d}$, and $\boldsymbol{x}_i \in \mathbb{R}^d$ represents the feature of the node v_i. The adjacent matrix is denoted as $\boldsymbol{A} \in \{0,1\}^{|\mathcal{V}| \times |\mathcal{V}|}$, where $\boldsymbol{A}_{ij} = 1$ if $(v_i, v_j) \in \mathcal{E}$.

The graph prompt tuning task involves adapting frozen pre-trained GNN models to downstream tasks. The downstream tasks we consider encompass node classification (NC) and graph classification (GC). Initially, we focus on the NC task, highlighting specific handling approaches for different tasks if necessary. The entire process involves a pre-trained GNN encoder f_{GNN} with frozen parameters, a learnable classifier f_{cls} for downstream tasks and a task-specified graph prompt process $f_{prompt} : \mathbb{G} \to \mathbb{G}$. During the prompt tuning process, $f_{prompt}(\mathcal{G})$ will replace $\mathcal{G}$ as input to f_{GNN}, our objective is to learn the parameter of f_{cls} and f_{prompt} by maximizing the likelihood of predicting the labels for few-shot data Y, can be formulated as:

$$\max_{\theta_{cls}, \theta_{prompt}} P_{\theta_{GNN}, \theta_{cls}}(Y | f_{prompt}(\mathcal{G})),$$

where $\theta_{cls}, \theta_{prompt}, \theta_{GNN}$ represent the parameters for the modules of f_{cls}, f_{prompt}, f_{GNN} respectively.

Proposed Model: HeterGP

In this section, we delineate the implementation of our proposed HeterGP for both homophily and heterophily scenarios, commencing with a framework overview and in-depth elucidations of each module.

Overview Framework

Our objective is to create a graph prompt-tuning framework capable of accommodating both homophily and heterophily across various network topologies. The comprehensive framework diagram is depicted in Figure 2. Inspired from All-in-One (Sun et al. 2023), we leverage node subgraph representations to capture embedded graph information and unify tasks within the graph through subgraph structures. To ensure that subgraphs encapsulate homophilic and heterophilic insights, we initiate a multi-view subgraph sampling strategy to capture dual perspectives of information concerning the target nodes (GC tasks bypass this step). Following the acquisition of homophilic and heterophilic subgraphs for the target node, distinct graph prompt construction and insertion patterns are devised for each subgraph type. Subsequently, the multi-view graphs, enhanced with prompts, are inputted into the pre-trained GNN to extract representations for both perspectives. The amalgamated representation effectively encapsulates homophilic and heterophilic information pertinent to the task at hand. Ultimately, the results of the downstream task are generated through an attention-based classification layer.

Path Extractor & Subgraph Reformer

The tasks handled by graph models are varied. In a multi-task setup, the key step involves unifying these tasks. Current methods predominantly tackle this challenge by reconfiguring downstream tasks at the graph level. Conventional subgraph generation typically relies solely on aggregating the K-hop neighbors of pertinent nodes. The aggregation technique commonly combines information from nodes across various classes in graphs characterized by high heterophily, resulting in subpar performance when applied to real-world prediction tasks. To tackle the challenges posed by the topological structure variations of homophily and heterophily, the primary focus should be on efficiently gathering neighborhood information that is pertinent to the downstream task at hand. Drawing inspiration from RAWGNN (Jin et al. 2022), Breadth-First Search (BFS) and Depth-First Search (DFS) are pivotal in acquiring representations of homophily and heterophily, respectively. Consequently, we employ a random walk technique incorporating two parameters, p and q, derived from Node2Vec (Grover and Leskovec 2016), to emulate DFS and BFS behaviors.

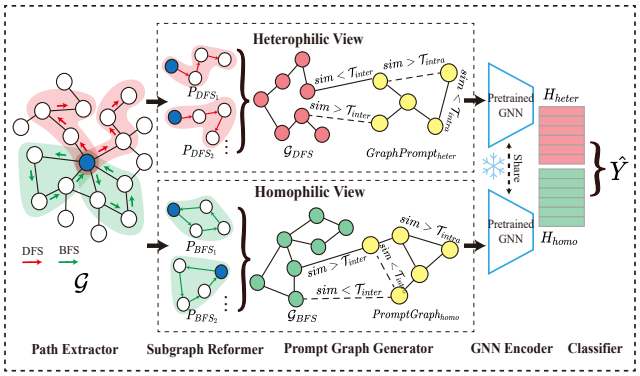

Figure 2: The framework of the proposed HeterGP. Initially, the Path Extractor is utilized to gather multiple paths and create subgraphs representing homophilic and heterophilic viewpoints (with GC tasks exempt from this process). Subsequently, the Prompt Graph Generator is employed to formulate prompts and establish a fresh computational graph. Dual-perspective representations are derived through the GNN encoder, with the downstream task loss utilized during the training phase.

