HGPROMPT: Bridging Homogeneous and Heterogeneous Graphs for Few-shot Prompt Learning (Technical Appendix)

Xingtong Yu[1*], **Yuan Fang**[2†], **Zemin Liu**[3], **Xinming Zhang**[1†]
[1]University of Science and Technology of China, China
[2] Singapore Management University, Singapore
[3] National University of Singapore, Singapore
yxt95@mail.ustc.edu.cn, yfang@smu.edu.sg, zeminliu@nus.edu.sg, xinming@ustc.edu.cn

A Overall Framework with Heterogeneous Graphs for Pre-training

HGPROMPT can be pre-trained using both homogeneous and heterogeneous graphs. While Fig. 2 in the main paper demonstrates an example of employing both graph types for pre-training, we further detail the exclusive use of heterogeneous graphs for this purpose in Fig. I.

To elucidate, given a heterogeneous graph as input, we employ a graph template to transition it into a series of homogeneous graphs for the pre-training phase (see Figs. I(a) and (b)). We then undertake link prediction as the pre-training task (Fig. I(b)). Subsequently, both node classification and graph classification can be executed as downstream tasks on the heterogeneous graphs (Figs. I(c) and (d)).

B Further Descriptions of Baselines

In this section, we present more details for the baselines.

(1) **End-to-end Homogeneous Graph Neural Networks**

- **GCN** (Kipf and Welling 2016): GCN utilizes mean-pooling-based neighborhood aggregation to aggregate messages from neighboring nodes.
- **GAT** (Veličković et al. 2018a): GAT also relies on neighborhood aggregation for end-to-end node representation learning. It differentiates itself by assigning varied weights to neighboring nodes, adjusting their significance in the aggregation.

(2) **End-to-end Heterogeneous Graph Neural Networks**

- **HAN** (Wang et al. 2019): HAN incorporates meta-paths to amalgamate different types of heterogeneity. It employs a hierarchical attention mechanism, capturing both node-level and semantic-level importance.
- **Simple-HGN** (Lv et al. 2021): By employing GAT as the backbone, Simple-HGN further considers edge types when calculating neighbor weights during neighborhood aggregation.

(3) **Homogeneous Graph Pre-training Models**

- **DGI** (Veličković et al. 2018b): DGI is a self-supervised pre-training method on homogeneous graphs. This technique hinges on mutual information (MI) maximization, striving to maximize the estimated MI between locally augmented instances and their global representations.
- **InfoGraph** (Sun et al. 2020): As an extension of DGI, InfoGraph focuses on the graph level tasks. It seeks to maximize the consistency between node and graph embeddings.
- **GraphCL** (You et al. 2020): GraphCL applies diverse graph augmentations for self-supervised learning in order to exploit the structural patterns within graphs. The objective is to maximize the agreement between different augmentations during graph pre-training.

(4) **Heterogeneous Graph Pre-training Models**

- **CPT-HG** (Jiang et al. 2021): CPT-HG also employs contrastive learning during its pre-training phase. Positive and negative samples are distinguished based on the presence of specific type links with a target node. The goal is to enhance the similarity between the target node and positive samples while diminishing the similarity with negative samples.
- **HeCo** (Wang et al. 2021): Adopting a cross-view contrastive mechanism, HeCo learns HIN representations from both network schema and meta-path. It uses a view mask mechanism to derive positive and negative samples.

(5) **Graph Prompt Models**

- **GPPT** (Sun et al. 2022): GPPT adopts a GNN model pre-trained through a link prediction task. It employs a prompt module to structure the downstream node classification task, aligning it with the link prediction format.
- **GraphPrompt** (Liu et al. 2023): GraphPrompt uses subgraph similarity calculation to unify pre-training and downstream tasks such as node and graph classification. A learnable prompt is then refined during the downstream task to assimilate task-specific details.