Assuming the transition from node v_k to v_j has occurred, the current node at v_j is contemplating the next step to take. Rather than choosing the subsequent neighbor from v_j randomly with equal likelihood, the initial weights are adjusted using parameters p and q. These weights are then normalized to establish the probabilities associated with choosing various nodes, which can be expressed as:

$$P_{pq}(v_i \mid v_j, v_k) = \begin{cases} 1/p \text{ if } d_{ik} = 0 \\ 1 \quad \text{if } d_{ik} = 1 \\ 1/q \text{ if } d_{ik} = 2 \\ 0 \quad \text{otherwise} \end{cases}, \tag{1}$$

where $d_{ik} = 0$ indicates that v_j returns to the starting point v_k, $d_{ik} = 1$ indicates that the next neighbor from the new starting point v_j has an edge relationship with the previous starting point v_k. In this case, the distance is 1, and $P_{pq} = 1$. $d_{ik} = 2$ indicates that the next neighbor from the new start-

ing point v_j has no relationship with the previous starting point v_k. When $p < 1$ and $q > 1$, the procedure turns to be BFS, while $p > 1$ and $q < 1$ results in a DFS-like behavior.

Typically, the BFS strategy is akin to the traditional method of extracting subgraphs from K-hop neighbors, which captures homophily information. In contrast, the DFS strategy tends to explore further from the source node, enabling the extraction of heterophily information. Both types of information are crucial, especially in heterophilous scenarios, where the DFS strategy can aggregate more information about nodes belonging to the same class. Therefore, we selected both strategies. We perform n random walks of length l starting from node v to obtain multiple BFS and DFS paths, respectively. This process can be expressed as:

$$\mathcal{P}_l^j = \left(v_0^j, v_1^j, \ldots, v_l^j\right), \quad \forall j \in \{1, 2, \ldots, n\}, \tag{2}$$

where $\mathcal{P}_l^j$ denotes collection of nodes from the j-th random walk of length l. The paths are aggregated separately to generate subgraphs under two different views which can be formulated as:

$$\begin{aligned} \mathcal{V}' = \bigcup_{j=1}^{n} \left\{ v_i^j \mid i \in \mathcal{W}_l^j \right\}, \mathcal{E}' = \{(u, v) \in \mathcal{E} \mid u, v \in \mathcal{V}'\} \\ \mathcal{G}_{BFS} = (\mathcal{V}_{BFS}, \mathcal{E}_{BFS}), \mathcal{G}_{DFS} = (\mathcal{V}_{DFS}, \mathcal{E}_{DFS}), \end{aligned} \tag{3}$$

where $\mathcal{V}'$, $\mathcal{E}'$ be the set of unique nodes and edges visited in all n walks. At this point, the subgraph containing multi-view information is obtained for subsequent prompt learning steps. The aforementioned steps are applied to the NC problem. In the case of GC tasks, where our area of interest is the entire graph, we use the whole graph directly as the sampling result for the subsequent prompt learning steps.

Graph Prompt Generator

This module encompasses the architecture design of the internal connections within the prompt subgraph and the connections between the prompt subgraph and the main subgraph. Initially, we focus on the design of graph prompts, inspired by the concept of "prompt as tokens" (Sun et al. 2023, 2022). Prompt tokens can be perceived as freshly introduced nodes, with their quantity defined as a hyperparameter, and their dimensions align with those of the original node features, denotes $X_{prompt}, X_{node} \in \mathbb{R}^d$.

Intra-connection After initializing the prompt tokens, the subsequent step entails crafting the internal structure within these tokens. To accommodate diverse homophilic and heterophilic scenarios, two distinct internal connection patterns are formulated for the prompt graph. In the context of homophily, we calculate the similarity S_{ij} between prompt tokens i and j, forming connections between token nodes where the similarity surpasses a predefined threshold. Conversely, in scenarios of heterophily, we assess similarities between tokens, linking token nodes where the similarity falls below another specified threshold:

$$\begin{aligned} \boldsymbol{A}_{ij}^{homo\_pg} &= \begin{cases} S_{ij} \text{ if } S_{ij} > \mathcal{T}_{intra\_homo} \\ 0 \text{ otherwise} \end{cases} \\ \boldsymbol{A}_{ij}^{heter\_pg} &= \begin{cases} S_{ij} \text{ if } S_{ij} < \mathcal{T}_{intra\_heter} \\ 0 \text{ otherwise} \end{cases} \end{aligned}, \tag{4}$$

where $\boldsymbol{A}^{homo\_pg}, \boldsymbol{A}^{heter\_pg}$ capture the adjacency matrix of the prompt graph for homophily and heterophily. These unique internal construction approaches guarantee that the distributions of homophily and heterophily correspond to the respective subgraphs from two distinct perspectives.

Inter-connection We have generated two prompt graphs from different perspectives. To ensure that the addition of prompts aligns with the distribution of structure and the purpose of the previously generated subgraphs, we use the following method: For the subgraph generated by BFS, we calculate the similarity S_{ij} between each node in the prompt graph and the nodes in the BFS-generated subgraph. We set a threshold $\mathcal{T}_{inter\_homo}$ and connect nodes with similarity greater than the threshold. Conversely, we calculate the similarity S_{ij} with the nodes in the DFS-generated subgraph and connect nodes with a similarity less than the threshold. The process can be expressed as:

$$\begin{aligned} \boldsymbol{A}_{ij}^{homo} &= \begin{cases} S_{ij} \text{ if } S_{ij} > \mathcal{T}_{inter\_homo} \\ 0 \text{ otherwise} \end{cases} \\ \boldsymbol{A}_{ij}^{heter} &= \begin{cases} S_{ij} \text{ if } S_{ij} < \mathcal{T}_{inter\_heter} \\ 0 \text{ otherwise} \end{cases} \end{aligned}, \tag{5}$$

where $\mathcal{G}_{homo}$ and $\mathcal{G}_{heter}$ symbolize the graph resulting from the amalgamation of prompt graphs and subgraphs, with $\boldsymbol{A}^{homo}$ and $\boldsymbol{A}^{heter}$ representing their corresponding adjacency matrices. Consequently, we finalize the establishment of the prompt graph linked to the integrated task subgraph, culminating in a graph enriched with prompts.

Attention-based classifier

The pretraining techniques are outlined in the experiments section, where the frozen pretraining model encodes sub-

Algorithm 1: Proposed model HeterPG

Input: Graph $\mathcal{G}$, pre-trained GNN model f_{GNN}, k-shot based labeled dataset $\mathcal{D} = \{(v_i, y_i), i = 1, 2, ..., k\}$
Output: The optimal task-specified model f_{prompt}, parameterized θ_{prompt} and classifier f_{cls}

Initialize θ_{cls}, θ_{prompt} and downstream task loss $\mathcal{L}_{down}$
while not converged **do**
  Sample subgraphs $\mathcal{S}_{v_i}, v_i \in \mathcal{D}$, based on Eq.1,2,3
  for each $S_{v_i} \in S$ **do**
    f_{prompt} inner structure update based on Eq.4
    $\mathcal{G}_{homo}, \mathcal{G}_{heter} \leftarrow f_{prompt}(S_{v_i})$, based on Eq.5
    $\mathcal{L} \leftarrow \mathcal{L}_{down}(f_{cls}(\mathcal{G}_{homo}, \mathcal{G}_{heter}), y_i)$
  end for
  Update θ_{cls} and θ_{prompt} by minimize $\mathcal{L}$
end while
return θ_{prompt} and θ_{cls}

Dataset	#N	#E	#F	#C	L.H.R
Cora	2,708	10,556	1,433	7	0.81
CiteSeer	3,327	9,104	3,703	6	0.74
PubMed	19,717	88,648	500	3	0.80
Texas	183	325	1,703	5	0.09
Wisconsin	251	515	1,703	5	0.19
Cornell	183	298	1,703	5	0.13
Actor	7,600	30,019	932	5	0.22
Chameleon	2,277	36,051	2,325	5	0.23
Squirrel	5,201	216,933	2,089	5	0.22

Table 1: Summary of NC datasets

graphs that represent homophilic and heterophilic information. To amalgamate these representations and ascertain their significance for downstream tasks, a straightforward MLP layer is employed:

$$\mathcal{H}_{\text{final}} = \sigma\left(\mathcal{W}(\mathcal{H}_{homo} \| \mathcal{H}_{heter}) + \mathbf{b}\right), \tag{6}$$

where $\mathcal{H}_{homo}$ and $\mathcal{H}_{heter}$ denote the representations of $\mathcal{G}_{homo}$ and $\mathcal{G}_{heter}$. The MLP layer discerns the significance of homophily and heterophily information within the given data domain, generating a conclusive representation applicable for subsequent tasks. The comprehensive training process is detailed in Algorithm 1.

Experiments

In our experiments, we compare our model with traditional GNN methods and state-of-the-art graph prompt models on node classification (NC) and graph classification (GC) tasks. We utilize real-world datasets that predominantly encompass both homophily and heterophily scenarios.

Experimental Setup

Datasets We test downstream tasks using the following datasets, which can be found in Table 1 & 2. For NC, the datasets with strong homophily: Cora, CiteSeer, and PubMed (Yang, Cohen, and Salakhutdinov 2016) are citation network datasets. Datasets with strong heterophily: Cornell, Texas, and Wisconsin (Pei et al. 2020) are web datasets collected from the computer science departments of the respective universities. The Actor dataset (Tang et al. 2009) is a actor co-occurrence network. The Chameleon and Squirrel datasets (Pei et al. 2020) are web page datasets.