C Implementation Details of Approaches

Details of baselines. For all the baseline models, we use the codes officially released by their respective authors. We tune

*Work was done while at Singapore Management University.
†Corresponding authors.

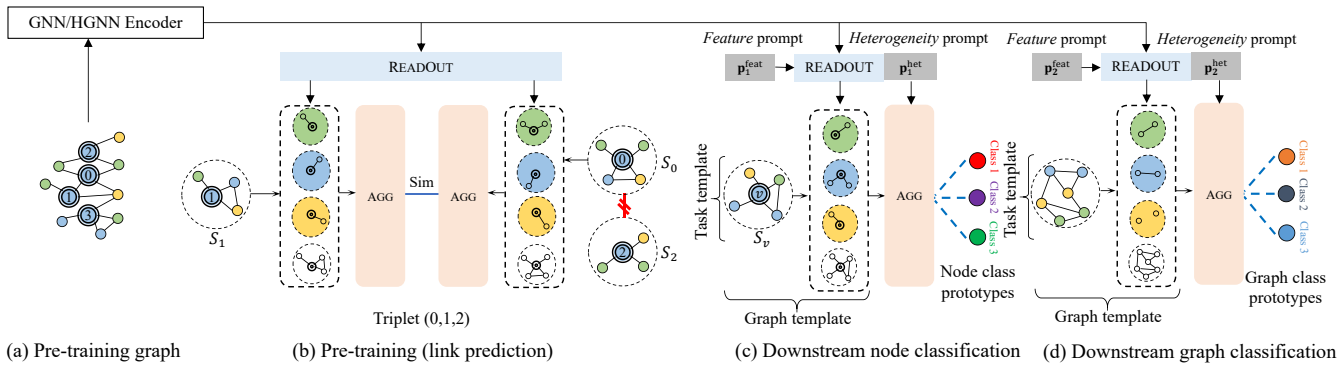

Figure I: Overall framework of HGPROMPT. (a) An input heterogeneous graph for graph pre-training. (b) Pre-training task with link prediction on heterogeneous graphs. (c) Downstream node classification task and (d) graph classification task on heterogeneous graphs.

each model in line with the settings recommended in their literature to ensure optimal performance.

For baseline GCN (Kipf and Welling 2016), we employ a 3-layer architecture, and set the hidden dimensions as 64. For GAT (Veličković et al. 2018a), we employ a 2-layer architecture and set the hidden dimension as 64. Besides, we apply 4 attention heads in the first GAT layer.

For HAN (Wang et al. 2019), we employ a 2-layer GAT with 4 attention heads in the first layer, and set the hidden dimension as 64. In this paper, we use the same meta-paths as utilized in their original paper. For Simple-HGN (Lv et al. 2021), we employ a 2-layer GAT with 4 attention heads in the first layer, and set the hidden dimension as 64. Besides, we utilize elu as the activation function.

For DGI (Veličković et al. 2018b), we use a 3-layer GCN as the base model, and set the hidden dimension as 64. Besides, we utilize prelu as the activation function. For InfoGraph (Sun et al. 2020), we use a 3-layer GCN as the base model, and set its hidden dimensions as 64. For GraphCL (You et al. 2020), we also employ 3-layer GCN as its base model, and set the hidden dimensions as 64. In particular, we choose the augmentations of node dropping and subgraph, with default augmentation ratio 0.2.

For CPT-HG (Jiang et al. 2021), we employ a 2-layer HGT with 1 attention head as its base model, and set the dimension of node representations to 64. In particular, the maximum number of negative samples from inconsistent relations and unrelated nodes are set to 100 and 200, respectively. The maximum number of queued negative samples at subgraph-level is set to 100. For HeCo (Wang et al. 2021), we use one-layer GCN for every meta-path, and we only consider interactions between nodes of the target type and their one-hop neighbors of other types, and set the hidden dimensions as 64.