For the GC tasks, we utilize diverse datasets. ENZYMES (Wang et al. 2022a) comprises protein structures classified into six enzyme classes. PROTEINS (Borgwardt et al. 2005) represents secondary structure elements for protein classification. MUTAG (Morris et al. 2020) focuses on classifying chemical compounds as mutagenic or non-mutagenic. COX2 (Rossi and Ahmed 2015) involves compounds classified by their activity against the COX-2 enzyme, and BZR (Rossi and Ahmed 2015) for bioactivity against the benzodiazepine receptor. Additionally, we calculate the *L.H.R* values for each dataset, representing the label-defined edge homophily ratio of the network.

Dataset	#G	Avg.#N	Avg.#E	#F	#C	L.H.R
ENZYMES	600	32.6	124.3	3	6	0.67
PROTEINS	1,113	39.1	145.6	3	2	0.66
MUTAG	188	17.9	39.6	7	2	0.72
COX2	467	41.22	43.45	3	2	0.41
BZR	405	35.75	38.36	3	2	0.41

Table 2: Summary of GC datasets

Baselines We compare three types of methods: 1) End-to-end graph neural networks such as GCN (Kipf and Welling 2017), GAT (Veličković et al. 2018) and Graph-Transformer(GT) (Shi et al. 2021). The first two methods primarily focus on aggregating messages from neighboring nodes. In contrast, GT considers global information, demonstrating superior performance on heterophily graphs (Müller et al. 2023). 2) Pre-trained models+fine-tuning, where unsupervised learning is performed using contrastive learning methods, followed by fine-tuning in a few-shot setting. For contrastive learning, we use methods like GraphCL (You et al. 2020) and SimGRACE (Xia et al. 2022). 3) Pre-trained models+prompt-tuning. This type of method consists of two stages: graph pre-training and then prompt-tuning using graph prompt methods in a few-shot setting. The prompt-tuning methods for comparison include All-in-One (Sun et al. 2023) and GraphPrompt (Liu et al. 2023c). These methods incorporate homophily assumptions either in the pre-training methods or in the prompt design.

Parameter Settings To assess the model's capacity in a few-shot context, the dataset is randomly divided into training and testing sets following the n-way k-shot setup. In the experiments, both 100-shot and 5-shot scenarios are used for NC, while 20-shot and 5-shot scenarios are employed for GC. To ensure fair comparisons, the training set is randomly sampled 10 times for training. The hyperparameters in the comparative models are set according to the original paper. In our model, a random walk step size of 10 is chosen, with 6 paths collected during random walks. BFS and DFS are utilized in the random walk strategy with probabilities $p = 0.1$ and $q = 10$, respectively. For prompt calculation, 10 prompt tokens are used, with internal similarity thresholds set to 0.7 in the homophilic perspective and 0.4 in the heterophilic perspective. In the pre-training phase, consistency is maintained by keeping the structure of the GNN encoder the same across pre-trained and prompt methods in baseline models. GT is selected as the backbone for our encoder. The learning rate for training classifiers and prompt tokens is set to 0.001, using the Adam optimizer from PyTorch. All experiments are conducted on Nvidia RTX 3090 graphics cards.

Performance Evaluation

We conduct experiments on 14 real-world datasets, including scenarios of NC and GC, to compare the learning abilities of various methods under few-shot settings. We use mean accuracy and standard deviation as evaluation metrics. Overall, HeterPG achieves the best performance on 19 out of 25 different settings. When comparing prompt categories, it excels in 21 cases. Here are some specific findings:

Method	GCN	GAT	GT	GraphCL+FT	simGRACE+FT	GraphPrompt	All in One	Ours
				100-shot				
Cora	77.46 ± 0.66	**79.79 ± 0.71**	79.73 ± 0.50	70.10 ± 2.47	69.12 ± 4.68	70.46 ± 2.21	71.52 ± 1.56	77.60 ± 2.43
CiteSeer	74.38 ± 0.73	74.68 ± 1.02	74.21 ± 1.30	76.59 ± 6.85	78.44 ± 1.82	67.93 ± 0.59	80.62 ± 0.36	**84.54 ± 0.65**
PubMed	**78.90 ± 0.54**	77.58 ± 1.13	77.15 ± 0.53	62.92 ± 1.32	64.22 ± 3.23	71.84 ± 0.51	64.91 ± 3.49	66.71 ± 1.44
Actor	23.31 ± 1.02	25.94 ± 0.68	30.11 ± 0.75	27.44 ± 0.77	28.76 ± 1.64	31.09 ± 1.44	31.87 ± 1.91	**33.97 ± 2.66**
Chameleon	38.57 ± 1.68	45.22 ± 1.67	48.42 ± 1.15	41.79 ± 6.23	40.92 ± 5.34	47.64 ± 3.09	44.67 ± 6.05	**49.17 ± 4.87**
Squirrel	26.14 ± 1.48	32.80 ± 2.24	33.50 ± 0.66	29.43 ± 4.15	30.29 ± 3.76	33.42 ± 2.04	31.41 ± 4.66	**34.53 ± 0.88**
				5-shot				
Cora	66.50 ± 1.93	69.50 ± 1.50	65.15 ± 1.51	60.71 ± 8.72	61.51 ± 7.61	53.42 ± 10.82	65.76 ± 9.65	**70.57 ± 7.56**
CiteSeer	53.62 ± 3.25	61.90 ± 8.01	51.95 ± 3.21	61.13 ± 8.51	59.28 ± 9.32	56.72 ± 2.49	66.52 ± 6.42	**68.33 ± 5.53**
PubMed	61.43 ± 4.26	62.43 ± 7.07	59.37 ± 4.58	56.83 ± 8.51	58.27 ± 5.48	**64.86 ± 3.47**	63.77 ± 7.44	64.43 ± 5.28
Texas	53.91 ± 3.16	51.72 ± 2.23	65.00 ± 4.88	54.04 ± 8.02	56.12 ± 7.12	56.60 ± 12.25	61.29 ± 3.99	**66.67 ± 4.78**
Wisconsin	46.51 ± 9.30	42.06 ± 6.35	53.71 ± 7.09	53.80 ± 7.68	51.75 ± 4.62	53.11 ± 7.83	64.11 ± 0.84	**65.09 ± 1.98**
Cornell	41.72 ± 6.32	42.66 ± 7.68	50.16 ± 6.00	58.86 ± 11.37	56.46 ± 8.48	45.97 ± 12.34	58.78 ± 8.77	**64.58±1.86**
Actor	20.55 ± 3.56	23.39 ± 2.34	23.49 ± 2.87	22.97 ± 3.91	23.65 ± 4.58	**30.45 ± 2.06**	23.50 ± 6.26	27.49 ± 3.89
Chameleon	30.07 ± 2.66	27.88 ± 3.56	31.07 ± 4.81	29.20 ± 3.97	30.82 ± 4.17	30.20 ± 11.60	33.59 ± 19.57	**35.29 ± 7.87**
Squirrel	21.97 ± 1.01	20.91 ± 0.67	27.07 ± 2.30	27.30 ± 5.36	26.73 ± 3.74	31.06 ± 2.90	29.82 ± 1.52	**32.41 ± 3.42**

Table 3: Performance on node classification with best **bolded**, runner-up underlined and third place dash lined.

Few-shot Node classification In NC task, We train the model on 10 randomly generated k-shot tasks, and the results are summarized in Table 3, revealing the following findings: 1) In scenarios with both abundant and limited training samples on heterophilic datasets, our method outperforms all baseline methods, demonstrating the effectiveness of our approach in handling heterophilic information. 2) Our method consistently outperforms most of the baseline methods, regardless of dataset homophily or heterophily and different sample setups. Particularly, compared with All-in-One, our method has a significant advantage in the Actor and Cornell heterophilic datasets. This indicates that our subgraph generator and inserting patterns for heterophilic datasets is effective. 3) In 5-shot scenarios, our method exhibits better performance compared to traditional GNN models. This suggests that pre-training plays a crucial role within our framework. 4) Contrasting the pre-trained model fine-tuning paradigms, prompt tuning consistently produces superior outcomes, suggesting that the added prompt graph plays a beneficial role. 5) In the 100-shot setting, the end-to-end method performs better in two homophilic datasets. This trend can be ascribed to the substantial scale of the training data, which is particularly favorable for such models, given the distinctiveness of node features. Moreover, these methods are inherently tailored for this specific structure.

Few-shot Graph Classification In GC task, the test results are presented in Table 4, revealing the following findings: 1) Our method performs on par with or even outperforms traditional GNNs in several datasets, indicating that the framework is not only applicable to node-level problems but also effective as a unified model. 2) Compared to the All-in-One approach, our method shows significant improvements, especially in ENZYMES and BZR datasets, suggesting the effectiveness of incorporating multi-perspective prompts in this task. 3) Under the condition of limited samples, the performance of our model does not show a significant improvement compared to traditional methods, unlike in NC. This could be related to the dataset size and characteristics. Following experiments will explore different quantities of k in the k-shot setting.

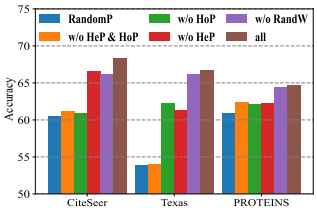

Figure 3: Ablation study

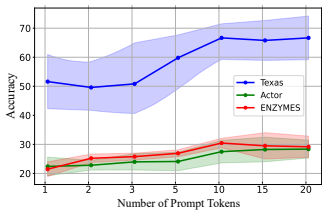

Figure 4: Impact of #tokens

Model Analysis

To gain a better understanding of the role of modules within the model and the impact of hyperparameters, we conduct the following experiments including ablation study and hyper-parameter analysis.