For GPPT (Sun et al. 2022), we utilize a 2-layer GraphSAGE as its base model, and set the hidden dimension as 64. For this based GraphSAGE, we also utilize the mean aggregator. For GraphPrompt (Liu et al. 2023), we employ a 3-layer GCN as the base model for *ACM* and *Freebase*, and a 2-layer GAT for *DBLP*. We set the hidden dimensions as 64. In addition, we sample 1-hop subgraph for *ACM* and *Freebase*, and 2-hop subgraph for *DBLP*.

Details of HGPROMPT. For our proposed HGPROMPT, we employ a 3-layer GCN as the base model for *ACM* and *Freebase*, and a 2-layer GAT serves as the base model for *DBLP*. We set the hidden dimensions as 64. In particular, the maximum number of negative samples for pre-training is set to 1.

D Further Experiments and Analysis

Few-shot link prediction. Beyond the primary experiments detailed in our main paper, which encompass few-shot node classification and graph classification, we leverage the shared pre-trained model to execute few-shot link prediction on heterogeneous graphs as a subsequent task.

Specifically, in the context of downstream few-shot link prediction, we are provided with k exemplary "heterogeneous links" to guide the prompt learning, namely, k-shot. Each of these heterogeneous links, or edges, denoted as (a, b), comes with associated heterogeneous types for both nodes a and b, as well as the edge itself. Note that while numerous other links may exist within the graph in the downstream link prediction task, their heterogeneity is not specified. Hence, they cannot be employed as supervision in the downstream prompt learning task, given the absence of heterogeneity.

For each target node, we select one of its neighbors to be the positive sample and choose 10 nodes with no connections as the negative samples. The evaluation criteria dictate that the positive sample should rank higher than the negative samples. To assess this, we employ the AUC (Lü and Zhou 2011; Zhang and Chen 2018) and NDCG (Han et al. 2020; Fu et al. 2022) metrics. From each dataset, we randomly pick 10,000 edges dedicated to the downstream task, ensuring these edges remain concealed during pre-training. Specifically, 500 of these edges are allocated for training, another 500 for validation, with the rest for testing. For the evaluative process, we create a 1-shot link prediction tuple (with one positive link and ten negative links) by randomly drawing one sample from the training dataset. We also establish a validation set, maintaining the 1-shot configuration.

Table I: Evaluation on link prediction.

Methods	ACM		DBLP		Freebase	
	AUC (%)	NDCG (%)	AUC (%)	NDCG (%)	AUC (%)	NDCG (%)
GCN	22.30±13.94	31.56±3.95	49.37±11.76	43.40±5.95	75.58±5.42	65.97±4.13
GAT	21.72±8.18	31.14±2.65	37.08±12.15	36.89±5.09	81.54±3.06	70.63±2.42
SIMPLE-HGN	22.70±11.57	31.81±3.58	43.60±10.24	39.66±4.72	71.50±3.76	61.37±2.83
HAN	35.29±9.97	35.39±2.98	56.88±12.35	49.88±5.89	83.85±6.02	71.76±4.86
DGI	46.67±15.74	40.81±5.72	64.88±6.93	54.00±2.82	80.42±5.03	69.71±3.38
GRAPHCL	48.78±13.58	41.49±5.20	66.51±8.09	55.54±3.53	81.46±4.82	74.31±5.19
CPT-HG	52.33±11.98	43.34±3.79	67.31±6.15	56.03±4.94	85.81±7.30	76.29±5.86
HECO	52.14±9.31	43.29±3.20	67.35±5.73	56.60±4.40	86.35±5.17	76.65±4.21
GRAPHPROMPT	54.68±6.39	45.13±3.14	70.41±2.85	58.79±1.60	87.21±3.73	79.55±4.02
HGPROMPT	57.17±5.18	46.14±2.78	72.81±2.19	60.21±1.32	88.96±2.43	80.46±2.91
HGPROMPT+	**58.08**±5.91	**46.88**±3.27	**73.64**±2.94	**60.80**±1.33	**90.04**±3.90	**81.43**±3.74

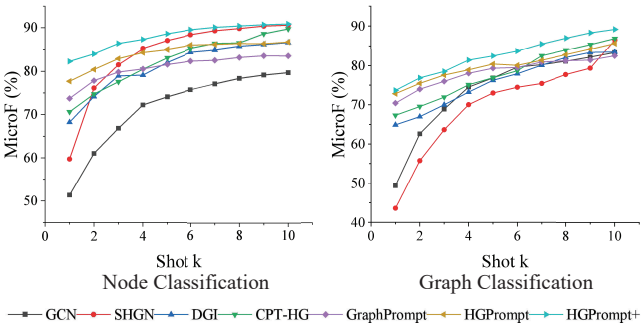

Figure II: Impact of shots on DBLP.

This sampling approach is repeatedly applied, culminating in a collection of 100 distinct few-shot tasks.

The outcomes of few-shot LP are presented in Table I, from which we have several observations. Initially, it is evident that our proposed HGPROMPT consistently outclasses all baselines on *ACM*, *DBLP*, and *Freebase*. This further underscores HGPROMPT's prowess in bridging the gap between homogeneous and heterogeneous graphs. Subsequently, even though HGPROMPT+ still outperforms HGPROMPT, the advantage is not as prominent as it is in NC and GC. This is primarily because there is not a discernible discrepancy between the objectives of pre-training and the downstream task since they are both LP. The sole distinction revolves around the presence or absence of heterogeneity which is adeptly addressed by HGPROMPT.

Performance with different shots. We also tune the number of shots for few-shot node and graph classification tasks to show its impact. Specifically, we experiment with a range of shots, spanning from 1 to 10, and benchmarked against some competitive baselines, including GCN, Simple-HGN, GraphCL, CPT-HG, and GraphPrompt. The task setting remains consistent with our previously established settings for both few-shot NC and GC. For our empirical studies, we focus on the *ACM* and *DBLP*. Notably, *Freebase* was excluded since it lacks a sufficient number of samples for certain classes beyond 3-shot learning. The result on *ACM* and analysis are shown in the main paper (Sect. 5.2). The result on *DBLP* is illustrated in Fig. II. Consistent with the results on *ACM*, HGPROMPT excels over baselines when only a limited amount of labeled data is available. Meanwhile, HGPROMPT+ consistently outperforms all baselines across multiple-shot scenarios.

Table II: HGPROMPT and HGPROMPT+'s performance with various backbones. MiF is short for MicroF (%) and MaF is short for MacroF (%).

Backbones	Methods	Node classification				Graph classification			
		DBLP		Freebase		DBLP		Freebase	
		MiF	MaF	MiF	MaF	MiF	MaF	MiF	MaF
GCN	SUPERVISED	51.36	48.62	17.11	15.35	38.12	35.01	17.38	16.50
	GRAPHPROMPT	65.79	63.32	19.86	18.18	43.58	40.23	19.77	18.55
	HGPROMPT	75.72	71.08	21.26	19.81	44.23	41.68	20.91	19.86
	HGPROMPT+	79.98	76.84	**22.64**	**21.08**	53.29	46.52	**21.70**	**20.91**
GAT	SUPERVISED	65.04	62.83	17.93	16.51	46.47	40.33	16.30	16.01
	GRAPHPROMPT	73.73	69.26	18.39	15.54	52.83	45.68	17.54	17.42
	HGPROMPT	77.69	73.22	20.22	18.26	54.49	46.75	18.86	18.52
	HGPROMPT+	**82.31**	**77.44**	21.48	20.21	**60.53**	**52.02**	19.92	19.34
SHGN	SUPERVISED	59.74	57.34	17.85	16.00	44.56	39.52	17.83	16.71
	GRAPHPROMPT	61.93	58.19	19.06	17.80	49.23	43.84	19.58	18.69
	HGPROMPT	62.81	60.95	19.73	18.26	50.87	45.02	20.33	19.34
	HGPROMPT+	66.05	64.08	21.50	19.74	57.02	49.97	20.58	20.45

Table III: Comparison for parameters on downstream node classification.