Ablation Study To better illustrate the effectiveness of each module in our method, we conduct an ablation study to evaluate the variables in the entire framework. **w/o HeP**: Removing the heterophilic perspective prompt from our framework; **w/o HoP**: Removing the homophilic perspective prompt from our framework; **w/o HeP & HoP**: Removing both the heterophilic and homophilic perspective prompt from our framework; **w/o RandW**: Removing subgraph extractor from our model, using K-hop subgraphs instead; **RandomP**: Using random parameters for the multi-views prompts in our framework, without training. We select the homophilic dataset CiteSeer and the heterophilic dataset Texas for NC, as well as the PROTEINS dataset for GC, under the 10-shot setting. The results are shown in Figure 3. We can see that HoP and HeP play different roles in homophily and heterophily NC datasets, which means integrating information from both designed prompts is effective. Train-

Method	GCN	GAT	GT	GraphCL+FT	simGRACE+FT	GraphPrompt	All in One	Ours
				20-shot				
ENZYMES	21.76 ± 3.16	19.33 ± 2.27	22.38 ± 3.43	18.87 ± 1.99	19.25 ± 2.29	18.67 ± 3.39	18.62 ± 1.40	**23.33 ± 2.40**
PROTEINS	60.82 ± 6.75	57.12 ± 9.16	64.18 ± 4.05	60.87 ± 8.31	59.04 ± 9.27	58.02 ± 11.34	64.62 ± 6.05	**66.29 ± 10.27**
MUTAG	66.87 ± 8.00	69.92 ± 1.64	68.09 ± 2.23	62.53 ± 11.08	64.26 ± 9.73	**74.67 ± 4.14**	72.00 ± 3.45	73.33 ± 4.27
COX2	68.56 ± 3.59	66.50 ± 2.98	63.37 ± 3.69	58.69 ± 9.69	62.33 ± 9.55	63.35 ± 7.75	67.10 ± 4.70	**69.20 ± 5.15**
BZR	**75.90 ± 6.40**	74.77 ± 4.43	77.53 ± 2.35	60.56 ± 15.58	58.92 ± 7.21	64.34 ± 7.63	63.64 ± 5.38	68.00 ± 5.11
				5-shot				
ENZYMES	16.71 ± 3.01	15.90 ± 4.13	19.14 ± 2.27	19.17 ± 0.92	17.17 ± 2.34	28.54 ± 3.56	27.92 ± 1.26	**30.46 ± 1.65**
PROTEINS	55.85 ± 10.66	48.78 ± 18.46	56.87 ± 6.51	62.38 ± 6.88	61.32 ± 10.25	60.29 ± 12.67	62.28 ± 6.09	**64.60 ± 10.24**
MUTAG	61.79 ± 6.15	67.40 ± 3.53	66.26 ± 5.58	61.87 ± 12.23	65.46 ± 8.44	66.20 ± 3.74	70.67 ± 12.50	**72.73 ± 6.19**
COX2	50.06 ± 9.18	50.20 ± 20.93	51.35 ± 8.60	50.40 ± 20.33	51.87 ± 7.45	51.15 ± 9.17	59.01 ± 3.83	**60.26 ± 6.97**
BZR	56.16 ± 11.07	54.29 ± 18.57	62.78 ± 7.58	57.84 ± 11.61	59.86 ± 12.33	56.28 ± 12.36	60.30 ± 7.54	**63.10 ± 6.47**

Table 4: Performance on graph classification with best **bolded** and runner-up underlined.

able prompt tokens significantly outperform random, non-trainable tokens. In the GC task, the information collection and prompt design from both perspectives are also effective.

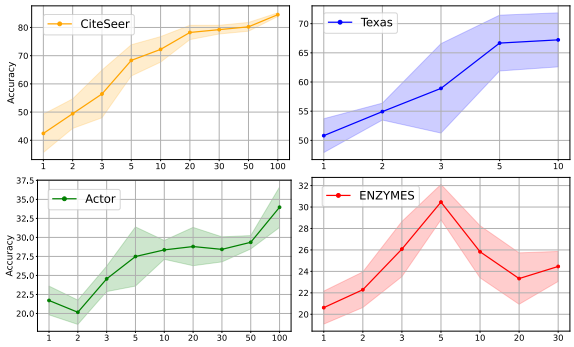

Figure 5: Impact of #shots analysis

Hyper-parameters Analysis We evaluate several key hyper-parameters affecting the effectiveness of our model HeterGP. 1) **Number of prompt tokens**: We test this on datasets of different scales, Texas and Actor for NC and ENZYMES for GC. The results are shown in Figure 4. For datasets of different scales, the performance reaches a bottleneck as the number of tokens increases. This implies that only a small number of tokens are needed to achieve good performance, thereby reducing the number of parameters that need to be trained. 2) **Number of few-shot samples**: We test this with different settings across various datasets and tasks. The results are shown in Figure 5. It can be observed that under the few-shot setting for NC, HeterGP improves with an increase in the number of samples, and the test variance decreases. However, in the GC task, an increase in the number of samples does not consistently lead to improved performance. Instead, performance tends to deteriorate with a larger sample size. This trend can be attributed to the random selection of a limited number of high-quality samples, which happened to include representative graphs of the class, thereby enhancing classification performance. In contrast, in the 20-shot scenario, the inclusion of more low-quality samples resulted in a decline in performance.

Dataset	CiteSeer	Texas	Actor
GCL(GCN) + FT	63.43 ± 5.28	52.76 ± 6.77	22.53 ± 5.78
GCL(GT) + FT	61.13 ± 8.51	54.04 ± 8.02	22.97 ± 3.91
GCL(GCN) + ProG	62.07 ± 5.35	54.37 ± 7.48	22.62 ± 1.22
GCL(GT) + ProG	66.52 ± 6.42	61.29 ± 3.99	23.50 ± 6.26
GCL(GCN) + HeterGP	**70.35 ± 2.15**	61.95 ± 4.74	26.96 ± 4.81
GCL(GT) + HeterGP	68.33 ± 5.53	**66.67 ± 4.78**	**27.49 ± 3.89**

Table 5: Impact of pre-trained model with best **bolded**.

Pre-train Model Analysis To assess the effectiveness of our prompt approach with diverse pre-training strategies in homophilic and heterophilic contexts, we conduct experiments utilizing the contrastive learning method GraphCL (GCL) with various GNN encoder backbones like GCN and GT. GCN focuses on local neighborhood information, whereas GT captures broader global perspectives. The results are presented in Table 5, demonstrating that our approach attains superior performance across both pre-training models. Particularly noteworthy is the significant accuracy improvement in heterophilic datasets, where our prompt boosts performance even when using GCN as the backbone, contrasting with prompts solely grounded in homophily assumptions that exhibit limited efficacy.

Conclusion

In this research, we delve into the fine-tuning of graph prompts in scenarios characterized by both homophily and heterophily within a multi-task few-shot framework. Our study presents a multi-view strategy aimed at encapsulating both forms of information, with a focus on optimizing information acquisition and prompt graph architecture. Through our experiments, we unveil the efficacy of selectively integrating tunable token information via tailored prompts across diverse graph configurations. Moving forward, our research opens avenues for exploring the adaptability and robustness of prompt tuning across more complex graph structures, including those with evolving dynamics. Furthermore, exploring the potential of incorporating domain-specific knowledge into prompt design is a promising avenue for extending this work.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62422210, 92370111, 62302333, 62272340, 62276187), the Open Research Fund from Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ) (No.GML-KF-24-16) and Hebei Natural Science Foundation (No.F2024202047).

References

Borgwardt, K. M.; Ong, C. S.; Schönauer, S.; Vishwanathan, S. V. N.; Smola, A. J.; and Kriegel, H.-P. 2005. Protein function prediction via graph kernels. *Bioinformatics*, 21: i47–i56.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, 1877–1901.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186.

Ding, K.; Li, J.; Bhanushali, R.; and Liu, H. 2019. *Deep Anomaly Detection on Attributed Networks*, 594–602. Society for Industrial and Applied Mathematics.

Ding, K.; Nouri, E.; Zheng, G.; Liu, H.; and White, R. 2024. Toward Robust Graph Semi-Supervised Learning Against Extreme Data Scarcity. *IEEE Transactions on Neural Networks and Learning Systems*, 1–10.

Fang, T.; Zhang, Y.; YANG, Y.; Wang, C.; and Chen, L. 2023. Universal Prompt Tuning for Graph Neural Networks. In *Advances in Neural Information Processing Systems*, volume 36, 52464–52489.

Grover, A.; and Leskovec, J. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 855–864.

He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; and Wang, M. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 639–648.

Jin, D.; Wang, R.; Ge, M.; He, D.; Li, X.; Lin, W.; and Zhang, W. 2022. RAW-GNN: RAndom Walk Aggregation based Graph Neural Network. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, 2108–2114.

Jin, D.; Wang, X.; He, D.; Dang, J.; and Zhang, W. 2021. Robust Detection of Link Communities With Summary Description in Social Networks. *IEEE Trans. Knowl. Data Eng.*, 33(6): 2737–2749.

Kim, Y.; Jeong, Y.; Kim, J.; Lee, E. K.; Kim, W. J.; and Choi, I. S. 2022. MolNet: A Chemically Intuitive Graph Neural Network for Prediction of Molecular Properties. *Chemistry – An Asian Journal*, 17(16): e202200269.

Kipf, T. N.; and Welling, M. 2016. Variational Graph Auto-Encoders. *CoRR*, abs/1611.07308.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017*.

Li, W.; Wang, C.; Xiong, H.; and Lai, J. 2023. HomoGCL: Rethinking Homophily in Graph Contrastive Learning. In Singh, A. K.; Sun, Y.; Akoglu, L.; Gunopulos, D.; Yan, X.; Kumar, R.; Ozcan, F.; and Ye, J., eds., *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023*, 1341–1352.

Liu, C.; Fan, W.; Liu, Y.; Li, J.; Li, H.; Liu, H.; Tang, J.; and Li, Q. 2023a. Generative Diffusion Models on Graphs: Methods and Applications. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, 6702–6711.