Methods	Datasets		
	ACM	DBLP	Freebase
GCN	70,496	1,676,932	11,531,463
Simple-HGN	1,264,838	2,070,600	12,177,806
DGI	192	256	448
CPT-HG	192	256	448
GraphPrompt	64	64	64
HGPROMPT	67	68	71

Performance based on different backbones. The evaluation is conducted on *DBLP* and *Freebase* datasets for NC and GC, as illustrated in Table II. Our observations from this study align with those on *ACM* as illustrated in Sect. 5.3 in the main paper.

Parameter efficiency. We further analyze the number of parameters necessitated for optimization during the downstream task across various representative models. To elucidate, we spotlight the node classification task as a representative example, though findings from other tasks largely mirror these conclusions. These statistics are presented in Table III. Notably, as both GCN and Simple-HGN adopt an end-to-end approach, they invariably require updates to a substantial portion of parameters—essentially, all parameters within their respective models, rendering them the most intensive. On the other hand, models like DGI and CPT-HG, which embrace a "pre-train, fine-tune" paradigm, only mandate updates to the downstream classifier. This results in a significant reduction in the number of parameters subject to modification. It is noteworthy that our proposed HGPROMPT not only eclipses other baselines in performance but also necessitates updates to a considerably leaner set of parameters, second only to GraphPrompt. This efficiency is especially commendable considering the remarkable performance boost achieved with such a modest parameter increment.

References

Fu, H.-M.; Poirson, P.; Lee, K. S.; and Wang, C. 2022. Revisiting neighborhood-based link prediction for collaborative filtering. In *Companion Proceedings of the Web Conference*, 1009–1018.

Han, Z.; Anwaar, M. U.; Arumugaswamy, S.; Weber, T.; Qiu, T.; Shen, H.; Liu, Y.; and Kleinsteuber, M. 2020. Metapath-and entity-aware graph neural network for recommendation. *CoRR, abs/2010.11793*.

Jiang, X.; Lu, Y.; Fang, Y.; and Shi, C. 2021. Contrastive pre-training of GNNs on heterogeneous graphs. In *the ACM International Conference on Information and Knowledge Management*, 803–812.

Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.

Liu, Z.; Yu, X.; Fang, Y.; and Zhang, X. 2023. Graphprompt: Unifying pre-training and downstream tasks for graph neural networks. In *the ACM Web Conference*, 417–428.

Lü, L.; and Zhou, T. 2011. Link prediction in complex networks: A survey. *Physica A: statistical mechanics and its applications*, 390(6): 1150–1170.

Lv, Q.; Ding, M.; Liu, Q.; Chen, Y.; Feng, W.; He, S.; Zhou, C.; Jiang, J.; Dong, Y.; and Tang, J. 2021. Are we really making much progress? Revisiting, benchmarking and refining heterogeneous graph neural networks. In *the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1150–1160.

Sun, F.-Y.; Hoffman, J.; Verma, V.; and Tang, J. 2020. InfoGraph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In *International Conference on Learning Representations*.

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 *the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1717–1727.

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

Veličković, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2018b. Deep graph infomax. In *International Conference on Learning Representations*.

Wang, X.; Ji, H.; Shi, C.; Wang, B.; Ye, Y.; Cui, P.; and Yu, P. S. 2019. Heterogeneous graph attention network. In *The Web Conference*, 2022–2032.

Wang, X.; Liu, N.; Han, H.; and Shi, C. 2021. Self-supervised heterogeneous graph neural network with co-contrastive learning. In *the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1726–1736.

You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; and Shen, Y. 2020. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33: 5812–5823.

Zhang, M.; and Chen, Y. 2018. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31.