Liu, H.; Feng, J.; Kong, L.; Liang, N.; Tao, D.; Chen, Y.; and Zhang, M. 2024. One For All: Towards Training One Graph Model For All Classification Tasks. In *The Twelfth International Conference on Learning Representations*.

Liu, Y.; Zheng, Y.; Zhang, D.; Lee, V. C. S.; and Pan, S. 2023b. Beyond Smoothing: Unsupervised Graph Representation Learning with Edge Heterophily Discriminating. In Williams, B.; Chen, Y.; and Neville, J., eds., *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023*, 4516–4524.

Liu, Z.; Yu, X.; Fang, Y.; and Zhang, X. 2023c. GraphPrompt: Unifying Pre-Training and Downstream Tasks for Graph Neural Networks. In *Proceedings of the ACM Web Conference 2023*, 417–428.

Long, Y.; Wu, M.; Liu, Y.; Fang, Y.; Kwoh, C. K.; Chen, J.; Luo, J.; and Li, X. 2022. Pre-training graph neural networks for link prediction in biomedical networks. *Bioinformatics*, 38(8): 2254–2262.

Morris, C.; Kriege, N. M.; Bause, F.; Kersting, K.; Mutzel, P.; and Neumann, M. 2020. TUDataset: A collection of benchmark datasets for learning with graphs. In *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020)*.

Müller, L.; Galkin, M.; Morris, C.; and Rampásek, L. 2023. Attending to Graph Transformers. *CoRR*, abs/2302.04181.

Pei, H.; Wei, B.; Chang, K. C.-C.; Lei, Y.; and Yang, B. 2020. Geom-GCN: Geometric Graph Convolutional Networks. In *International Conference on Learning Representations*.

Qiu, X.; Sun, T.; Xu, Y.; Shao, Y.; Dai, N.; and Huang, X. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63: 1872–1897.

Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I.; et al. 2018. Improving language understanding by generative pre-training.

Rossi, R. A.; and Ahmed, N. K. 2015. The Network Data Repository with Interactive Graph Analytics and Visualization. In Bonet, B.; and Koenig, S., eds., *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 4292–4293.

Shi, Y.; Huang, Z.; Feng, S.; Zhong, H.; Wang, W.; and Sun, Y. 2021. Masked Label Prediction: Unified Message Passing Model for Semi-Supervised Classification. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 1548–1554.

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 '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1717–1727.

Sun, X.; Cheng, H.; Li, J.; Liu, B.; and Guan, J. 2023. All in One: Multi-Task Prompting for Graph Neural Networks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Tang, J.; Sun, J.; Wang, C.; and Yang, Z. 2009. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 807–816.

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*.

Velickovic, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2019. Deep Graph Infomax. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

Wang, S.; Dong, Y.; Huang, X.; Chen, C.; and Li, J. 2022a. FAITH: Few-Shot Graph Classification with Hierarchical Task Graphs. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*.

Wang, X.; Dong, Y.; Jin, D.; Li, Y.; Wang, L.; and Dang, J. 2023. Augmenting Affective Dependency Graph via Iterative Incongruity Graph Learning for Sarcasm Detection. In Williams, B.; Chen, Y.; and Neville, J., eds., *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023*, 4702–4710.

Wang, Z.; Mu, C.; Hu, S.; Chu, C.; and Li, X. 2022b. Modelling the Dynamics of Regret Minimization in Large Agent Populations: a Master Equation Approach. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, 534–540.

Wei, W.; Ren, X.; Tang, J.; Wang, Q.; Su, L.; Cheng, S.; Wang, J.; Yin, D.; and Huang, C. 2024. LLMRec: Large Language Models with Graph Augmentation for Recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, 806–815.

Xia, J.; Wu, L.; Chen, J.; Hu, B.; and Li, S. Z. 2022. SimGRACE: A Simple Framework for Graph Contrastive Learning without Data Augmentation. In *Proceedings of the ACM Web Conference 2022*, 1070–1079.

Xie, Y.; Xu, Z.; and Ji, S. 2022. Self-Supervised Representation Learning via Latent Graph Prediction. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, 24460–24477.

Yang, Z.; Cohen, W. W.; and Salakhutdinov, R. 2016. Revisiting Semi-Supervised Learning with Graph Embeddings. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48, 40–48.

You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; and Shen, Y. 2020. Graph Contrastive Learning with Augmentations. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Yu, Z.; Feng, B.; He, D.; Wang, Z.; Huang, Y.; and Feng, Z. 2024. LG-GNN: Local-Global Adaptive Graph Neural Network for Modeling Both Homophily and Heterophily. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI 2024*, 2515–2523.

Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; and Wang, L. 2020. Deep Graph Contrastive Representation Learning. *CoRR*, abs/2006.04131.

Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; and Wang, L. 2021. Graph Contrastive Learning with Adaptive Augmentation. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, 2069–2080.